*Author:*
**Martin, Nick D**

*Title:*
**Selectivity in Neural Networks**

# University of Bristol Thesis

*Selectivity in Neural Networks*

By

NICHOLAS MARTIN

School of Psychological Science
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY in the Faculty of Life Science.

17TH MARCH 2021

Word count: 69155

# Abstract

It is difficult to interpret individual units in neural networks, which use distributed representations. One way to interpret units is through analysis of their selectivity for an item, feature or class. Authors have drawn different conclusions about the role of units in neural networks based on their selectivity. It has been claimed that highly selective units are harmful, unimportant, important but only for a few classes; or that highly selective units are not part of distributed representations and are in fact localist units.

I present a series of studies investigating how selectivity is measured; factors that may increase selectivity; and the relationship between selectivity and unit importance. I compare the selectivity of a wide variety of models, using several measures of selectivity. I experimentally vary factors that might lead to increased selectivity, namely, the systematicity of input-output mappings and pressure to avoid the superposition catastrophe. I lesion units to infer their importance for particular classes. I show that commonly used selectivity measures are unable to capture the full range of selectivity scores, showing either floor or ceiling effects. Reducing the systematicity of a class or dataset leads to reduced selectivity, while increasing it leads to higher selectivity. However, increased potential for the superposition catastrophe does not lead to an increased number of highly selective units. Finally, I find that although unit importance for a class is weakly associated with class-selectivity, the most important class is rarely the most selective, highlighting the risks of inferring a unit's function from its selectivity.

These results highlight the need for better measures of selectivity; implicate systematicity as a driver of selectivity and demonstrate that claims regarding unit function based on selectivity alone may be unreliable.

# Dedication and acknowledgements

First and foremost I would like to thank Jeff Bowers for inviting me along this path and for the knowledge, enthusiasm and patience he has shown along the way. I want to thank Ella Gale for being a fountain of wide-ranging knowledge and for fixing every computer I've broken! I am grateful to Bobby Stuijfzand, Rowland Baddeley, Gaurav Malholtra, Ben Evans and Ivan Vankov for the wise words and insights that helped when I was stuck. I've had great friends and colleagues along the way. The Upstairs Bears have been warm, fun, generous comrades. Finally, a huge thanks to Will Chapman, Stella Lucia Becci and Phil Collard for all the time chatting in our sanctuary, for listening to my grumbles and celebrating the successes. Thanks for being there.

# Author's declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: Nicholas Martin DATE: 18/03/2021

# Table of Contents

# List of Tables

# List of Figures

**Figure** **Page**

# Chapter 1

# Introduction

Neural networks are a class of connectionist models that use distributed representations, which are notoriously difficult to interpret. Selectivity analysis offers one way to better understand the workings of neural networks at the level of individual units, allowing a unit's activation to be interpreted with respect to some aspect of the input. For example, class-selectivity can show the extent that units are tuned to inputs from one particular class.

There is debate about the extent to which a unit's selectivity relates to the functional role a unit plays in a model. The importance of highly selective units has been debated, they are seen by some as unimportant or even harmful for model performance (Morcos et al., 2018); others suggest that highly selective units are highly important but only for a subset of classes (Zhou et al., 2018b). Others have described highly selective units as 'detectors' for a particular feature or even as 'localist' units for a feature (Zhou et al., 2015; Bowers et al., 2014). These claims imply that a unit with high selectivity for a class *is* the model's representation for that class. This challenges the notions that units in neural networks may contribute to multiple representations, and that information may be represented across multiple units (Rumelhart et al., 1986a).

The thesis has two main aims; the first is to investigate how selectivity is measured and the relationship between selectivity and the functional role played by a unit. The second aim is to investigate dataset, model and task properties thought to influence levels of selectivity. In this chapter I will introduce the research problems, give them some context, explain the aims of the research and the significance of the findings.

## 1.1 Topic and aims

The main theme of this thesis is selectivity in neural networks. Neural networks have existed for 70 years, but it is in the last decade that they have emerged as a leading approach to tasks such as object recognition, speech recognition and image captioning. Neural networks are a type of machine learning algorithm that use layers of connected units to transform input data and extract task relevant features in order to produce a particular output. The input is represented at each hidden (e.g., internal) layer as a pattern of activation across all units in the layer, known as a distributed representation. Distributed representations are hard to interpret as units may contribute to representing many different items. Distributed representations can be contrasted with localist representations, where task relevant features are represented with their own dedicated unit.

One method for interpreting units in neural networks is selectivity analysis. This involves relating the activation of a given unit to the presence of some feature or property of the dataset. The implications of high selectivity have been debated in recent literature. One claim is that highly selective units are unimportant or even harmful to model performance (e.g., Morcos et al., 2018) and that methods for understanding models based on their interpretability may be misleading. Others have suggested that highly selective units are important, but only for a subset of items or classes (e.g., Zhou et al., 2018b). Others go further and claim that highly selective units are in fact localist units. That is, the model has learned to represent the presence of a task relevant feature through the activation of a dedicated unit rather than through a distributed representation (e.g., Bowers et al., 2014, 2016, etc.). Under this interpretation, highly selective units are vitally important for the recognition of the feature they are selective for.

The overarching theme of this thesis is the reliability and scope of selectivity analysis for understanding unit function. This will be addressed through a series of studies looking at how selectivity is measured; the impact of the model, task and dataset on the levels of selectivity; and on the ability of selectivity measures to predict the functional role played by a unit, as shown through lesioning. The objective is to provide context for other researchers interested in how neural networks solve tasks. The findings highlight the strengths and limitations of several popular selectivity measures. They also may provide indications for why selectivity is higher in one model than another model or for one class than other classes. Finally, the results demonstrate that there may be surprisingly low correspondence between a unit's selectivity and its functional role.

## 1.2 Key terms

The term selectivity relates to the differences in a unit's activation for one particular item, feature or class of items, in comparison to all other items, features or classes. 'Items' refers to the entities in a dataset. For example, if a dataset consists of 10 images of cats and 10 images of dogs, then there are 20 items. If the aim of the task was to classify the images as cats and dogs, then the dataset consists of two classes. Features refers to aspects that comprise the items such as whiskers, paws, collars, grass, carpet etc. A unit with high item-selectivity might be strongly activated by a single image of a cat. A unit with high class-selectivity for ‘cats’ would have high activation for all ten images of cats and low activation for all images of dogs. A unit with high feature-selectivity for ‘grass’ could have high activation for images of cats or dogs on grass, but no images of cats or dogs on carpet.

Another key term is sparsity, which is used in two ways when discussing neural networks. Population-sparsity refers to the proportion of units in a given layer than respond to a particular item. A sparse distributed representation is one with few active units (i.e., high population-sparsity). Lifetime-sparsity refers to the proportion of items that a given unit responds to. A sparse unit is one that fires to few items (i.e., high lifetime-sparsity). These two terms are related on average – the mean lifetime sparsity and mean population sparsity will be similar, but an individual unit can have high lifetime sparsity (e.g., fires to few items), yet be part of a representation with low population-sparsity (e.g., many active units). Lifetime-sparsity is equivalent to item-selectivity - they both refer to the proportion of items that activate a unit.

## 1.3 Relevant literature

The last five years have seen an increase in interest in the interpretability of neural networks. There is a large body of research on selectivity and related methods for interpreting units, such as activation-maximization and lesioning. For example, recent research has shown that class-selectivity is lower in early layers of models than in later layers, and that batch-normalization regularization reduces selectivity. However, there is little consensus on how selectivity should be measured or what constitutes high selectivity, as different groups prefer different selectivity measures. These measures are typically continuous and some groups report it as such, while others report the number of units with selectivity above some threshold (e.g., there were $n$ highly-selective units in the model). Both of these factors make it difficult to compare selectivity results between studies. Levels of selectivity vary between models so a unit considered to be highly selective in one study would be considered to have low selectivity in a different study. A unit presented as an example of high selectivity might be weakly tuned to one class more than any other class, or it might have a pattern of activation similar to that which would be expected from a localist unit.

There are gaps in the literature where findings from models trained on small tasks have not been tested on larger tasks. For example, it has been suggested that arbitrary tasks are best handled by increasing the selectivity of the representations compared to systematic tasks (e.g., Kumaran et al., 2016; Coltheart et al., 2001; Kumaran et al., 2016; Morcos et al., 2018). However, this literature relates to tasks where models learn the mapping between input-output pairs individually (e.g., associating the spelling of a word with its pronunciation or meaning) for a small number of items; rather than tasks where large numbers of items are categorized into classes.

Finally, there is disagreement on the implications of high selectivity regarding unit function. One position is that highly selective units are less important than units with low selectivity in terms of their contribution to a model's performance. As such, studies that focus on highly-selective units may be misleading. A second position is that highly selective units are very important, but only for a subset of classes. This view advocates that a unit's selectivity correlates with its importance of units for the subset of classes. A third position is that highly selective units acts as localist units. That is, the model has a dedicated unit for representing the feature in question, and the unit is vitally important for that feature, and not important for other features.

## 1.4 Research Question

This thesis focuses on methodology relating to interpreting units in neural networks, and on factors that may lead to increased selectivity. Chapters 3 and 6 are methodology oriented, investigating methods for measuring selectivity and the class-specific effects of lesioning respectively. These chapters compare evaluate different measures and highlight cases where there performance is unexpected. Chapters 4 and 5 investigate factors thought to be associated with increased selectivity: the arbitrariness of input-output mappings and the superposition-catastrophe. In focusing on these questions, I seek to address two common assumptions in the literature:

- That distributed representations are uninterpretable, therefore finding a highly selective unit is remarkable in itself.
- That high selectivity in a unit means that the unit must have a privileged status or importance for the model.

## 1.5 Methodology

Neural networks are a broad class of models which can vary along many dimensions. Model architecture, size, weight-initialization, dataset, task and many other parameters may all affect the selectivity of the units. As such, I have generally trained large numbers of models with different parameters on each task in order to identify principles that apply to neural networks in general. A notable exception is chapter 6 where I study a single model in depth. I do not experimentally manipulate this model and the levels of selectivity are not the main focus, rather this chapter investigates ways of measuring the class-specific effects of lesioning. Models in this thesis may be considered small or medium-sized; far smaller than the state-of-the-art models trained on large datasets that are the norm for much of the computer-science literature included here. Hopefully this makes my findings more applicable to researchers outside computer-science, that have an interest in representation and learning (e.g., in psychology or neuroscience), that may be reliant on small and medium sized models.

Three of the chapters are based around attempts to replicate a finding from the recent literature. Chapter 4 replicates simulation 2 from Vankov and Bowers (2017); chapter 5 replicates Bowers et al. (2014) and chapter 6 analysis 1 replicates Zhou et al. (2018b). These replications were intended as starting points before extending the research in new directions. These are not identical replications, however, in all three cases I get very different results to the original studies. This failure to reproduce the findings of other researchers meant some of the experimental aims were dropped and the thesis has more of a focus on methodology (e.g., how selectivity and the effects of lesioning are measured) and on the implications of selectivity. The methodology for all experiments involves training multiple models on a particular task. Once a model is trained to a sufficiently high accuracy, the activation levels of all units in the model are recorded in response to all correct items in the dataset. I only analyse correct responses as I am interested in the representations that relate to how a model solves a task, not representations that support inaccurate responses. I analyse each unit's selectivity for each class (e.g., or item or feature if relevant) and report a unit as being most selective for whichever class has the highest selectivity score. Example of scripts to train models, record hidden activations, analyse the selectivity and perform lesioning are available at `https://github.com/Nickdotmartin`.

## 1.6 Contributions

In chapter 3 I highlight the fact that two commonly used selectivity measures (class-conditional mean activation and precision) can overestimate class-selectivity in sparse units of multi-layer perceptrons (MLPs). This served as pilot data for a comparison of selectivity measures in large convolutional neural networks (CNNs), where the limitations of these measures were discussed (Gale et al., 2019). Since its publications this paper was cited in Leavitt and Morcos (2020b), where they refer to the limitations of the CCMA and precision measures which were raised in Gale et al. (2019). Chapter 3 also identifies maximum-informedness (max-info) as a more reliable measure of selectivity in MLPs, less prone to inflating selectivity scores in sparse units than other measures. This was also found to be true of max-info when it was applied to large scale CNNs in Gale et al. (2020). This suggests that the findings relating to these selectivity measures are robust and generalizable. Nonetheless, all the selectivity measures I compare (including max-info) have floor or ceiling effects. This highlights the need for a measure of selectivity that can reflect differences in units with the whole range of possible levels of selectivity.

In chapter 4 I show that there is a relationship between class-selectivity and systematicity. Selectivity is higher where data has systematic rather than arbitrary mappings. This finding had not been experimentally demonstrated before. This finding is surprising in the light of data showing that item-selectivity is lower for datasets with systematic rather than arbitrary mappings, and of literature suggesting higher selectivity for arbitrary tasks. Further, I show that within one dataset, selectivity will be higher for more systematic classes than for arbitrary classes. Throughout the thesis I also highlight a strong relationship between selectivity and task difficulty: selectivity is consistently higher for easier tasks (e.g., with shorter training times/higher accuracy) than more difficult tasks.

One compelling case for the usefulness of highly selective representations comes from Bowers et al. (2014), where highly selective units emerged in conditions where the model was most likely to suffer the superposition-catastrophe. They suggested that the highly selectivity units emerged so the models could avoid the superposition-catastrophe. I have demonstrated that models can succeed in such tasks with high superposition-pressure, without utilizing highly selective units.

Finally, Zhou et al. (2018b) measured the class specific effects of lesioning and found that units were specialised for a subset of classes. The claimed that selectivity was a good predictor of this specialisation. I apply their methodology for measuring the effects of lesioning on each class and find that it is poorly suited to identifying specialised units. I adapt the method for measuring the effects of lesioning to show that there are some units with apparent specialisation, but that the selectivity for these units is below the model average. Further, the class that these units are most selective for is typically not the class that the units are specialised for. This challenges the claim that selectivity is a good predictor of the functional role of a unit.

## 1.7 Overview

Chapter 2 reviews the literature on selectivity in neural networks. In chapter 3 I train models on three datasets and compare the selectivity results from four selectivity measures. I discuss the principles that the different measures operate on; the differences between selectivity score with respect to a particular units and highlight edge cases where they may give unintuitive results. The work I carried out on investigating and comparing various selectivity measures in chapter 3 was developed and included in work for which I was a co-author, published for a conference (Gale et al., 2019) and a journal (Gale et al., 2020).

In chapter 4 I investigate the effect of data with arbitrary mappings on selectivity. I find that selectivity is higher for datasets and classes where mappings are systematic rather than arbitrary.

In chapter 5 I attempt to replicate a study from Bowers et al. (2014) showing that recurrent models learn localist representations in order to overcome the superposition catastrophe. I find no localist units in my models and no difference in selectivity between conditions where there is high chance of the superposition-catastrophe (e.g., long sequences) and conditions with no chance of the superposition-catastrophe (e.g., single items).

Chapter 6 investigates the relationship between selectivity and the functional role played by units. I apply a measure of the effects of lesioning proposed by Zhou et al. (2018b) to a convolutional neural network. I investigate Zhou's claims that units are specialised for a particular class and that this specialisation is predicted by selectivity. I find that the importance of a unit to a class is weakly associated with the selectivity for that class. However, the class that a unit is most 'important' for is typically not the most selective class. This highlights the low correspondence between selectivity and unit function.

## 1.8 Publications

- Gale, E., Blything, R., Martin, N., Bowers, J. S., & Nguyen, A. (2019). Selectivity metrics provide misleading estimates of the selectivity of single units in neural networks. In CogSci (pp. 1808-1814).
- Gale, E. M., Martin, N., Blything, R., Nguyen, A., & Bowers, J. S. (2020). Are there any 'object detectors' in the hidden layers of CNNs trained to identify objects or scenes?. Vision Research, 176, 60-71.

Chapter 3 is based on work that I did in 2018 and 2019 into comparing selectivity measures; highlighting their limitations and proposing new measures of selectivity. This work was subsequently developed by myself, Ella Gale and Jeffrey Bowers into a project comparing various selectivity measures on a large CNN. This was combined with research by Ella Gale, Jeffrey Bowers, Ryan Blything and Ahn Nguyen on the human interpretation of activation-maximization images for submission to a conference and journal (Gale et al., 2019, 2020). The CredIt author statement for the 2020 paper lists me as contributing in the form of validation, writing, review and editing and visualization. My contributions only apply to the work comparing selectivity measures, I was not involved in the project on human interpretation of activation maximization images. The preliminary work I did that informed these papers involved comparing various selectivity measures in small models. This research highlighted the fact that scores for CCMA and precision may be higher than for other measures. This research included generating visualizations highlighting edge cases where CCMA scores are surprisingly high. I also researched alternative selectivity measures that would not inflate scores in sparse units, as as a result I proposed Max-info, which was included in Gale et al. (2020). I did not conduct the analysis of various selectivity measures on models trained on imageNet in these papers: this was conducted by Ella Gale. I did not produce the visualization comparing measures with hypothetical data (e.g., figure 5 in the 2019 paper, or figure A2 in the 2020 paper), this version was produced by Ella Gale, based on my visualizations. I was involved in writing, reviewing, editing and validating the sections of the paper on comparing the selectivity measures, along with Ella Gale and Jeffrey Bowers.

# Chapter 2

# Background

## 2.1 Neural Networks and Deep Learning

Neural networks have existed in some form for over 70 years and have been known as 'multi-layer perceptrons' (MLPs), 'neural networks', 'parallel distributed processing', 'connectionism' and 'deep learning'. It is a statistical technique that can be used to classify patterns using multiple layers of interconnected processing units. The term 'deep learning' highlights the fact that the number of layers of units has increased in recent years. They are a leading technology for image recognition and language processing, and deep learning has many other applications which are beyond the scope of this review (e.g., being used as part of a generative model to discover latent representations, Bau et al., 2018; Higgins et al., 2017).

Although neural networks have a rich history (for an overview, see Schmidhuber, 2015), this brief introduction to deep learning models will begin in 2012, when neural networks beat all competitors in annual competitions for both object recognition and speech recognition (Krizhevsky et al., 2012; Hinton et al., 2012). These models won their respective competitions by large margins. Since then neural network models from various groups have gone on to become the dominant approach in computer vision and language processing.

The model that won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 was named AlexNet after one of its designers (Krizhevsky et al., 2012). The ILSVRC allows participants to train a model on over a million images from a thousand object categories, with models performance scored on a test set of 50000 items. Models make 5 predictions for the class of the object in each image. AlexNet predicted the correct category in its 5 guesses 84.7% of the time, with its nearest competitor scoring 73.9%. AlexNet is a convolutional neural network (CNN) consisting of 5 layers of convolutions and max-pooling and two fully-connected layer before the output layer. Convolutional, max-pooling and fully-connected layers had been around for a long time (e.g., LeCun et al., 1998). However, AlexNet was different to other CNNs around at the time: it used relu activation function which train more quickly and avoid the saturation problems associated with sigmoid or tanh units; it used data augmentation during training to increase invariance to transformations (e.g., presenting an image multiple times with different transformations such as rotated or mirror symmetric version etc.); and it used novel regularization techniques such as local contrast normalization and dropout to avoid over-fitting the training data. Krizhevsky et al. (2012) were able to train their models more quickly than had previously been possible by training their model across two GPUs in parallel. This was made possible by the development of general-purpose GPUs running Nvidia's CUDA library, released in 2008 (NVIDIA et al., 2020). CNNs have consistently won ILSVRC since 2012, with many well known models such as GoogLeNet (which introduced 'inception' modules), VGG and ResNet (which introduced 'skip-connections'), featuring as previous winners or runners up (Szegedy et al., 2015; Simonyan and Zisserman, 2015; He et al., 2016).

Language based tasks (e.g., speech recognition) require processing sequential data, which is typically modelled with recurrent neural networks (RNNs). A collaboration between Microsoft, Google, IBM, and the University of Toronto (led by Geoff Hinton) led to the development of a deep recurrent NN that won a speech recognition contest in 2012 (Hinton et al., 2012). However, the following year, another model from Hinton set a new record on a speech recognition competition, this was a deep recurrent model with Long Short-term Memory units (LSTMs, Graves et al., 2013). LSTMs are powerful units that avoid the 'vanishing-gradients problem' that had previously hindered attempts to train sequence learning models on large datasets. They were developed in 1997 (Hochreiter and Schmidhuber, 1997), and models using LSTMs had previously won AI competitions in 2006 (Graves et al., 2006). But from 2013, LSTMs became a key component of sequence modelling NNs, and speech recognition and language translation could be solved reliably enough to become standard software on mobile phones.

### 2.1.1 Selectivity, Distributed Representations and Localist Coding

Selectivity refers to the tuning of a unit in a neural network (or a neuron in the brain). If a unit is *selective for $x$*, then one expects that $x$ is a reliable source of activation for the unit. For example, a unit with highly selectivity for $x$ would fire for all $x$s but never for $y$s or $z$s etc. A more relaxed interpretation of 'selective for $x$' would allow some low-level activation for $y$s and $z$s, as long as $x$s are associated with higher levels of activation (e.g., Bowers, 2011; Coltheart, 2017; Thomas and French, 2017).

Issues of selectivity relate to an early debate amongst neuroscientists about whether individual neurons were the basic unit of computation in the brain (e.g., Barlow, 1972; Konorski, 1967) or whether information was coded by populations or ensembles of units (e.g., Sherrington, 1940; Hebb, 1949). These positions came to be known as 'localist' representations (i.e., high level concepts can be represented by a dedicated by a neuron) and ‘distributed representations’ (e.g., populations of neurons represent multiple concepts).

Researchers studying cognition build models which operate in line with localist or distributed coding principles. Models may also include separate localist layers and distributed layers (e.g., Doumas and Martin, 2018; Martin and Doumas, 2019). The difference between these approaches is given in these definitions taken from the Parallel Distributed Processing books (Rumelhart et al., 1986a,b) which introduced the basic principles of the distributed coding approach, from a chapter on distributed representations (Hinton et al., 1986). Local representations are defined as follows:

> Given a network of simple computing elements and some entities to be represented, the most straightforward scheme is to use one computing element for each entity. Rumelhart et al. (pg.77, 1986a)

Distributed representations are defined as:

> Each entity is represented by a pattern of activity distributed over many computing elements, and each computing element is involved in representing many different entities. Rumelhart et al. (pg.77, 1986a)

These definitions do not describe *a* local or distributed representation, but refer to a complete set of representations, e.g., ‘computing elements’ refers to all units in a layer of a model, and ‘entities’ refers to all items in a dataset. Although these definitions refer to entities (e.g., items in the dataset), selectivity can also be discussed at the level of features in the dataset (e.g., Zhou et al., 2017); or classes (e.g., an output class in a classification task; Morcos et al., 2018).

Note: models that learn distributed representations in the hidden layers are the focus of this thesis. I will refer to these as *neural network* or just *models*. I shall use ‘localist model’ when referring to models where one or more hidden layers is designed to use localist coding throughout. With these definitions, convolutional neural network, recurrent neural networks and multi-layer perceptrons are all types of *neural network* (e.g., Zhou et al., 2015; Bowers et al., 2014; Vankov and Bowers, 2017); whereas the interactive-activation model and other psychological models with a dedicated localist layer are *localist models* (e.g., McClelland and Rumelhart, 1981; Dell, 1986; Grainger and Jacobs, 1996; Hummel and Holyoak, 1997; Davis, 2010; Martin and Doumas, 2019).

The Parallel Distributed Processing books laid out the principles behind the use of distributed representations as an alternative to localist models (Rumelhart et al., 1986a,b). The basic premise was that rather than using dedicated units for each task-relevant feature, cognition could be modelled with distributed representations, where units could contribute to the representation of multiple features, and features could be represented by the co-activation of multiple units. Unlike localist models, that are designed around specific sub-tasks, with units performing specific functions; distributed representations favour bottom-up, data-driven learning guided by the backpropagation algorithm, with the assumption that if the model can solve the task, the representations it has learned must be appropriate for the task (Rumelhart et al., 1986a,b; Plaut and McClelland, 2000). These models can be trained in a variety of ways, but for the present I will focus on a method called supervised learning, which is typically used for classification tasks. A model is initialized with a particular depth (e.g., number of layers) and width (e.g., number of hidden units per-layer). The modeller does not specify which features the model must learn prior to building the model. Units have no assigned function nor is the level of analysis for each layer specified. Units are connected to other units in adjacent layers (or sometimes to more distant layers, e.g., He et al., 2016), and prior to training the connection weights (e.g., strength of connections) and biases are initialized with small random values. In training, the model is presented with an item from the training set. The model then predicts which class the item belongs to by having a pattern of activation across the output units where the most active unit is the unit for the predicted class. Where the prediction is incorrect (as it often is at the start of training), the difference between the ideal levels of activation for the output units, and the predicted output pattern are used to compute an error signal. The backpropagation algorithm allows this error signal to be propagated back through the model, and specifies where the strength of the connections need be adjusted to minimise the error. The process of learning a task involves iteratively making the small adjustments to the connections between units after each prediction (or batch of predictions) to minimise error on the training set.

Lateral-inhibition is often used in localist models to maximise the signal-to-noise ratio such that only the most relevant units for an item have high activation (e.g., McClelland and Rumelhart, 1981; Rumelhart and McClelland, 1982; Davis, 2001; Doumas and Martin, 2018). This results in high population-sparsity for representations (e.g., few active units for each item), and high selectivity for units (e.g., high activation for a few items). Neural networks typically do not use lateral-inhibition in the hidden layers. As such, features (e.g., or items or classes of items) are typically represented through the co-activation of different combinations of units, rather than by the activation of dedicated units, as would be the case for in a localist model. The activation of a given unit typically does not typically correspond to a specific feature, but rather, units actively contribute to the representations of multiple features.

Distributed representations are good for capturing the statistical regularities of a domain (e.g., capturing the linguistic context of words in a corpus, Goldberg and Levy, 2014) or system of mapping (e.g., from orthography to phonology, Coltheart et al., 2001). A defining feature of a distributed representations is that the the similarity between two representations in a distributed layer reflects some similarity between the concepts themselves, such have the number of common attributes shared by both concepts (Rumelhart et al., 1986a; Goodfellow et al., 2016).

The success of models using distributed representations demonstrates that for many tasks it is not necessary for units to be dedicated to representing a particular item, feature or class. For example, Hinton et al. (1986) trained a model to map the spelling of 20 words to their meanings. They lesioned hidden-layer units individually (e.g., set their weights and biases to zero so they did not contribute to solving the task) and found that removing each unit led the model to fail on multiple items. This was taken to show that the model did not use a localist coding (e.g., a dedicated hidden unit for each word). However, there are domains where models using only distributed representations perform poorly compared to models which include localist layers (e.g., relational reasoning; Doumas and Martin, 2018; Ricci et al., 2018; Martin and Doumas, 2019).

### 2.1.2 The Interpretability of Distributed Representations

Distributed representations have been described as being uninterpretable (e.g., Thorpe, 1995; Rogers and McClelland, 2008; Vankov and Bowers, 2017). For example:

> with distributed coding individual units cannot be interpreted without knowing the state of other units in the network. (p.g. 550, Thorpe, 1995)

or

> the learned internal representations have no directly interpretable content (pg. 693, Rogers and McClelland, 2008)

and finally

> each piece of meaningful information is coded by the simultaneous activation of multiple processing units and at the same time, each unit is involved in representing many different things. As a result, it is impossible to identify what a single-unit codes for. Distributed representations are thus different from both localist representations in psychology and grandmother cells in neuroscience, where the activation of units is interpretable. (pg. 392, Vankov and Bowers, 2017).

I argue that although units in distributed representations are not as interpretable as units in localist models; it is an overstatement to therefore claim that they are uninterpretable. A unit in a distributed representation is interpretable with respect to their inputs, insofar as they are selective: highly selective units are more interpretable than units with low selectivity. However, units in distributed representations are far less interpretable with respect to the model's output than units in localist models.

The definitions for localist and distributed representation given above (in section 2.1.1) clearly capture the main difference in approach regarding the role of units (Rumelhart et al., 1986a,b). These definitions emphasize the differences between localist and distributed approaches, and in doing so, they overlook the variability in selectivity and sparsity that is possible with distributed representations. To be specific, it is not necessary for *every* entity to be represented by the co-activation of multiple units. Representations can vary in sparsity such that some entities might be represented by the activation of a single unit. Similarly, it is not necessary for *every* unit to contribute to representing multiple entities, some units may contribute to the representation of a single entity. These definitions discuss the representation of items (e.g., entities), however, the same principles apply for features or classes. An example of the variability in selectivity and sparsity that is possible with distributed representations is shown in table 2.1, reproduced from Földiák (2009). It shows a set of hypothetical distributed representations for 10 stimuli, across 10 units. The proportion of units which are involved in representing each stimulus is the *population-sparsity* (or just sparsity), and is shown down the right side of the table. The proportion of stimuli that each unit is involved in representing is the *selectivity* and is shown along the bottom of the table. Although these representations are not from real data, they demonstrate that selectivity and sparsity may vary, and in principle, representations can be maximally sparse or selective (e.g., values of .1 in this example).

| | u1 | u2 | u3 | u4 | u5 | u6 | u7 | u8 | u9 | u10 | Sparsity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| stim1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | .2 |
| stim2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | .2 |
| stim3 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | .2 |
| stim4 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | .1 v.sparse |
| stim5 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | .4 |
| stim6 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | .2 |
| stim7 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | .2 sparse |
| stim8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | .2 |
| stim9 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | .5 dense |
| stim10 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | .2 |
| Selectivity | .3 | .1 | .3 | .1 | .4 | .2 | .1 | .2 | .5 | .2 | .24 |
| | | v.narrow | | | | narrow | | | broad | | |

Table 2.1: **A set of binary representations of ten stimuli (rows) across ten units (columns) displaying varying levels of selectivity and sparsity**. Sparsity (far right column) is given as the proportion of units that are active for a given stimulus. Highlighted rows are stim4 with a very high sparsity of .1, stim7 is a sparse distributed representation with a value of .2 and stim9 is a dense representation with a sparsity of .5. Selectivity (bottom row) is the proportion of stimuli that a given unit responds to. Highlighted columns shows unit2 with selectivity of .1 (e.g., very narrow tuning); unit6 has narrow tuning and responds to two stimuli; unit9 has broad tuning and a selectivity of .5. The bottom right box shows that the average sparsity is equal to the average selectivity with a value of .24. From Földiák (2009) and also reproduced in Bowers (2011).

The extent to which units are interpretable with respect to the inputs, relates to their selectivity. For example, unit 2 is interpretable, in that it only fires for stimulus 7 (stim7). As such, one can infer whether stim7 is being presented to the model by observing unit 2. Similarly, if unit 6 is active, I can infer that either stim7 or 9 is being presented. Therefore, unit 6 is somewhat interpretable, but less interpretable that unit 2. Unit 9, fires for five of the ten stimuli, giving it low interpretability. It is not uninterpretable, if it fires one can infer that either stimulus 2, 5, 6, 8 or 10 were presented, but not stimuli 1, 3, 4, 7 or 9. This demonstrates that interpretability with respect to the input is a property of units in distributed representations which directly relates to their selectivity.

However, interpretability with respect to the model's output is less clear - that is, understanding the causal relationship between a unit's activation and a particular response from the model. If stim7 was represented by the activation of unit 2 alone, it might be reasonable to claim that the model's recognition of stim7 is triggered when unit 2 is active. However, stim7 is represented by the co-activation of units 2 and 6, so it is not clear how the model would respond to the activation of unit 2 alone. The table suggests that recognition of stim7 follows from the activation of a distributed representation, consisting of units 2 and 6. However, even this can not been known from the activations of units 2 and 6. Neural networks typically do not use lateral-inhibition, which ensure that only the few most relevant units are active, and as such, not all activation in the model is *important*, or functionally relevant to the model. The extent to which a unit contributed to representation depends not only on the incoming signal or its activation, but also on the strengths of the weights *out* of the unit. For example, unit 6 might only be weakly connected to the output unit for stim7. In this case, unit 6 might not actually contribute to the model's response for stim7, which could be triggered by unit 2 alone. There is nothing in the activation of units 2 or 6 which can reliably indicate their importance to representing stim7. The sparsity of the representations can give an indication of the importance of a unit to a particular classification. For example, there are five units which fire for stim9, and in the absence of weights analysis or lesioning, it is not clear how much each of these units contributes to the model's recognition of stim9. However, there is only one unit which fires for stim4, so one can be confident in stating that unit5 is important for the identification of stim4. But, unit 5 is not a dedicated detector for stim4, as unit 5 is also involved in representing stimuli 1, 5 and 9.

The point is that distributed representations can take a variety of selectivity and sparsity values. In principle, this can include units with maximally high sparsity and selectivity which would be as interpretable as units in localist models. However, in reality, units often respond to multiple items (e.g., are not maximally selective). Their interpretability with respect to the input relates to their selectivity, and they are typically less interpretable than units in localist models. Items are typically represented by the co-activation of multiple units (e.g., not maximally sparse), rather than due to the firing of a 'detector'. Some active units will contribute to a response more than others, due to differences in connection weights between hidden units and the following layers. However, nothing in the activation gives clues to the outgoing weights. As such, attempts to interpret a unit's activation with respect to the output may be unreliable. Whilst this may sound bleak, unreliable interpretation may be more promising for understanding neural networks than the position that activations are uninterpretable! One of the main aims of studying representations is to understand how they relate to a model's behaviour (e.g., a particular output).

The following section will discuss evidence of a weak association between selectivity and a unit's contribution to a particular output. Research on the interpretability of neural networks aims to bridge the gap in our understanding of the association between activations and function. There is a large body of research using selectivity, sparsity, lesioning, weight analysis and other methods to interpret units in neural networks (e.g., Fong and Vedaldi, 2017; Murdoch et al., 2019; Adadi and Berrada, 2018; Dosilovic et al., 2018; Gilpin et al., 2019).

## 2.2 Selectivity in Neural Networks

The following section will review research on selectivity in neural networks. The first part will discuss ways of measuring selectivity, along with other methods methods for interpreting units (e.g., visualisations and lesioning) which can be used in conjunction with selectivity analysis to better understand the role of individual units. Then the subsequent parts will highlight examples of selective units, discuss whether these units are meaningful, factors that affect selectivity and the relationship between selectivity and unit function. The final sections will discuss limitations of selectivity analysis, the merits of treating selectivity as a discrete variable, and will introduce open questions which this thesis will address.

In the discussion of sparsity and selectivity relating to figure 2.1, I defined selectivity as the proportion of items that a unit responds to, which is only strictly correct when considering item-selectivity (e.g., the most selective unit would fire for a single item ). However, I am predominantly interested in class-selectivity, where a highly selective unit would fire exclusively for items from one of the output-classes. For example, in a model trained to classify images of cats and dogs, a unit which fires to all cats but no dogs would have low item-selectivity (e.g., it fires for many images of cats), but would have high class-selectivity as it only fired to for items from the 'cats' class.

### 2.2.1 Methods for interpreting units

#### 2.2.1.1 Qualitative methods: Visualizations

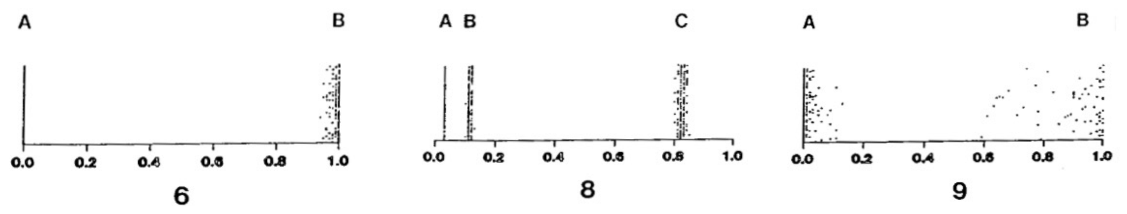
There are several qualitative methods that have been developed for analysing and visualizing the receptive fields of neurons (Zeiler and Fergus, 2014; Yosinski et al., 2015; Mahendran and Vedaldi, 2016, 2015). These methods fit with a long tradition of researchers inferring a unit's tuning properties by identifying stimuli that cause strong activation (Hubel and Wiesel, 1959, 1962; Lettvin et al., 1959). One simple approach is to visualize the images from the training or test set which most strongly activated a unit (e.g., Szegedy et al., 2013). Where there is a clear theme in these images (i.e., all images that highly activate a unit contain 'white flowers'), it suggests that this unit is tuned to that feature. However, there is no guarantee that the images can be easily grouped around a theme. This method is also limited in that an images may contain many features (e.g. white flower, green leaves, grey concrete, blue sky), and it is not always clear which features are responsible for driving the unit. Methods have been developed to resolve this issue by identifying the regions of the input responsible for driving the unit; such as segmented images (e.g., Zhou et al., 2015) and saliency maps (Itti et al., 1998).

A different approach to visualizing units uses synthetic images which are generated through an 'activation maximization' process to create an image that will maximally activate a given unit (Mahendran and Vedaldi, 2016). The method is typically used to produce multiple images from different initializations, to see if a clear theme emerges. Images generated in this manner might include a regularization term or some prior to ensure that they appear naturalistic. However unrealistic images can still be informative. For example, an image of a multi-headed dog-like creature would suggest that the unit is tuned to dog's heads, rather than to dog's paws or dogs in general. Producing these preferred inputs for units in different layers has revealed similarities between convolutional neural networks and the ventral pathway, with units in earlier layers being maximally activated by oriented bars, and by more complex shapes in later layers (Yosinski et al., 2015).

The team from OpenAI have used the most active dataset images and activation maximization images, along with analysis of the weights to understand the tuning of units in a convolutional neural network (Olah et al., 2020; Open AI, 2020). Through this work they identified novel low-level features which were present in the early layers of models. These included units which fire in response to curved lines, or fire to adjacent patches of high and low contrast. They also show how pairs of units in one layer, such as units which fire for dog's heads facing left and right, link to a unit in the following layer which fires to dog's heads from any point of view. This shows how viewpoint-invariance can emerge in a model.

However, Gale et al. (2019) have shown that there is not necessarily a correspondence between the interpretability of activation-maximization images and the selectivity of the units. They found that units which were shown to be highly interpretable for a given feature through visualizations, were found not to have high selectivity for that feature.

A different approach to unit visualization is to use scatter plots to show the activation of each item at a particular unit. This approach was used by Berkeley et al. (1995), who found remarkable structure in that units in models trained on logic questions (see figure 2.1). For example, in panel B of figure 2.1, all items relating to 'OR' had no activation, items relating to 'IF...THEN' had low activation and all items relating to 'NOT BOTH...AND' had high activation. Such neat structure is unusual, and may be due to the 'value units', a type of activation function used by his models which is no longer popular, although similar 'banding patterns' have been identified in models with sigmoid activation functions (Berkeley and Gunay, 2004; Bowers et al., 2014, 2016). Scatter-plot visualization can still be useful, even if they do not show such clear structure. For example Gale et al. (2020) used visualizations to highlight the activations for all



a) b) c)

Figure 2.1: **Jittered-density plots** showing banding patterns in units 4, 8 and 9 from an MLP with ten units trained to solve logic problems. The labelled bands correspond to different features of the data. In all units, items in band A do not share a common feature. a) Unit 6: items in band B all have the connective OR. b) Unit 8: item in band B have the connective IF...THEN, item in band C have the connective NOT BOTH...AND. c) Unit 9: This unit clearly shows two bands of activation, however neither band is associated with a common feature. Image from from Berkeley et al., 1995.

'bus' items in a unit identified by Zhou et al. (2017) as a 'bus-detector' using Network Dissection (a selectivity measure). The scatter-plot in Gale et al. (2020) showed that although few busses had zero activation, none of the most strongly activated images contained busses. Gale et al. (2020) used this scatter-plot to challenge claims that the unit was a 'bus-detector'.

#### 2.2.1.2 Quantitative methods: Selectivity

The qualitative methods described above can give a good intuition about the features that drive a unit. However, quantitative methods are required for a more detailed analysis. A variety of selectivity measures have been used in the literature. Here I discuss three popular approaches, that can be summarised as precision-based, means-based and distance-based. They all use the level of activation for items in a dataset to describe a unit's tuning curve.

**Precision-based measures** involve analysing only a subset of activations: those that are most active, and giving the proportion of these that are from a given class. For example, in Zhou et al. (2015), they showed human-raters the 60 segmented images that most strongly activated a given unit. They were then asked to identify a common feature across those images and the proportion of the 60 images that contained the feature. Units were considered to be 'detectors' for a feature if 75% of images contained the feature (e.g., precision > .75; I shall return to their use of the term 'detector' in section 2.3.4.3). Using this method they found units with 84% precision for 'table lamps' or 60% precision for 'tables'. In using human-labellers, they were able to look for any common concept in the images, rather than just the class labels. I describe these methods as precision-based as they use Precision: an information-retrieval measure which is defined as the proportion of retrieved documents that are relevant (Van Rijsbergen, 1979). In later research, Zhou et al. (2017) proposed automating the process in a method they call Network-dissection. Network-dissection is precision-based, in that only the subset of items that most strongly drive the unit are included in the analysis.

One limitation of precision based methods is that they focus only on the items that most strongly drive a unit, and therefore they overlook details of how a unit represents items with low activation. For example, in the unit from Zhou et al. (2015) with 84% precision for 'tables', 50 of the 60 images with highest activation contained a table. This certainly suggests some tuning to 'tables'. There were 200000 items in this dataset, although I do not know how many of these images included tables. If there were only 50 tables, then this unit's is indeed highly selective for tables. However, if there are 100 or 1000 tables in the dataset, then the unit's selectivity for tables *in general* is not clear with a precision measure that only considers the 60 most active items. If there are many tables with an activation of zero, then the unit might be better described as being selective for a subset of tables. Precision measures using a small threshold (e.g., 60 items) may be unsuitable for measuring selectivity in large datasets (for further examples, see Gale et al., 2019, 2020). The threshold of 60 images seems small for a test set of 200000 images, and presumably relates to the attention-span of the humans categorising the images. If there were 2000 images in the test set containing tables, then the precision score for the most active 2000 items might give a better reflection of the units performance as a 'table-detector'. However, even with a larger threshold, precision-based methods have the assumption that only the most active items are required to interpret a unit's tuning. For example, precision based methods tells us nothing about the difference between items with low activation and items that do not activate the unit at all.

**Distance-based measures:** Bowers et al. (2014) proposed a selectivity measure that gives the minimal distance between activations of items from one class, and the activation of all other items. I will refer to this measure as Bowers' selectivity or B-sel. If *all* items from class-$x$ have higher activation than any other class, then B-sel will give the distance between the least active item from class-$x$ and the most active item from another class. Conversely, if all items from class-$y$ have lower activation than all other classes, then B-sel will give the distance between the most active item from class-$y$ and the least active item from other classes. Bowers et al. (2014) proposed that units where the B-sel is greater than .5 can be labelled as 'localist' units (I shall discuss their use of the term 'localist' in section 2.3.4.3). Since Bowers typically applied this measure to sigmoid units, with activations in the range 0 to 1, a B-sel greater than .5 indicates that there is more variance between classes than within classes. A limitation of B-sel is that it only scores classes where all items from that class have higher activation than all other items (or all items have lower activation). If all class-$x$ items have medium activation, and there are items from other classes with higher and lower activation, then B-sel will score 0 for this class. Similarly, B-sel would give a score of 0 if class-$x$ contains some items with high activation and others with low activation, and there are items from other classes in between. As a consequence, B-sel typically gives a score of 0 to the majority of units (e.g., Bowers et al., 2014, 2016; Vankov and Bowers, 2017).

**Means-based measures** don't use the individual item activations, but instead they use the mean activation for each class. One popular approach, class-condition-mean-activation selectivity (CCMA) gives each class mean relative to the mean of all other classes (Morcos et al., 2018). This measure is based on selectivity measures used in neuroscience (e.g., Freedman and Assad, 2006; Tsao et al., 2006). In principle, the use of mean activations should avoid the situation where a high selectivity score relates only to a subset of items, as could be the case with precision, which considers only a subset (e.g., 60 items). However, in reality, it has been shown that CCMA can over-estimate class-selectivity, especially for sparse units. For example, one can conceive of a unit where all items except one have zero activation. CCMA will score 1.0 to the class with the active item (this point was made with hypothetical data in Gale et al., 2019, 2020). This is because the class with the active item will have a mean activation greater than zero, while the mean for all other classes is zero. As a result, the class mean relative to other classes is 1.0. These inflated scores can occur in any unit with many zero activations, which are common, especially in relu units, which tend to have much sparser activation than sigmoids (Glorot et al., 2011).

#### 2.2.1.3 Functional methods: Lesioning

Selectivity analysis can identify an association between a unit's activation and the presence of some dataset property or feature. However, it does not show the causal role between the unit activation and the model's output. In theory, the importance of a unit for a particular classification can be understood through analysis of the weights connecting hidden and output units. In practice, neural networks are often too complex for such a process to be undertaken. One way to infer the importance of a unit is to remove it, known as ablation or lesioning, typically by fixing the unit's weights to zero (e.g., Bowers et al., 2016; Morcos et al., 2018; Zhou et al., 2018b). The role of a unit can be inferred by noting differences in the performance of the lesioned and unlesioned models. Morcos et al. (2018) demonstrated that highly selective units are no more important in terms of overall accuracy than units with low importance. That is, the drop in overall accuracy is similar after lesioning a highly selective unit or a unit with low selectivity. However, Zhou et al. (2018b) demonstrated that highly selective units lead to selective impairments (e.g., they are important for a subset of classes) whereas units with low selectivity will cause a small impairment to multiple classes. Zhou et al. (2018b) also found significant correlations between the size of the drop in accuracy for the class with the largest drop, and the selectivity of the unit, using a variety of selectivity measures. They took this to suggest that that selectivity is a good predictor of the effects of lesioning. In other words, Zhou et al. (2018b) claimed that a unit's activation is somewhat interpretable with respect to the model's output.

## 2.3 Selective Units

Table 2.1 provided hypothetical examples of distributed representations, which I argued were interpretable with respect to the input to the extent that they were selective. The association between activation and the presence of certain stimuli allowed inferences to be made about which stimuli were being presented to the model. Researchers have found many similarly interpretable units in trained neural networks (e.g., Berkeley et al., 1995; Le et al., 2013; Bowers et al., 2014; Zhou et al., 2015, 2017; Karpathy et al., 2016; Lakretz et al., 2019; Radford et al., 2017, etc. ). Highly selective units have been found in a variety of different architectures including MLPs (e.g., Berkeley et al., 1995; Vankov and Bowers, 2017); CNNs (e.g., Rafegas et al., 2017; Zhou et al., 2017); and RNNs (e.g., Bowers et al., 2014; Karpathy et al., 2016; Radford et al., 2017). Highly selective units emerge in models training using different methods, such as supervised learning (e.g., Zhou et al., 2015; Lakretz et al., 2019; Karpathy et al., 2016); unsupervised learning (e.g., Le et al., 2013; Netzer et al., 2011); and in generative-adversarial training (e.g., Bau et al., 2018). In other words, far from being uninterpretable, interpretable units have been found in distributed representations wherever people look. One example of an interpretable unit comes from Le et al. (2013). They trained a locally-connected sparse auto-encoder on an unlabelled set of images of faces and objects. They measured the selectivity of units by computing 20 equally spaced thresholds between the maximum and minimum activation values for each unit and reported the ability to infer whether a face or object was being presented to the model at the best of the 20 thresholds. They could predict whether a face was present with 81.7% accuracy. That is, faces had activation above the threshold, and non-faces had activation below the threshold for 81.7% of the 37000 images in the test set. This was a striking finding that demonstrated that an interpretable unit for a high-level complex object could emerge in a neural network.

### 2.3.1 Are selective units meaningful?

Having established that interpretable units can emerge, it is reasonable to ask whether they are meaningful in any way, or are just random artifacts. Units in early layers are typically selective for low-level features, and units in later layers are often selective for high-level features (Zhou et al., 2015, 2017). Low-level features include oriented lines, colours, materials and textures (e.g., wood, square grid), regions and surfaces (e.g., road, grass), while high-level features include object parts (e.g., head, leg), objects (e.g., lamps, person), and scenes (e.g., bathroom, corridor) (Zhou et al., 2015). Levels of class-selectivity, (e.g., units selective for high-levels features) has been shown to be lower in early layers than in later layers (Morcos et al., 2018). This progression from low to high-level features from early to late layers suggests that selective units do not emerge by chance, but rather they reflect the hierarchical nature of representations in neural networks. However, this is not a strict hierarchy and does not imply that neural networks are modular or have dedicated layers for particular levels of analysis. Units showing selectivity for high-level features may emerge in early-layers and selective units for low-level features may be found in late layers (Zhou et al., 2015, 2017; Guest and Love, 2019).

Zhou et al. (2017) rotated learned representations in order to investigate whether ‘detectors’ (e.g., with an IoU $>$ .04 according to their network dissection selectivity measure) are axis-independent. If interpretability is axis-independent, then detectors would emerge when representations are projected to any axis. In which case, interpreting units would be no more meaningful than interpreting randomly rotated representations. On the other hand, if detectors emerge due to the model converging to a specific basis, then the units could be said to be meaningful. Previous research has claimed that there is no difference between detectors and random linear combinations of units, suggesting that units are not meaningfully aligned with axis (Szegedy et al., 2013). Zhou et al. (2017) found that the number of detectors drops by 80% when random rotations are applied. When the size of the rotation is systematically varied, they find that the number of highly selective units decreases as the size of the rotation increases. These findings demonstrate that interpretability is not axis-independent, and that selective units are meaningful.

### 2.3.2 Factors that affect selectivity

In neural networks, the representations are not stipulated by the modeller. In other words, the levels of sparsity and selectivity are not decided in advance. Rather, the task determines the appropriate representations, which are then discovered by the model (Plaut and McClelland, 2000, 2010). Or put differently, the appropriate representations for a task are whatever works! If a model solves a task, then the representations must be appropriate. Studies have identified many factors that affect the level of selectivity or the number of highly selective units.

**Dataset and task** The richness of the dataset has been shown to affect the number of highly selective unit (Zhou et al., 2015, 2017, 2018b). For example, a greater number of selective units were found in CNNs when they were trained on a scene classification task (e.g., beach, chapel, hallway etc), compared to an object classification task (e.g., balloon, beagle, desk etc). One possible reason is success on the objects task requires representing a single object, which is typically in the centre of the image. On the other hand, scenes might be recognized by the co-occurrence of multiple objects, e.g., lamp and bed for 'bedroom', or sink and bath for 'bathroom' etc. That is, the pressure to represent multiple objects as compositional elements of scenes led to higher numbers of highly selective units.

Several authors have implied that the arbitrariness (or systematicity) of a dataset will affect selectivity (e.g., Hinton et al., 1986; Morcos et al., 2018; Vankov and Bowers, 2017). However, in the case of Hinton et al. (1986) these claims relate to item-selectivity, and it is not clear how this relates to class selectivity. The claims from Morcos et al. (2018) were not empirically tested and Vankov and Bowers (2017) found no effect of arbitrariness. As such, the role of arbitrariness on selectivity remains an open question.

One task related factor that impacts on selectivity relates to the superposition-catastrophe. This occurs in RNNs, which represent sequence data over a set of units, and is caused by multiple, overlapping distributed representations forming ambiguous patterns. The chance of a model suffering from the superposition-catastrophe increases with sequence length. Bowers et al. (2014)

suggested that models with highly selective units were less susceptible to the superposition-catastrophe than models with low selectivity. Bowers et al. (2014) trained models to recall various length sequences of words and found more highly selective units in models trained on long sequences than those trained on short sequences. Bowers et al. (2014) suggested that models utilized highly selective units in order to avoid the superposition-catastrophe. They also varied the dataset size and found that for the small dataset, units were selective for individual words. However, for the larger vocabulary, units were selective for individual letters. This study shows that for RNNs, selectivity is higher for long sequences than short ones. It also shows that units might switch from representing high-level features (e.g., words) to lower-level (e.g., letters) in response to dataset size.

**Models** Aspects of the model architecture have been shown to affect selectivity. Berkeley et al. (1995) trained models to to classify logic problems and found that units had highly interpretable 'banding-patterns' (see figure 2.1). Their models used 'value units', that have a Gaussian activation function. They only respond strongly in response to a narrow range of net input strengths, increasing the input beyond this range causes a drop in output. Berkeley et al. (1995) suggested that the value units might be responsible for the banding patterns, however similar results were also reported for a network using sigmoid units (Berkeley and Gunay, 2004). Berkeley and Gunay (2004) also highlights the importance of network size on finding banding patterns, which they claim will only emerges in the smallest possible networks (fewest units) that can solve the task. This suggests that there may be a trade off between computational capacity and interpretability. However, Zhou et al. (2017) found a greater number of highly selective units in deeper models than in shallow models. They also found that increasing the width of a model (e.g., the number of units per layer) increased selectivity, but only up to a point, after which the number of highly selective units plateaus.

A model's selectivity for particular features is not dependent on a particular weight initialization, models trained from different initializations will learn selective units for similar features (Li et al., 2016; Olah et al., 2020).

**Training** Zhou et al. (2017) showed that the number of highly selective units increases over training. They also found that during training, units can change the feature that they are selective for. For example, a unit in a model trained on scene classification was selective for the colour 'green' early in training, which became more specific as it was later selective for 'grass', and finally it had highest selectivity for 'baseball field'.

The use of regularization during training has been shown to impact on selectivity. Morcos et al. (2018) found that selectivity is lower for models trained with batch-normalization than those without. Zhou et al. (2017) also found that using batch-normalization reduces the number of highly selective units. However, the number of detectors was higher for models with dropout regularization than those without. Gale et al. (2018) systematically varied the level of dropout used to train MLPs and found a small reduction in the number of highly selective unit for dropout of .2 compared to no dropout or dropout of .5. However, for dropout levels of .7 and .9 there was a dramatic increase in the number of highly selective units.

### 2.3.3 Neural networks as models of human cognition

Some researchers claim that deep learning models are good models of human neural processing (e.g., visual ventral processing; Rust and Dicarlo, 2012; DiCarlo et al., 2012; Yamins et al., 2014; Kornblith and Tsao, 2017; Zhuang et al., 2021; Higgins et al., 2020). For example, Yamins et al. (2014) demonstrated a correlation between object categorization performance in the penultimate layer or a deep CNNs, and individual inferior temporal (IT) neural unit response data. CNNs trained on object categories predicted neural activation as well as model trained on neural activation data. CNNs could explain nearly 60% of the variance in neural responses, leading to the claim that CNNs trained to classify objects are good models of IT unit activity (Kornblith and Tsao, 2017). Subsequent research has found that CNNs trained with unsupervised learning may be a better fit to neural data than CNNs trained on large supervised datasets (Zhuang et al., 2021; Higgins et al., 2020). However, despite the good fit for neural activation, there are several objections to treating CNNs as models of the brain. One objection relates to CNNs' *performance* in response to adversarial attacks. Adversarial attacks are inputs to a neural network that are designed to fool the model into giving an unexpected response (Szegedy et al., 2013; Nguyen et al., 2015). The existence of such images highlights the fact that CNNs and humans prioritise different cues when recognizing objects. Humans recognise objects primarily based on their shape, rather than by the size or texture (Landau et al., 1988), wheras CNNs have been shown to focus more on textures rather than on shape (Geirhos et al., 2019). That is, they are biased more towards high-spatial-frequency details than humans are. This bias towards texture allows adversarial attacks to fool CNNs, yet be imperceptible to humans. Another implication of the bias towards texture is that CNNs can learn datasets that would be impossible for a human, such as learning to categories thousands of 'noisy' images of seemingly random pixels (Zhang et al., 2016). The fact that CNNs can exhibit such un-human-like performance, is ground enough for some researchers to reject them as models human vision (Tsvetkov et al., 2019; Dujmović et al., 2020).

Another reason for caution around claims that neural networks learn representations that are similar to the brain relates to the fact that neural networks have a high degree of changeability in the representations they learn. The representations learned by neural networks (i.e., sparsity and selectivity) arise in response to task demands (Plaut and McClelland, 2000, 2010). However, the representations are also sensitive to model parameters (e.g., number of layers, number of units) and training parameters (e.g., regularization; the amount of training) (Zhou et al., 2017; Morcos et al., 2018; Gale et al., 2018). This variability in terms of the representations learned by neural networks makes interpreting their similarity to humans potentially challenging.

### 2.3.4 What are the implications of high selectivity?

Having established that highly selective units are found in distributed representations, it is reasonable to ask what their significance is.

#### 2.3.4.1 How important are selective units?

As discussed above, the contribution made by each unit can in theory been understood through analysis of the outgoing weights, but the complexity of neural networks makes this impractical. Rather than analyse the complete model to understand its workings, the importance of units is typically inferred indirectly, by removing (e.g., lesioning) units and comparing the performance of the lesioned and unlesioned models.

Morcos et al. (2018) performed lesioning on large CNNs trained on imageNet and found that class-selectivity was a poor predictor of the importance of a unit. Lesioning units with high selectivity led to a similar drop in accuracy as lesioning units with low selectivity. However, for deep models they found a strong negative correlation between selectivity and unit importance in the early layers. This may be because units in the early layers typically have low class-selectivity, with higher selectivity for low-level features which are important for many classes (Zhou et al., 2015, 2017). Zhou et al. (2018b) also found a weak negative correlation between selectivity and unit importance. However, they also analysed the importance of units to each class and found that highly selective units were important to a subset of classes. For example, lesioning a unit with selectivity for 'wheels' only led to a -.2% drop in overall accuracy, but led to a large drop in accuracy for 'Model-T Ford' (-41%), 'bicycle' (-31%) and 'unicycle' (-16%). For the class with the largest drop in accuracy (e.g., Model-T Ford') the size of the drop in accuracy (e.g., -41%) and selectivity of the unit had weak negative correlations. In the case of imageNet, where there are 1000 classes, a complete failure for one class would only result in a .1% drop in total accuracy. Zhou et al. (2018b)'s findings suggests that the importance of highly selective units is better understood at the level of class-accuracy rather than total accuracy.

However, units are not always important for a feature that they have high selectivity for, as demonstrated in two studies from Radford et al. (2017) and Donnelly and Roegiest (2019). Radford et al. (2017) trained RNNs with 4092 units, to generate reviews by training it to predict the next character in the Amazon product reviews dataset of 82 million reviews. They identified one particular unit that predicted the sentiment of the review (e.g., positive or negative), referred to as the 'sentiment unit'. By setting a threshold to the activation of this unit they could interpret the unit with respect to the input (e.g., sentiment) with 92.3% accuracy (the full model achieves 92.88%). Setting the activation of the unit to either -1 or 1 causes the model to generate negative or positive reviews. This suggests that almost all of the relevant information about sentiment that is available to the model is represented by this unit. Donnelly and Roegiest (2019) replicated the model from Radford et al. (2017), and also found a highly predictive sentiment unit. They confirm the high performance of this single unit, but also note that there is almost no drop in performance when the unit is lesioned. Further, they find that they could predict sentiment with better performance by fitting classifiers to 100 randomly selected units than from the sentiment unit alone, suggesting that the sentiment related information in the model was not localised at the sentiment unit. Donnelly and Roegiest (2019)'s finding suggest that while the sentiment neuron might be sufficient to predict sentiment, it is not necessary. It also demonstrates that whilst a high amount of sentiment relevant information is available to the sentiment unit, the same information is also distributed around the other units. Donnelly and Roegiest (2019) shows that just because a unit is highly selective for a feature, it does not imply that the model is reliant on the unit for high performance regarding that feature. These studies illustrate the fact that high interpretablity with respect to the input, does not equate to interpretability with respect to the output. These models do not appear to represent sentiment through a single dedicated sentiment 'detector', but rather sentiment was represented in a distributed manner, to which the sentiment neuron was only one of many contributing neurons.

Morcos et al. (2018) has suggested that models with good generalization performance have reduced selectivity compared to those with poor generalization. They found that models with batch-normalization had lower levels of selectivity and better test-set accuracy than models trained without batch-normalization. They suggest that one mechanism by which batch-normalization improves generalization is by reducing the levels of selectivity. However, in Bowers et al. (2016), their recurrent models *only* succeeded in generalizing to new items when they had learned highly selective units. This demonstrates that highly selective units can support generalization and may even be necessary for good generalization in some circumstances.

Leavitt and Morcos (2020b) used a regularizer which included a selectivity term which allowed them to manipulate the selectivity of units as they trained models. They found that there was little impact of reducing the level of selectivity on model accuracy. This is inconsistent with research suggesting that mixed populations with a variety of levels of selectivity have greater flexibility, reliability and efficiency than populations with exclusively high selectivity (Zylberberg, 2018; Jeffrey Johnston et al., 2020). Leavitt and Morcos (2020b) found that increasing the selectivity was harmful to performance. In Leavitt and Morcos (2020a), they again used a selectivity term in the regularizer as they trained models. They found that reducing selectivity increases robustness to image corruption (e.g., adding blur or pixelating images), but increases vulnerability to adversarial attacks (e.g., images designed to fool neural networks). They suggests that there is a causal-relationship between selectivity and robustness.

These results show that high selectivity does not necessarily equate to high importance. Yet highly selective units may be important for accuracy on a subset of classes and in some circumstances, they might be essential for good generalization or model robustness. However, if selectivity is too high, or there are too many highly selective units, this might harm generalization performance, performance in general, and increase vulnerability to adversarial attacks.

#### 2.3.4.2 Limitations of Selectivity Analysis

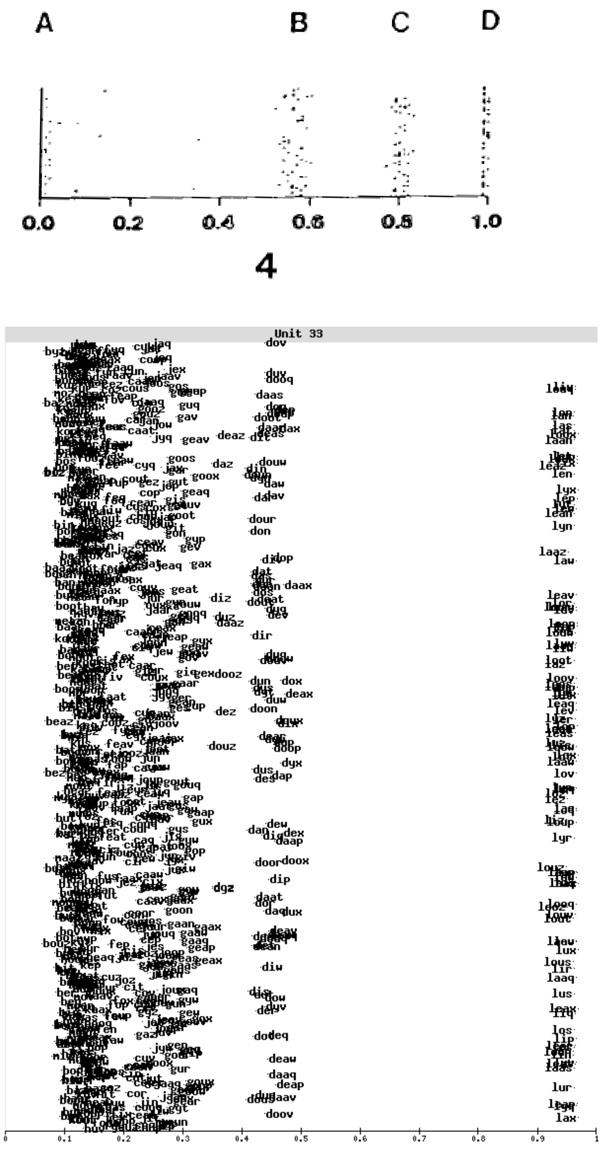


Figure 2.2: **Highly selective units with structured representations for multiple classes** *Top*: Unit 4 taken from Berkeley et al. (1995) showing a unit with four distinct bands of activation. *Bottom*: Unit 33 from Bowers et al. (2014). This image was not published and is used with the authors permission (personal correspondence, March 2018).

Selectivity analysis is a useful tool for summarising a unit's activation and providing a simplified description. But as with all simplifications, it is important to note that some information is necessarily absent in the description. A unit's selectivity is typically reported for whichever class (or item or feature) has the highest selectivity, in a one vs all manner: e.g., high selectivity for $x$ rather than $y$ or $z$ etc. However, units may be structured in ways that are not well captured by one vs all descriptions. Figure 2.2 shows two such units. The image in the top panel is taken from Berkeley et al. (1995). All items in band B relate to the connective IF...THEN; band C related to NOT BOTH...AND; and band D relates to OR. The items in band A have no activation and do not share a common feature. Nonetheless, the unit has structured activations into four bands. If the unit was described only as being selective for the items in band D, this gives an incomplete description of the activations. The lower panel in 2.2 shows a unit from Bowers et al. (2014) with high selectivity for the letter 'l'. However, the unit does not simply give low activation to all words that do not begin with 'l'. All words beginning with 'd' have intermediate activation between .35 and .5, while all words beginning with 'b' have activation below .15. Whilst this unit does not have neat bands as in Berkeley et al. (1995), there is still some structure for items with activation less than .5. These units are entirely consistent with the claim that units in neural networks are can contribute to the representations of multiple items. These units have high selectivity for items in band D (top) or the letter 'l' (bottom), yet their patterns of activation suggests that they also contribute to the representation of other features. Selectivity provides a simplified summary with respect to one class, but not a complete description of a unit's responses to a dataset.

#### 2.3.4.3 Discrete measures of selectivity

I have presented selectivity as a continuous measure ranging from low (i.e., units show no preference for any one item, feature or class), to high (i.e., high activation is associated with one item, feature or class). However, some researchers prefer to report selectivity with reference to some threshold on selectivity values, above which units are considered to be 'detectors' (Zhou et al., 2015, 2017), or 'localist' units (Bowers et al., 2014, 2016; Vankov and Bowers, 2017). This approach has been fruitful insofar as they were able to report differences in the number of detectors or localist units for different conditions. For example, Zhou et al. (2015) reported that there were more detectors for low-level features in early layers than late layers. Bowers et al. (2014) found that there were more localist units for models trained on long lists than short lists.

However, one limitation of this approach of counting the number of units above some arbitrary threshold relates to consistency. Both Zhou and Bowers have adopted this approach across a series of papers, but they both changed the threshold that separates highly selective units and units with low selectivity. Bowers used the same measure (B-sel) in three studies, but set the threshold for localist units at .5 (Bowers et al., 2014), .3 (Vankov and Bowers, 2017) or .1 (Bowers et al., 2016). Zhou defined detectors as having a precision score > .75 in Zhou et al. (2015), but as having an IoU > .04 from the network dissection method in Zhou et al. (2017). By changing the criterion, it is difficult to make comparison between studies, even from the same lab group. Similarly, if one asks "how many detectors or localist units are in this model", they might get different results depending on which threshold they use.

Another limitation of focusing only on units with selectivity above some threshold is that it directs interest away from units with lower selectivity. Highly selective units are not more important in terms of their overall contribution to model accuracy then units with low selectivity (Morcos et al., 2018; Zhou et al., 2018b). There are large differences in selectivity between models: the most selective units might have activation patterns similar to that which would be expected of a localist unit or an object detector (Bowers et al., 2014, 2016), but for other models, the most selective units might have much lower selectivity (e.g., Zhou et al., 2017; Gale et al., 2019, 2020). In Vankov and Bowers (2017) they investigated the effects of arbitrary input-output mappings on the number of 'local codes' (e.g., units with B-sel > .1). They manipulated the arbitrariness of the data, yet across the 40 models used in these experiments they found no local codes. In dismissing units with low selectivity, Vankov and Bowers (2017) failed to address the question of whether there is an effect of arbitrariness on the selectivity of representations in neural networks. The lack of any effect here is perhaps is an example of why Morcos et al. (2018) argued that "methods for understanding neural networks based on analyzing highly selective single units, or finding optimal inputs for single units, such as activation maximization (Erhan et al., 2009) may be misleading" (pg. 10, Morcos et al., 2018).

The term 'detector' has meaning within psychology and neuroscience, detectors are individual units that represent significant events and trigger a response (e.g., Barlow, 1953, 1972; Martin, 1994). Detectors in this sense, are consistent with localist representations, but not with distributed representations (Eichenbaum, 2017). Zhou does not give definitions for the term 'detector' other than with respect to their selectivity measures (Zhou et al., 2015, 2017). Neither do they claim that detectors are qualitatively different to units with lower selectivity. As such, Zhou et al. (2015, 2017) do not appear to be claiming that detectors in their models are equivalent to detectors as discussed in Barlow (1953). Nonetheless, describing the units as detectors implies that they have good detection ability. For example, a 'bus' detector might be expected to fire strongly for images containing a bus, and fire weakly for images not containing a bus. Gale et al. (2020) re-analysed models from Zhou et al. (2017). They found that even the most selective units had a low hit rate (e.g., the unit did not fire for many items from the selective class) and a high false-alarm rate (e.g., there was high activation for many items that were *not* from the selective class). In other words, Zhou et al. (2017) used the word 'detector' as shorthand for 'the most selective units in the model' which in this case, were neither particularly selective, nor were they good detectors.

Bowers makes strong theoretical claims about the highly selective units in his models (Bowers et al., 2014, 2016; Vankov and Bowers, 2017). He claim that these are localist units, for reasons relating to the superposition-catastrophe and to selectivity. Recurrent models superimpose multiple representations across a set of units, which may result in an ambiguous blend known as the superposition-catastrophe (Page, 2000; Von Der Malsburg, 1986; Rosenblatt, 1961). Although the superposition-catastrophe can occur in localist models (Rosenblatt, 1961), Bowers points out that increasing the sparsity of representations can reduce the risk of ambiguous blends forming, and therefore localist representations (e.g., maximally sparse) should be least susceptible. Bowers et al. (2014) identify several factors which may relate to the superposition-catastrophe, including list-length: there is a greater chance of ambiguous blends where models have to recall long lists than short lists as they have to superimpose more representations. Bowers et al. (2014) did not measure the sparsity of representations, but they do note the selectivity of units, and find that there are a greater number of 'localist units' (e.g., with B-sel > .5) in models trained on long lists than short lists. They claim that models learn localist representations to avoid the superposition catastrophe. This paper gives a compelling argument that models can contain both distributed and localist representations, which they claim is a challenge for the assumption that neural networks exclusively learn distributed representations. However, the onus is on them to demonstrate that these are localist units, rather than simply being highly selective units in a distributed representation (e.g., there are other units which are also involved in representing these words or letters). This could be demonstrated by analysing the sparsity, weights or performing lesioning. As discussed in Földiák (2009) (see table 2.1), high selectivity is possible for units in distributed representations.

As discussed in section 2.3.4.1, Bowers et al. (2016) performed lesioning on similar models to those used in Bowers et al. (2014). Models were trained to recall lists of words. Bowers et al. (2016) defined units as localist if B-sel was > .1, and found 33 localist units with selectivity for a particular letter. They lesioned the 33 localist units and found mixed effects of lesioning. Only three of the 33 of the units labelled as localist units functioned as might be expected from units in a localist model (e.g., after lesioning the model failed on ≥ 97% of words that contained the selective letter, with < 1% failure on words not containing the letter). This shows that these letter-selective units were vitally important for the recognition of words containing the selective letter. I would argue that these units are better described as being *functionally equivalent to* localist units, rather than being described *as* localist units, but this is a terminological point. The selectivity of these three units was .42, .45 and .17, meaning they would not have been identified as localist in Bowers' previous study (e.g., where localist was defined as having a B-sel > .5, Bowers et al., 2014). These three units were not the most selective units in the model. There were 13 units with B-sel > .5, but none of these had greater than 50% drop in accuracy for their selective letters. There was one unit with B-sel of .99, but it was not important for *any* class. There were eight localist units in total where was no effect of lesioning (e.g., accuracy did not drop for any words), meaning they were not uniquely important for any words. Localist models can have multiple, redundant dedicated units for a feature (e.g., Davis, 2001), in which case the predictions are less clear about the effects of lesioning a single unit. However, differences in the effect of lesioning in Bowers et al. (2016) can not be attributed to there being multiple redundant localist representations: the three units with the most localist-like failures were selective for letters for which there were other localist units. There were other units that were selective for the same feature, but had different B-sel scores, suggesting that were not identical, redundant copies of the same representation. Conversely, four of the units where lesioning had no impact were the only localist unit for that letter. The mean effect of lesioning these 33 units was failure on 25% of words containing the selective letter and less than 1% on words not containing the letter. That means that these units only contributed to the representation of words containing the selective letter, and not to words that did not contain the letter. This certainly suggests some relationship between selectivity and function. However, they only lesioned localist units, so it is not possible to see if there were qualitative differences between the effects of lesioning localist units and less selective units. The fact that on average, lesioning only caused a 25% drop in accuracy for words containing the letter suggests that there were other units which were also involved in representing words containing the selective letter. That is, these units formed part of distributed representations, which were only partially degraded by lesioning. This study appears consistent with the weak correlation between a unit's selectivity and the effect of lesioning the unit (Zhou et al., 2018b). This study, along with Donnelly and Roegiest (2019), demonstrates that it is possible for a unit to have high selectivity for a feature and yet to be of minimal importance to accuracy relating to that feature. This lesioning study shows that high selectivity does equate to localist-like behaviour as only 3 of the 33 units defined as localist functioned as would be expected of localist units. These results are consistent with the view that it is the weights, rather than the activation that define an unit's importance, and as such, units in distributed representations are hard to interpret with respect to their contribution to the output. Their lesioning study undermines their claim that units with high selectivity are localist.

It is not clear what it would mean for a model to learn both distributed and localist representations. Perhaps if a unit was both maximally sparse and selective then such a description would be apt. However, the units described as localist by Bowers are not maximally selectivity, and as such they might contribute to the representation of other items (e.g., see figure 2.2). Similarly, the representations were not maximally sparse for the letters represented by the most localist-like units, as there were other units with high selective for those letters. The lesioning suggested that in general the representations were not highly sparse, as lesioning these localist units led to only a 25% drop in accuracy. Perhaps I am being too stringent, and localist units should be defined by their selectivity (as has been argued, e.g., Barlow, 1972; Thorpe, 1995; Page, 2000; Bowers, 2010), regardless of whether they are functionally equivalent to units in localist models. Is is possible that future research will identify qualitative differences between units with high and low selectivity, such that neural networks are best understood as functioning on the basis of 'detectors', or mixed populations of distributed and localist units (Bowers et al., 2014; Zhou et al., 2015; Bowers et al., 2016; Zhou et al., 2017; Vankov and Bowers, 2017). However, so far there have been no insights from this approach other than the finding that units can be highly selective. I argue that their data is consistent with the view that distributed representations can vary in their sparsity and in the selectivity of units. As such, I will focus on selectivity as a continuous variable, rather than as a discrete property.

## 2.4 Open Questions

Having reviewed the literature on selectivity, there are several open questions which this thesis will address.

**What is best way to measure selectivity?** The research discussed here has used a variety of methods for measuring selectivity (e.g., Bowers et al., 2014; Zhou et al., 2015; Morcos et al., 2018; Gale et al., 2019, 2020). This makes it difficult to compare the levels of selectivity between different studies. Chapter 3 compares four measures of selectivity that have been used in the literature. I train a variety of models on three different datasets. I find that none of the measures are able to reflect the full range of selectivity scores found in the units, each measure displays either floor or ceiling effects. I also find that two of the measures inflate selectivity scores for sparse units. I propose that Maximum-informedness is the most reliable measure of selectivity.

**What is the effect of arbitrariness on class-selectivity?** Several authors have suggested that there will be differences in the level of selectivity for models trained on data with with systematic input-output mappings (i.e., items with similar inputs are mapped to the same output) compared to data with arbitrary mappings (i.e., items with similar input are no more likely to be mapped to the same input as items with different inputs, McClelland et al., 1995; Kumaran et al., 2016; Hinton et al., 1986; Morcos et al., 2018; Vankov and Bowers, 2017). Some of these claims suggested higher selectivity for arbitrary data, but these claims related to item-selectivity rather than class-selectivity (McClelland et al., 1995; Kumaran et al., 2016; Hinton et al., 1986). Morcos et al. (2018) made claims relating to class selectivity, but did not directly test them. Vankov and Bowers (2017) did experimentally test the effect of arbitrariness for both item-selectivity and class-selectivity, but found no effect. In chapter 4 I will directly test the effect of arbitrariness on class-selectivity. I find that class-selectivity is higher for systematic tasks than for arbitrary tasks.

**How strong is the role of the superposition-catastrophe on selectivity in RNNs?** Bowers et al. (2014) found highly selective units in RNNs trained to recall lists of words. They defined units as localist if they had a B-sel > .5. They found that the number of localist units increased with the length of the lists that models were trained on. They suggested that models learn local units to avoid the superposition-catastrophe: an ambiguous blend that forms when models represent multiple items across the same set of units. They suggested that a unit which only fires strongly for words containing the letter 'n' "appears to be a localist detector for the letter n." (pg.256, Bowers et al., 2014). However, unit's responses were only analysed for lists of one item, even if they were trained to recall longer lists. This raises the question of whether the 'n detector' had the same role for all list positions, or whether it was an 'n detector' for the first item in list, and then a 'x detector' for the second and a 'y detector' for the third item in the list etc. I replicate their study with a variety of models and extend the analysis to all list positions. I find no 'detector' in any trained models, and only a small proportion of the most selective units have high selectivity for the same feature across all list positions. Selectivity does not increase with list length suggesting that the pressure to avoid the superposition catastrophe does not have a strong effect on selectivity.

**How well does selectivity predict the effects of lesioning?** Zhou et al. (2018b) combined selectivity analysis and lesioning to show that selective units are highly important for a subset of classes. They measured the effects of lesioning with the change in total accuracy and the change in accuracy per-class (max-class-drop), and found significant correlations with various selectivity measures. They claimed that the relationship between selectivity scores and max-class-drop showed that units were specialised for a subset of classes. However, the selectivity measures they used have been shown to over-estimate selectivity, which raised the question of whether the effects of lesioning would have stronger correlations with a more reliable selectivity measure. I applied their max-class-drop to a model and found that max-class-drop did not identify specialised units as it confounded class-specific effects with the change in total accuracy. I propose a different measure of the effects of lesioning and confirm that selectivity is weakly associated with the effects of lesioning. However, I note that the the most selective class is rarely the class with the largest drop in accuracy following lesioning. I conclude that selectivity scores are unreliable predictors of the class-specific effects of lesioning.

# Chapter 3

# Selectivity Measures

## 3.1 Compare Measures

This chapter will survey and evaluate several different methods for measuring the selectivity of units in neural networks. I shall examine three measures that have been used in recent literature and propose one new measure. I apply these measures to multi-layer perceptrons (MLPs) with one or four hidden layers of varying size, using two different activations functions, trained on one of three datasets. This will provide us with a range of activation distributions. The selectivity scores from different measures are discussed in relation to a small selection of units, which are chosen to highlight the limitations and weaknesses of the different measures.

The research conducted in this chapter was used as pilot work which informed two published papers on the comparison of selectivity measures in large-scale convolutional neural networks (CNNs), for which I was a co-author (see Gale et al., 2019, 2020). The findings that CCMA and precision may overestimate class-selectivity scores for some units (Gale et al., 2019, 2020); and the proposal of maximum-informedness as a selectivity measure (Gale et al., 2020) originate from the research presented here. These publications presented data that was either from large CNNs for which I did not conduct the analysis, or was artificial data (e.g., to demonstrate the limitations of CCMA). The research here uses MLPs rather than CNNs and provides a more in-depth look at the differences between measures than was included in those publications.

## 3.2 Introduction

Researchers have developed and used many different selectivity measures, which operate on different variables relating to a unit's representation of a dataset. This includes the minimal distance between classes (Bowers et al., 2014, 2016; Vankov and Bowers, 2017); differences in class means (Morcos et al., 2018; Leavitt and Morcos, 2020b,a) and the proportion of the most active items that are from the same class (Zhou et al., 2015; Leavitt and Morcos, 2020b). The diversity in measures perhaps reflect different researchers assumptions, preferences, backgrounds and expertise.

### 3.2.1 Selectivity for a Class, Item or Feature

To analyse the selectivity of a unit, the activations for that unit are compared to some dataset property. In this thesis, the focus is on *class-selectivity*, which examines whether the activations for items from one class are different to the activations for items from other classes. However, selectivity analysis does not need to be restricted to classes, one can test the selectivity of units in relation to other sets of features (e.g., feature-selectivity), or individual items (e.g., item-selectivity).

To clarify the difference between these different approaches to analysing selectivity, one can imagine a model where the input is an image of a cat, dog or mouse, and the task is to activate the relevant output unit: either 'cat', 'dog' or 'mouse'. The dataset contains 30 images, 10 images for each of the three categories. Each image contains one animal, and therefore, an image can only belong to one of the output classes.

A unit would have high *item-selectivity* if it has strong activation for one image of a cat, but has minimal activation for all other images of cats, dogs and mice. This unit has high item-selectivity because it has high activation for just one item; but it would have low class-selectivity, as it did not fire for most images of cats. An example of item selectivity come from Bowers et al. (2014). They trained recurrent models to identify words that had been presented in a list of words. They found units that fired strongly for one word and had weak firing for all other words.

A unit would have high *class-selectivity* for the class 'cats' if it is strongly activated by all images of cats, with weak activation for all images of dogs and mice. Importantly, the three output classes (e.g., 'cat', 'dog', 'mouse') are mutually-exclusive and exhaustive. Mutually-exclusivity means an item from this dataset can only belong to one class, if an item is labelled as a cat, it is not a dog or mouse. Secondly, the three output-classes account for all items in the dataset, and as such, the list of output-class is exhaustive. Exhaustiveness means one can test units for selectivity in relation to *all* output classes in the dataset; and if scores are low, the unit has low class-selectivity. These properties also make it possible to report the mean level of class-selectivity of a model, with respect to *all* classes. The same principles of mutual-exclusivity and exhaustiveness apply for item-selectivity: one can systematically test for selectivity for *all* items in the dataset, and a unit with high selectivity for one item can not have high selectivity for a different item. An example of class-selectivity comes from Morcos et al. (2018). They lesioned units and performed correlations on the class-selectivity and unit importance, measured by the drop in overall accuracy. They found that class-selectivity was poor predictor of unit importance.

The third type of selectivity is *feature-selectivity*. For example, a unit might fire strongly to some images of cats and dogs, but no images of mice. Further inspection reveals that the unit exclusively fires whenever a cat or dog collar is visible in the image. This unit does not have high class-selectivity, as 'collar' is not one of the output classes (e.g., cat, dog or mouse). Similarly, there could be a unit which is strongly activated for images of animals outdoors (e.g., on grass) but not indoors (e.g., on carpet). I consider collars and context (e.g., indoors or outdoors) to be features of the dataset. Crucially, this list of features (e.g., collar or context) is neither mutually-exclusive, nor exhaustive. It is not exhaustive as one can propose many other features that might be relevant: image brightness, animal pose/orientation, animal breed, fluffiness etc. As such, it is not always possible to test for selectivity in relation to *all* features, but only a *set* of features. Therefore, it might not always be possible or meaningful to report the mean feature-selectivity of a model, but only the mean selectivity for a particular set of features. In some cases it is possible to test for a complete set of features, for example, testing for selectivity for *all* letters in models trained on words (e.g., Bowers et al., 2014, 2016). However, in this case the features were not mutually exclusive (e.g., words contain multiple letters), as such it might not be clear which of the features drives the unit. A unit could even be selective for conjunctions of features, such as a unit which only fires when the animal is outdoor *and* the collar is visible; or a unit that fires for words containing the letters 'a' *and* 'b'. Whilst it may be desirable to attach such specific semantic labels to a unit, as it gives a precise description of its firing, this situation can lead to a combinatorial-explosion of potential features to test for selectivity.

An example of feature-selectivity comes from Zhou et al. (2017). They investigated the selectivity of models trained on image recognition tasks. The datasets were imageNet (Deng et al., 2009), which contains images of animals and objects from 1000 classes; and Places (Zhou et al., 2018a) which contains images of scenes. They developed the BroDen dataset which allowed them to test for selectivity for 1200 visual concepts including colours, materials, objects-parts, objects and scenes. They used this same dataset to analyse feature-selectivity of models trained on imageNet or Places. They found more units with high selectivity for low-level features in early layers than in late layers. This highlights an advantage of testing for selectivity to features rather than classes: testing for different level features can give an understanding of the hierarchical nature of representations at different layers of a model. Also, they were able to compare the selectivity for the same set of features between model trained on different datasets. Feature selectivity is arguably the most appropriate method for understanding why a model gives a particular output in response to an input. By choosing features which are relevant for the task at different levels of analysis it is possible to build up a picture of the transformations and processing done by each layer in a model (e.g., Zhou et al., 2017; Olah et al., 2020; Open AI, 2020).

One difficulty associated with testing for feature-selectivity is that the process of deciding which features to test for is not trivial. First one must decide which features are potentially relevant, then one must identify all instances of the features in the dataset and label items accordingly. When deciding which features are relevant for the fashion dataset, which consists of grey-scale images of ten types of clothing (e.g., shirts, boots, dresses etc): would sleeves be a relevant feature, or are short and long sleeves separate features? Are buckles and buttons both types of fastener, or separate features? Having decided on the list of features, each item in the dataset then needs to be labelled to show which features are present in the dataset. Images in the fashion dataset are only 28 x 28 pixels, so visual inspection might be imprecise: it is easier to tell if there are buttons on a white dress than on a patterned dress. The difficulties of deciding on the features to test, and annotating the dataset could be one reason why few studies have thoroughly and systematically tested a set of potentially relevant features. Notable exception include using the letters that compose words as a set of features (Bowers et al., 2014, 2016), or the BroDen dataset of around 1200 visual concept in (Zhou et al., 2017). However, many studies that have found feature-selective units by testing for a single feature (Lakretz et al., 2019; Radford et al., 2017). Other studies appear to have identified feature-selective units post-hoc (e.g., Zhou et al., 2015; Karpathy et al., 2016). Olah et al. (2020) identified units in models trained on visual object recognition that were selective for image patches with adjacent high-contrast and low-contrast areas. They state that they did not anticipate finding selective-units for this feature. Having identified these selective units in one model, they subsequently identified them in many other models trained on visual tasks. This demonstrates that it is not always intuitive to anticipate which features will be relevant. Finally, a list of features might only be relevant for one particular task. For example, the BroDen dataset (Zhou et al., 2017) contains many features relevant for real-world visual recognition tasks such as imageNet and Places (e.g., colours, textures, object-parts etc), but they would not be relevant for different tasks (e.g., handwritten-digit recognition, short-term memory, etc.).

In the examples so far, selectivity for classes, items or features relates to discrete categories or properties, e.g., an item is either in class A, or it is not; a feature either is present or it is not. This probably relates more to the ease of testing than to any theoretical claim that units can only represent discrete properties. It is entirely conceivable for a unit to represent a continuous property. For example Radford et al. (2017) trained a recurrent model to generate product reviews. They identified a single unit with selectivity for the sentiment of the review, with high activation indicating a positive review and low activation indicating a negative review. However, a systematic search for such units requires each item in the dataset to be labelled in a continuous manner in relation to the relevant continuous variable. A different approach allows selectivity to be tested against discrete properties, but for activation to seen as representing this property in a continuous manner. For example, a unit might show selectivity for faces such that activation is high for images that contain face and low for images that don't. However, activation is even higher where there are multiple faces, or a face close-up; activation is moderate for face-like items such as partially-occluded faces; drawings of faces; or other round objects. As such, this unit could be said to represent 'face-ness' in a continuous manner (or perhaps 'similarity-to-faces' or ‘number-of-faces’ etc). However, there are few examples in the literature of selectivity being described in relation to a continuous property. One example is from Karpathy et al. (2016), who identified units in language models which track line-length (e.g., the distance between ‘new-line’ characters).

For our purposes, class-selectivity truly is a discrete property, class membership within a dataset is all or none, and a model is trained to treat all instances of a class as being equally good examples. As such, class-selectivity can be referred to in discrete terms. If ‘faces’ is an output class in the example above, then high selectivity for faces will mean that items from other classes should have lower activation than items from the face class. Any within-class differences (e.g., different levels of activation for items containing one or multiple faces; or differences between non-face items relating to their similarity to faces) are less important than differences in activation between classes.

In summary, by focusing on class-selectivity, one can utilize an agreed on property of the data (e.g., class membership); for which the data is already labelled; all labels are known to be relevant for the task; and labels for different datasets are at the same level of analysis (e.g., output-classes are high-level features). This makes it possible to compare the class-selectivity models trained on very different tasks.

### 3.2.2 Measuring Selectivity

Class-selectivity (or just ‘selectivity’ from here on) is a measure of how much the activation of a unit is different for one class, compared to all other classes. Selectivity can be thought of as a continuum from low to high. A unit with low selectivity all classes would have similar levels of activation. At the other end of the continuum, a highly-selective unit would have high activation for *all* items from a given class and low activation for all other items. I do not expect that these claims are controversial. However, things are a little more complex in the middle of the continuum as there are many ways in which a unit can be somewhat selective.

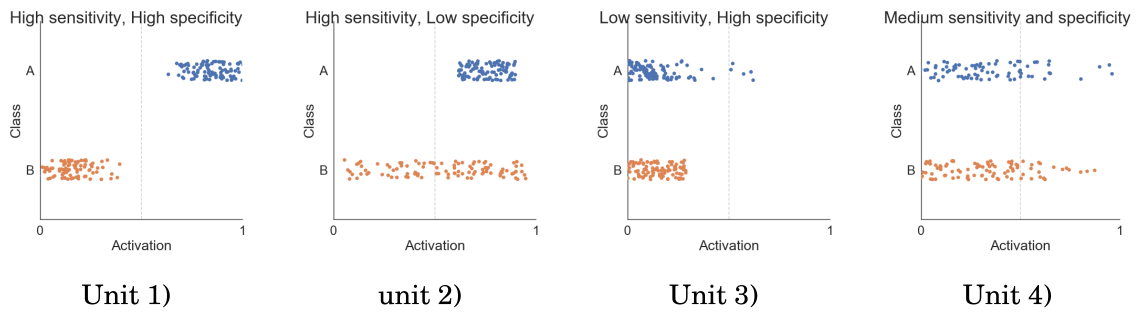


Figure 3.1: **Hypothetical units with with low, medium and high selectivity** The $x$-axis shows the normalized activation and classes are shown on the $y$-axis. Unit 1 has high selectivity for class A (blue markers). Units 2 and 3 have moderate selectivity for class A. Unit 4 has low selectivity for class A.

There are two properties that describe performance in a binary classification task that are useful for thinking about intermediate levels of selectivity: sensitivity and specificity. Figure 3.1 shows four hypothetical units for a task with two classes, A and B. There are 100 items in each class, and to keep things simple, activations will be described as either high (e.g., > .5) or low (e.g., < .5), which is marked with a grey line in the plots. A unit's sensitivity, also known as the hit-rate, for class A is the proportion of items from class A with high activation. Units 1 and 2 have high sensitivity for class A as they strongly activate all items from that class. A unit's specificity for class A is the proportion of items that are *not* from class A with high activation. Put differently, specificity for class A is an indication of how specific strong activations are to class A. Units 1 and 3 have high specificity for class A as there are no items from class B with high activation. Unit 2 has low specificity for class A as there are lots of items from class B with strong activation. When considering the selectivity for class A, items from class B with high activation are known as false-alarms, and as such, specificity is high if the false alarm rate is low.

For the present discussion, the main point to emphasise is that units 2 and 3 both display moderate levels of selectivity for class A. For unit 2, selectivity would be higher if there were fewer items from class B with high activation. For unit 3, selectivity would be higher if there were more items from Class A with high activation.

A good selectivity measure should reflect both the sensitivity and specificity of a unit's activations in relation to a given class. A good selectivity measure should give a high selectivity score *only* if the unit has high sensitivity *and* high specificity. One implication of this view of selectivity relates to very sparse units: units where the majority of items have an activation of zero. If there are many items from class A with with an activation of zero, the sensitivity for class A will be low, meaning class A can not have very high selectivity.

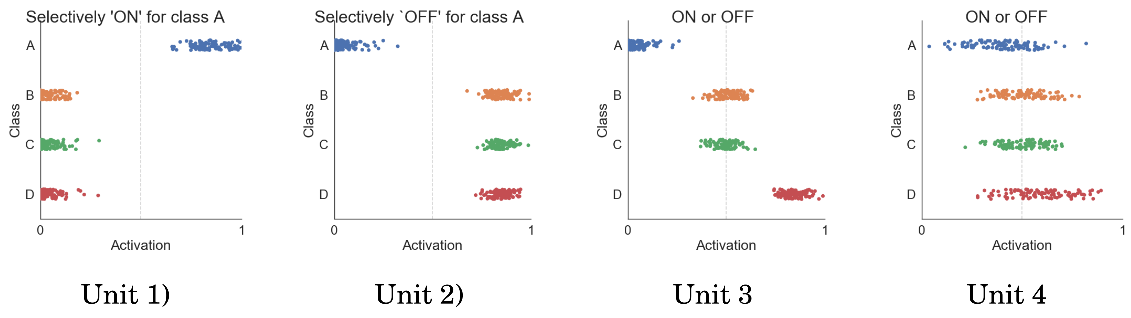


Figure 3.2: **Hypothetical units displaying ON or OFF selectivity** The $x$-axis shows the normalized activation and classes are shown on the $y$-axis. Unit 1 is selectively ON for class A (blue markers), as class A has higher activation than the other classes. Units 2 is selectively OFF for class A, which has lower activation than the other classes. Units 3 and 4 could be described as being selectively OFF for class A or selectively ON for class D (red markers).

So far, I have have described a unit with high selectivity for class A as a unit where items from class A have higher activation than items from other classes. However, a unit can also display selectivity for a class that has *lower* activation than all other classes. Figure 3.2 shows hypothetical data with four classes. Units 1 and 2 both respond in a selective manner to class A. For unit 1, activations are higher for class A than for other classes, as discussed above. For unit 2, the activations for class A are lower than for the other classes. It might therefore be appropriate to test the selectivity of each unit twice, once to test to see if the unit is selectively ON for any classes, then a second time to test for OFF selectivity. However, unit 3 is 3.2 demonstrates that it is possible for a unit to be selectively ON for one class and *also* selectively OFF for another class. Whilst unit 3 is perhaps an extreme example, unit 4 presents a more likely situation, where a unit is not highly selective for any class, but it could be reported as have low OFF selectivity for class A, or low ON selectivity for class D. Units 3 and 4 highlight a limitation of selectivity analysis, in that it describes a units activation in a one-vs-all manner: the activations for *one* class are compared to the activations of all other classes. If unit 3 is describes as selectively OFF for class A, it obscures the high activations for class D. One could find ways of describing the activations for unit 3 more thoroughly, with reference to both classes. However, more complex descriptions of unit activation would make comparison between units or models more difficult.

One approach to dealing with OFF units is to ignore them, for example, by only analysing the most active items (e.g., study measuring selectivity with ‘precision’, discussed below, Zhou et al., 2015, 2017; Rafegas et al., 2020). Some measures (e.g., B-sel and CCMA, see below for details) have been used to score selectively ON classes with positive values and OFF classes with negative values. However, researchers can ignore OFF units by reporting the the selectivity of each unit for the class with the maximum selectivity score, which will necessarily be a class that is selectively ON (Morcos et al., 2018; Zhou et al., 2018b). This suggests that OFF units are either though to be unlikely to occur or are unimportant. A different approach was used in Bowers et al. (2014). They also used a measure which scores ON units with positive values and OFF units with negative values, but they reported the selectivity of units based on their absolute values. As such they find both ON and OFF units. They also are unsure of the importance of OFF units, stating that their findings are not changed if they only consider ON units. They suggest that OFF units might have emerged for idiosyncratic reasons relating to their models, data or task. Later, Bowers et al. (2016) used the same measure and found four OFF units and 29 ON units with high selectivity in RNNs trained to recall lists of words. Units were selective for letters that composed the words. For ON units (e.g., the unit has higher activation for words containing the letter ‘a’ than words that do not contain an ‘a’), lesioning led the model to fail on an average of 25% of words that *did* contain the letter (e.g., ‘a’) and no drop in accuracy for words that did not. For the OFF units the results were different. For two of the OFF units, there was no effect of lesioning. For the other two OFF units, lesioning caused the model to fail on all words that did *not* contain the letter, and none of the words that *did* contain the letter. In other words, these units acted like ON units rather than OFF units. I will focus units that are selectively ON for a given class, unless otherwise stated.

The aim of this chapter is to evaluate the different measures and to identify any biases that might go unnoticed - especially where measures are applied to a large number of units without careful inspection of visualization of the underlying activations. With that in mind, I will discuss the performance of the selectivity measures with references to visualizations of units.

## 3.3 Methods

### 3.3.1 Selectivity measures

The following section will give details of the four selectivity measures tested in this chapter. Three of the measures (B-sel, CCMA and precision) are included as they have been used in recent research. The fourth measure (maximum-informedness) is a measure that I propose as an alternative to the first three, which has now also been included in a published paper as a selectivity measure (Gale et al., 2020).

For discussing the selectivity or activation of a given class in relation to all other classes, I will use the terms ‘class A’ and ‘not A’. For all measures I calculate the selectivity of each class. In the main analysis, I will report ON selectivity only, which is scored in the range 0 to 1 for all measures. Each unit is reported as being selective for the class with the highest score. Later I will discuss the impact of testing for both ON and OFF selectivity. OFF selectivity is also scored in the range 0 to 1, and a unit is reported as being selective for the class with the highest ON or OFF selectivity. For example if class A has ON selectivity of .7, and class B has OFF selectivity of .8, then the unit would be selectively OFF for class B.

**Bowers’ Selectivity (B-sel)** is based on a measure used in research from Jeff Bowers’ lab (e.g., Bowers et al., 2014, 2016; Vankov and Bowers, 2017). B-sel is defined as the minimal difference in activations between items from class $A$ and all other items (i.e., the distance between the least active item from class $A$ and the most active item that is not from class $A$). This measure was designed to test for highly-selective units with activation patterns similar to units in localist models, rather than to measure the selectivity of units in general. In other words, this measure was designed with the highest possible levels of selectivity in mind. Any unit with a B-sel > .5 was reported as being a ‘localist’ unit (i.e., Bowers also refers to such units as ‘grandmother cells’ Bowers et al., 2014, 2016, etc.). Based on the B-sel scores of units in recurrent models trained on a short-term memory task, Bowers et al. (2014) claimed that neural networks use localist units in certain situations.

Bowers’ selectivity for a given class (e.g., class $A$) is calculated by subtracting the maximum activation *not* from class $A$ from the minimum activation for class $A$. If the result is $> 0$, then this is the B-sel, otherwise the B-sel for that class is 0. A unit is reported as being selective for the class with the highest B-sel score, apart from for units where all classes have a selectivity score of zero, in which case there is no selectivity score reported for that unit.

**Algorithm 1** Calculate B-sel (ON)

```
if $min$(class $A$) > $max$(not $A$) then
    B-sel = $min$(class $A$) - $max$(not $A$)
else
    B-sel = 0
end if
```

B-sel scores are in the range 0 to 1, with values greater than zero indicating that the given class is linearly-separable from all other classes (e.g., I use the term linearly-separable to mean that it is possible to threshold the activations such that all items from class $A$ are on one side of the threshold and all other items are on the other side). When class A is not linearly-separable the unit will have a B-sel of zero.

I will also report the selectivity of units when tested for ON or OFF selectivity.

**Algorithm 2** Calculate B-sel (OFF)

```
if $max$(class $A$) < $min$(not $A$) then
    B-sel = $max$(class $A$) - $min$(not $A$)
else
    B-sel = 0
end if
```

OFF selectivity is also scored in the range 0 to 1. Where units are tested for ON and OFF selectivity, the unit will be reported as being selectivity for whichever class has the highest value.

B-sel, as defined here is slightly different from the original version used in Bowers et al. (2014, 2016); Vankov and Bowers (2017). They scored ON units in the range 0 to 1, and OFF units in the range -1 to 0. In the version used here, both ON and OFF units are scored in the range 0 to 1, which makes it easier to compare the magnitude of the selectivity score (e.g., how selective is this unit?) rather than on the direction. This also avoids ON and OFF units cancelling each other out where mean selectivity is reported.

**Class Conditional Mean Activation Selectivity (CCMA)** was originally used as a selectivity index in neuroscience (e.g., Freedman and Assad, 2006; Tsao et al., 2006), and has been used to measure selectivity in neural networks in several papers by Ari Morcos (e.g., Morcos et al., 2018; Leavitt and Morcos, 2020b,a), as well as in papers from other groups (Zhou et al., 2018b; Gale et al., 2019, 2020). Morcos has used CCMA to argue that highly selective units are not necessary for good model performance, and that they may even be harmful (Morcos et al., 2018; Leavitt and Morcos, 2020b). However, it has been argued that CCMA can give misleadingly high scores for very sparse units (Gale et al., 2019, 2020).

The CCMA for class $A$ gives the mean activation of all images in class relative to the mean activation of all images not in class $A$:

$$CCMA = \frac{\text{mean(class } A\text{) - mean(not } A\text{)}}{\text{mean(class } A\text{) + mean(not } A\text{)}} \tag{3.1}$$

Values can be in the range -1 to 1, with positive values indicating that the given class' mean is higher than the mean of other classes, negative values indicating that the given class has a lower mean than the other classes (e.g., the unit is selectively OFF for that class) and a score of zero where the given class has an equal mean to the other classes.

In previous research (e.g., Morcos et al., 2018; Zhou et al., 2018b), with units labelled as as being selective for the class with the *highest* CCMA score. This will necessarily mean that the most selective class is always selectively ON. I will follow this method when testing only for ON selectivity. When testing for ON and OFF selectivity, the *absolute* value of CCMA scores will be used such that ON and OFF selectivity are both in the range 0 to 1.

One nice aspect of CCMA is that its values indicate the a ratio of the means (Tsao et al., 2006). A CCMA of .33 means that the class $A$ mean is twice as high as the mean of other items (e.g., if the class A mean is .6 and the not A mean is .3, CCMA = $\frac{.6-.3}{.6+.3} = \frac{.3}{.9}$ = .33); CCMA of .5 indicates a 3:1 ratio; .8 means a 10:1 ratio and a CCMA of .98 means the class A mean is 100 times higher than the not A mean.

**Precision** is a term from the information-retrieval literature which relates to the proportion of retrieved documents that are relevant (Zeugmann et al., 2011). In the context of class-selectivity it is a measure of the proportion of items above a given threshold from a given class. Precision has been used to test the selectivity of units in neural networks in several papers (e.g., Rafegas et al., 2017, 2020; Zhou et al., 2015). In Zhou et al. (2015) they used a precision measure based on the 60 most active items. However, in Zhou's method, humans labelled segmented versions of the most active items with a common feature (e.g., feature-selectivity). They suggested that any unit with precision > .75 was an 'object-detector' for the feature that it was selective for (e.g., a dog detector etc). With this method they found that most detectors in early layers were selective for low-level features, and the number of detectors for high-level features increasing for later layers. The network dissection method for measuring selectivity can also be seen as a precision based measure as it involves thresholding activations, analysing the items the most strongly drive the unit, and disregarding all items that are below this threshold (Zhou et al., 2017).

Precision as used here will differ from previous research (e.g., Zhou et al., 2015; Rafegas et al., 2017; Leavitt and Morcos, 2020b) where they used a threshold size of 60, 100 or 1000 items respectively. The datasets have either 50 or 1000 items per class, so using the same size threshold for all datasets would make comparison between datasets difficult. For example, if the threshold size is 60 items, the highest possible score for iris, which has 50 items per class, would be .83 (e.g., 50 items out of 60). On the other hand, for MNIST and fashion, which have 1000 items per class, a threshold of 60 might lead to ceiling effects. That is, by using too small a threshold, precision might overestimate the selectivity of units (Gale et al., 2019, 2020). For example, a unit with a precision of .95 (for the class ‘Monarch-butterfly’, see figure 3 in Gale et al., 2019) based on the 60 most active items. There are around 1000 items in this class, and they note that the mode activation for this class was zero. They suggest that it is misleading to describe the unit as highly selective based on the 57 highly active items (e.g., in the top 60), when there are items from this class with high, medium, low and no activation. Precision in this study will therefore consider a larger proportion of items, equal to the class-size, as suggested in the appendix of Leavitt and Morcos (2020b). To account for any difference in class sizes (e.g., if accuracy is higher for one class than other classes), precision will use an adaptable threshold, equal to the number of correct items in the class being analysed. This avoids any issues that could arise from different size classes and in theory, a precision score of 1.0 should indicate that all items from a given class have higher activation than all items from other classes. However, for units with sparse activation, were the number of items with activation > 0 is fewer than the number of items in the class being analysed, the threshold will be set to include all active items. In this case a score of 1.0 would indicate that all active items are from the same class, although this would not include all member of that class.

$$Precision(ON) = \frac{\text{number of items above the threshold from class } A}{n \text{ items above the threshold}} \tag{3.2}$$

Where $n$ is equal to the number of correct items in class $A$, unless the number of active items is smaller than the class size. In this case $n$ equals the number of items with activation greater than zero.

Where units are tested for ON and OFF selectivity, OFF precision is calculated as:

$$Precision(OFF) = \frac{\text{number of items below the threshold from class } A}{n \text{ items below the threshold}} \tag{3.3}$$

Where $n$ is equal to the number of correct items in class $A$, unless the number of items with zero activation is greater than the number of items in class $A$. In this case, $n$ equals the number of items with activation of zero.

**Maximum Informedness (Max-info)**, also known as the bookmaker odds is a multi-class generalization of Youden’s J statistic which estimates the probability of an informed decision, rather than a guess (Powers, 2003, 2011). In the context of selectivity, it involves testing the sensitivity and specificity for a class at multiple thresholds, and identifying the threshold with the highest sensitivity and specificity. Sensitivity (also known as the hit-rate or true-positive-rate) refers to the proportion of items from the given class that are above the threshold; specificity (or the true-negative-rate) is the proportion of items *not* from the given class that are below the threshold. Maximum-informedness was used to measure selectivity in Gale et al. (2020) to challenge the claim that there were ‘detectors’ in models trained by Zhou et al. (2017). Gale et al. (2020) found that even at the best thresholds, 99% of the images with activation above the threshold, did not contain the feature that the unit was a ‘detector’ for.

$$\text{Sensitivity} = \text{Proportion of class A above the threshold} \tag{3.4}$$

$$\text{Specificity} = \text{Proportion not from class A below the threshold} \tag{3.5}$$

$$\text{Informedness (at a given threshold)} = \text{Sensitivity} + \text{Specificity} - 1 \tag{3.6}$$

$$\text{Max info} = \max(\text{Informedness}) \tag{3.7}$$

Previous research has only applied max-info to test for on selectivity (e.g., Gale et al., 2020). Here I will test for OFF selectivity. The same equation is used, but sensitivity and specificity refer to the proportion of items *below* the threshold rather than above the threshold. OFF selectivity will be scores in the range 0 to 1.

Several other measures related to signal detection theory were considered for inclusion in this chapter; including receiver operating characteristic area under the curve (ROC), area under a precision-recall curve (PR) and average precision (AP). Preliminary studies showed that ROC was unreliable in settings where class sizes were unequal. However, AP and PR had similar performance to max-info. For brevity I only include max-info here, as it has been used in published research and is relatively intuitive to understand.

### 3.3.2 Datasets

In order to identify general principles relating to interpretable units in neural networks, three datasets were selected to use for these simulations (See Figure 3.1 for dataset details). These include one small dataset and two medium datasets of varying difficulties. The aim was that these different tasks would result in a wide variety of representations to evaluate the performance of the selectivity measures on.

For all datasets, the output data was in the form of one-hot categorical codes (e.g., one output unit for each class). During training I used separate training and validation sets for all datasets. The validation set is not used to train the model (e.g., weights do not update after validation items are presented). Instead, the model accuracy and loss are calculated on the validation set, to ensure that learning on the training data generalizes to the validation data. This can prevent overfitting (e.g., 'memorising' the training set rather than learning generalizable features). Good performance on the validation set is suggests that the model will perform well on a test set.

**Iris** The iris dataset (Fisher and Marshall, 1936) consists of 150 items from three classes (e.g., species) of Iris flower (50 items per class). Inputs are four features describing the flower which correspond to the length and width of the petals and sepal measured in centimeters. Features were in the range .1 to 7.9. Iris is a small dataset, with few classes, and it is a relatively easy task to solve. It was chosen because models trained on this dataset in preliminary studies contained many highly selective units. This was thought to be valuable here for evaluating the measures' performance in this context. During training 30 items were set aside to use as a validation set. However, all 150 items were used as the test set.

**MNIST** The MNIST dataset (LeCun et al., 1998) consists of 60000 training images of hand-written digits (0 to 9) and 10000 test images. The training set was split to give 48000 training images and 12000 validation images. Images are greyscale, consisting of 28 x 28 pixels with values are in the range 0 to 255. The MNIST dataset (LeCun et al., 1998) was chosen as it was a popular benchmark test in machine learning for many years (e.g., Hinton et al., 2015; Samek et al., 2016; Nguyen et al., 2015). It also has been used as a measuring selectivity (e.g., Morcos et al., 2018).

**Fashion** The fashion dataset consists of images of ten types of clothing (e.g., dress, shirt, sandal etc., Xiao et al., 2017). It was specifically designed as a replacement for MNIST, with the fashion dataset being a more difficult task and more representative of real-world computer vision tasks. As with MNIST, these are 28 x 28 greyscale images, with values from 0 to 255. There are 48000 training images, 12000 validation images and 10000 test images.

| | Input | | Items | | |
|---|---|---|---|---|---|
| Dataset | features | Classes | Train | Validation | Test |
| Iris | 4 | 3 | 120 | 30 | 0 |
| MNIST | 784 | 10 | 48000 | 12000 | 10000 |
| Fashion | 784 | 10 | 48000 | 12000 | 10000 |

Table 3.1: **Dataset details.** Features refers to the number of input channels. For MNIST and fashion, items in the test set are not included in the training or validation sets. For iris, there is no test set, so all training and validation items were used for testing.

### 3.3.3 Models

The simulations were run on the Keras neural network API (Chollet and Others, 2015) with the Tensorflow library (Abadi et al., 2016) running on single Nvidia Titan X Pascal GPU. I have chosen to keep models and training simple in this experiment, as such I do not use drop-out, batch-normalization, convolutions or, data-augmentation. The goal is not to achieve state of the art performance, but to provide examples of units with a range of representations (e.g. selectivity scores). This experiment used three types of multi-layer perceptrons (MLPs, or 'fully-connected' models), each in four sizes (e.g., 20, 40, 200 or 400 hidden units). The three model types were single-layer models with sigmoid units (Sig1); single-layer models with relu units (Relu1); and four-layer models with relu units (Relu4). The four-layer models have an equal number of units at each layer (i.e., 5, 10, 50 or 100 units per layer). I will be able to asses the impact of activation function by comparing Sig1 and Relu1 models. The impact of the number of hidden layers will be shown in comparison of Relu1 and Relu4 models. Relu units may have more sparse representations than sigmoids (e.g., higher proportion of units with zero activation in response to a particular stimulus; Glorot et al., 2011). All together there were 48 conditions (3 x datasets, 3 x MLP type, 4 x model size). Each condition was run 3 times, from different initializations, giving a total of 108 trained models.

### 3.3.4 Training

Models were trained with categorical cross-entropy (softmax) loss function and the adaptive moment estimation (Adam) optimizer (Kingma and Ba, 2014) with an initial learning rate of .001. Weights were updated after each item (e.g., not mini-batches). All network weights were initialized using the 'He normal' initialization (He et al., 2015). Models were set to train for a maximum of 1000 epochs. There was no explicit regularization, although early stopping was used, a form of implicit regularization which can prevent over-fitting to the training data (Goodfellow et al., 2016). Training and validation loss were calculated at the end of each epoch. There were two early stopping rules: training stopped if the training loss was less than .01; or if the validation loss did not improve by $>$ .001 for 20 epochs. Models were saved after each epoch where the loss decreased, as such if early stopping is activated at epoch 100 (due not no improvement in loss for 20 epochs), then the saved model would be the one saved at the end of epoch 80.

Following training, the model accuracy on the test set was calculated. Models were only included in the following analysis if the accuracy was above chance (e.g., above .34 for iris, or .11 for MNIST and fashion).

### 3.3.5 Recording the activation of hidden units

For models where the accuracy was above chance, the process of recording hidden unit activations was as follows. First the model was presented with all items in the test set. After the test set has been presented once, all incorrect items were removed, as I am interested in the representations that support the model to succeed in the task. At this point the per-class accuracy was checked to make sure that there were at least two correct items per class, as at least two items are required to show that a unit is class-selective rather than item-selective. The test set, consisting of correctly classified items only, was presented to the model again, and the activation of each unit was recorded in response to each item. For sigmoid units, the original activation values were used for selectivity analysis, which are in the range 0 to 1. For relu units, which can take values from 0 to $\infty$, the activations were normalised to the range 0-1. This was done to make comparison of the visualizations of units easier and has no effect on the selectivity scores.

### 3.3.6 Comparing groups

The Kruskal-Wallis test by ranks was used to compare results between different groups (Kruskal and Wallis, 1952). This is a non-parametric one-way analysis of variance used to determine whether samples are drawn from the same underlying distribution. If the test is significant, post-hoc Dunn's tests, with a Bonferroni adjusted *p*-value were used to identify significant differences in the median selectivity scores for the groups (Dunn, 1964).

## 3.4 Results

Of the 108 simulations, 6 did not have an accuracy above chance and were not included in the analysis. A further 24 simulations had accuracy above chance, but they had poor accuracy on one or more classes (e.g., fewer than two correct), which is counted as failing to learn the task. Once these simulations had been removed there were 78 simulations remaining. Once dead relus (e.g., units with zero activation for all items) had been removed from the 78 models there were 9547 units in the analysis. Figure 3.3 shows a plot of the distributions of selectivity scores for these 9547 units.

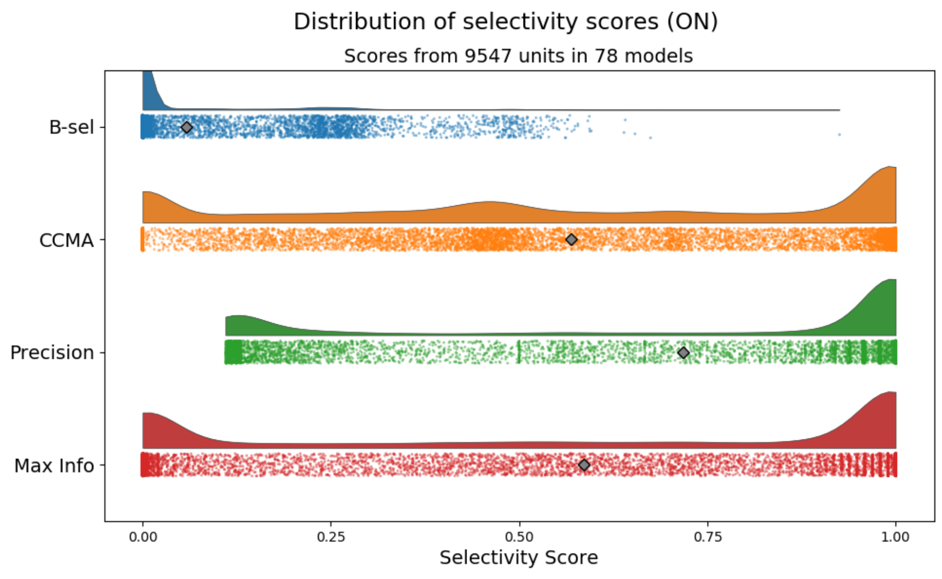


Figure 3.3: **Distribution of selectivity scores across 78 simulations.** Selectivity measures shown on the $y$-axis, selectivity scores are shown on the $x$-axis. A scatter-plot shows the selectivity score for each unit, with the distribution plotted above. Mean values are marked with a diamond. All plots have the same maximum width.

For B-sel the distribution was logarithmic, with a peak at zero, which accounted for 70% of all B-sel scores. The highest B-sel score was .93. For CCMA the distribution was multi-modal with peaks at each end and near the middle. For precision and max-info the distributions were bi-modal. For precision there were no units with scores below .11. This is because the lowest possible precision scores were for units where there were an equal number of items from each class above the threshold, which would give a score of .1 for MNIST and fashion and .33 for iris. None of the measures have a normal distribution. This has some relevance for the types of analysis that are appropriate on unit selectivity scores (e.g., non-parametric tests).

### 3.4.1 Datasets

For details of the simulations grouped by dataset (e.g., averaged across model type and size) see table 3.2. The 'Models' column shows the number of correctly trained model for each dataset. The selectivity of the class with the highest selectivity is reported for each unit.

| Dataset | Models | Mean Epochs | Test Acc | B-sel | CCMA | Prec-ision | Max Info |
|---|---|---|---|---|---|---|---|
| Iris | 36 | 167 | .98 | .27 | .76 | .98 | .96 |
| MNIST | 28 | 23 | .86 | .01 | .81 | .78 | .68 |
| Fashion | 14 | 49 | .77 | .01 | .70 | .72 | .59 |
| Total | 78 | 94 | .90 | .13 | .77 | .86 | .79 |

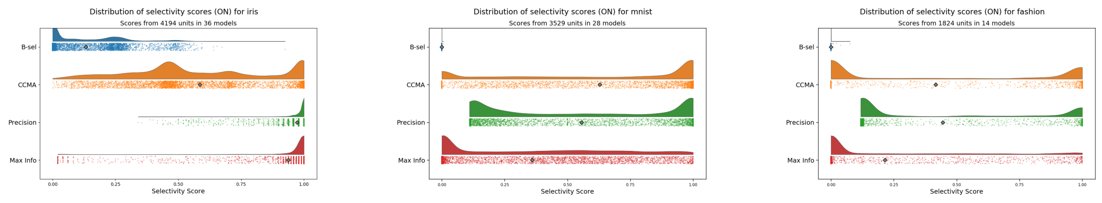
Table 3.2: **Selectivity results by dataset.** The Models column shows the number of trained models (with accuracy above chance) in each condition. All other values are the average across units in these models. Results are for ON selectivity only.

There were 14 successfully trained models on the Fashion dataset, 28 for MNIST and 36 for Iris. This is consistent with my expectation that iris was the easiest task and fashion was the hardest. Similarly, accuracy was highest for Iris (mean = .98), followed by MNIST (mean = .86) with Fashion having the lowest mean accuracy (mean = .77). These accuracy levels are good, but far below the best performing models on these datasets (e.g., LeCun et al., 1998), although this is unsurprising as my models were small and use no convolutions or regularization. MNIST trained the fastest (mean = 23 epochs), followed by Fashion (mean = 49 epochs) with iris taking the longest (mean = 167 epochs). This was somewhat surprising as Iris was the smallest dataset in terms of number of items and number of classes. This could be because the training set was too small to adequately cover the space of solutions needed for the loss to plateau on the validation set. The four selectivity measures give different impressions of the levels of selectivity for the three datasets (see table 3.2).

A Kruskall-Wallis test found a significant effect of dataset on B-sel (H(2) = 4830.63, $p$ < .001). Dunn's tests with a Bonferroni adjusted alpha level of .016 per test found significant difference between iris (median = 0.07) and MNIST (median = 0.0, $p$ < .001); iris and fashion (median = 0.0, $p$ < .001); but not MNIST and fashion ($p$ = 1.0). For CCMA there was also an effect of dataset on selectivity (H(2) = 340.62, $p$ < .001). Post-hoc tests found that iris (median = 0.52), MNIST (median = 0.79) and fashion (median = 0.08) were all significantly different to each other (all $p$s < .001). There was also an effect of dataset on precision scores (H(2) = 3761.66, $p$

< .001), with significant differences between iris (median = 1.0), MNIST (median = 0.55) and fashion (median = 0.15, all *ps* < .001). The pattern was the same for max-info, with an effect of dataset (H(2)=6236.44, p<.001) and significant differences between each dataset (iris median = 1.0, MNIST median = 0.32, fashion median = 0.001, all *ps* <.001).

Figure 3.4 shows the distribution of selectivity values for each dataset. For B-sel the majority of scores greater than zero come from models trained on the iris dataset, with very few for MNIST of fashion. For CCMA, precision and max-info, scores were positively skewed for models trained on iris, and scores were negatively skewed for fashion. For MNIST scores were positively skewed for CCMA, negatively skewed for max-info and for precision there were similar numbers of units with high or low selectivity.



a) Iris b) MNIST c) Fashion

Figure 3.4: **Distributions of selectivity scores by dataset** Selectivity measures shown on the $y$-axis, selectivity scores are shown on the $x$-axis. A scatter-plot shows the selectivity score for each unit, with the distribution plotted above. Mean values are marked with a diamond. All plots have the same maximum width.

### 3.4.2 Model-type

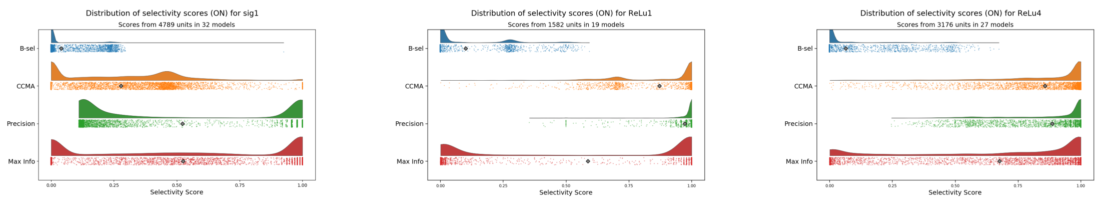
| Model Type | Models | Mean Epochs | Test Acc | B-sel | CCMA | Prec-ision | Max Info |
|---|---|---|---|---|---|---|---|
| Sig1 | 32 | 113 | .91 | .11 | .64 | .74 | .77 |
| Relu1 | 19 | 134 | .91 | .21 | .91 | .99 | .82 |
| Relu4 | 27 | 44 | .88 | .09 | .83 | .91 | .80 |

Table 3.3: **Results grouped by model type**. The Models column shows the number of trained models (with accuracy above chance) in each condition. All other values are the average across units in these models. Results are for ON selectivity only.

Table 3.3 shows the results grouped by model-type (e.g., averaged across dataset and model size). After removing models with low overall accuracy or low accuracy on particular classes, there were 32 Sig1 models, 19 Relu1 models and 27 Relu4 models in the selectivity analysis. This suggests that models with relu units struggled in these tasks more than models with sigmoid units. This might stem from the dying relu problem, where all items in a unit have an activation of zero, and as such there was no gradient to learn from. Relu4 models trained most quickly (mean = 44 epochs), yet had the lowest accuracy (mean = .88). Relu1 models took longest to train (mean = 134 epochs), followed by Sig1 models (mean = 113 epochs). Both Relu1 and Sig1 models had a mean accuracy of .91.

All measures had the highest selectivity in Relu1 models. For B-sel a Kruskall-Wallis test found a significant effect of model type (H(2) = 155.68, $p < .001$). Dunn's test with a Bonferroni adjusted alpha level of .016 per test showed that the selectivity was significantly higher for Relu1 models (median = 0.0) than for Sig1 (median = 0.0) or Relu4 models (median = 0.0, all $p$s < .001). There was no significant difference between Sig1 and Relu4 models ($p$ = 0.03). There were also significant differences in the CCMA of the three models types (H(2) = 5812.15, $p < .001$). Dunn's test showed that the median values for Sig1 (median = 0.28), Relu1 (median = 0.99) and Relu4 models (median = 0.96) were all significantly different (all$p$s < .001). Similarly, Kruskall-Wallis found differences in precision scores (H(2) = 2064.75, $p < .001$). All pairwise comparisons found differences for Sig1 (median = 0.29), Relu1 (median = 1.0) and Relu4 (median = 0.98, all $p$s < .001). For max-info, Kruskall-Wallis found a significant effect of model type (H(2) = 277.25, $p$ < .001). Median scores were significantly different between Sig1 (median = 0.55) and Relu1 (median = 0.94, p < .001) and between Sig1 and Relu4 (median = 0.86, p < .001); but not between Relu1 and Relu4 ($p$ = 0.87).

Figure 3.5 shows the distributions for each model type. For B-sel there were fewer units with a score above .2 in Sig1 models than in Relu1 or Relu4 models. For CCMA and precision scores were negatively skewed for Sig1 models and positively skewed for Relu1 and Relu4 models. This shows that selectivity tends to be higher for relu units than sigmoids. For all four measures there were more units with moderate selectivity in Relu4 models than in Relu1 models. In other words, there was a greater range of selectivity in deep relu models than shallow relu models.



a) Sig1 b) Relu1 c) Relu4

Figure 3.5: **Distributions of selectivity scores by model type** Selectivity measures shown on the $y$-axis, selectivity scores are shown on the $x$-axis. A scatter-plot shows the selectivity score for each unit, with the distribution plotted above. Mean values are marked with a diamond. All plots have the same maximum width.

### 3.4.3 Number of Hidden Units

Table 3.4 gives details of the simulations grouped by the number of hidden units (e.g., averaged across dataset and model-type). Low accuracy meant 13 models with 20 units were not included in the analysis, as were ten models with 40 units, three with 200 units and 4 with 400 units.

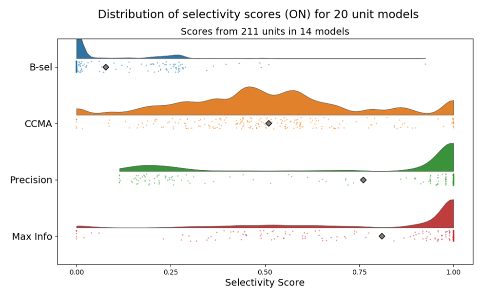
Time to train decreased as the number of hidden units increased, with the average number of epochs at 180, 139, 57 and 47 for models with 20, 40 200 and 400 units respectively. This shows that larger models were more successful at solving these tasks than small models. However, accuracy was similar for all sizes, the lowest was for models with 400 units (mean = .89), then 200 units (mean = .9) with slightly higher accuracy for models with 20 or 40 units (mean = .91). The different selectivity measures suggest very different effects of model size on selectivity. For B-sel and max-info, average selectivity was higher in smaller models than in large models, whilst for CCMA and precision there was no clear effect of the number of hidden units on mean selectivity.

Kruskall-Wallis found a main effect of model size on selectivity (H(5) = 30.93, $p < .001$). Dunn's tests with a Bonferroni adjusted alpha level of 0.008 per test found significant differences between models with 20 units (median = 0.0) and 200 units (median = 0.0, $p < .001$); 20 units and 400 units (median = 0.0, $p < .001$); and 40 units (median = 0.0) and 200 units ($p = 0.004$). All other pairwise comparisons were not significant (all *p*s > .08). Max-info had a main effect of model size (H(5) = 101.44, $p < .001$); with no difference between the two smallest sizes (20 median = 0.98, 40 median = 0.94, $p = 0.03$); or two largest sizes (200 median = 0.74, 400 median = 0.68, $p = 0.17$). All other comparisons were significantly different (all *p*s < .001). Kruskall-Wallis tests found no effect of model size for CCMA (H(5) = 6.82, $p = 0.07$) or precision (H(5) = 6.83, $p = 0.08$).

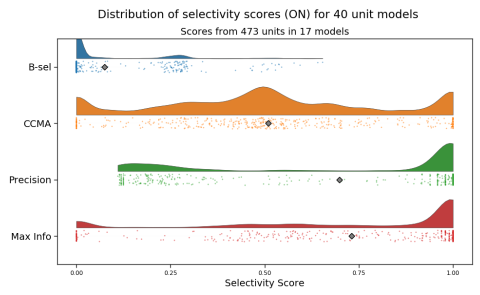
| Units | Models | Mean Epochs | Test Acc | B-sel | CCMA | Prec-ision | Max Info |
|---|---|---|---|---|---|---|---|
| 20 | 14 | 180 | .91 | .15 | .72 | .88 | .90 |
| 40 | 17 | 139 | .91 | .15 | .73 | .85 | .85 |
| 200 | 24 | 57 | .90 | .10 | .79 | .86 | .74 |
| 400 | 23 | 47 | .89 | .12 | .80 | .86 | .72 |

Table 3.4: **Results grouped by number of units**. The Models column shows the number of trained models (with accuracy above chance) in each condition. All other values are the average across units in these models. Results are for ON selectivity only.

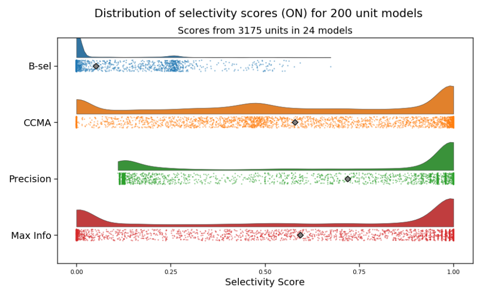
Figure 3.6 shows the distribution of selectivity score by model size. For B-sel the maximum scores decrease as the number of units increases with maximum scores of .93, .65, .67 and .64 for models with 20, 40, 200 and 400 units respectively. CCMA scores were normally distributed in small models, and the number of units with very high or very low selectivity increased with model size. For precision and max-info scores are positively skewed in small models with an increasing number of units with low selectivity as the number of units increases.



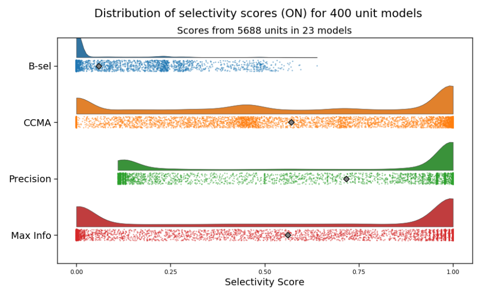
a) 20 units



b) 40 units



c) 200 units



d) 400 units

Figure 3.6: **Distributions of selectivity scores by model size** Selectivity measures shown on the $y$-axis, selectivity scores are shown on the $x$-axis. A scatter-plot shows the selectivity score for each unit, with the distribution plotted above. Mean values are marked with a diamond. All plots have the same maximum width.

### 3.4.4 Selectivity-per-layer

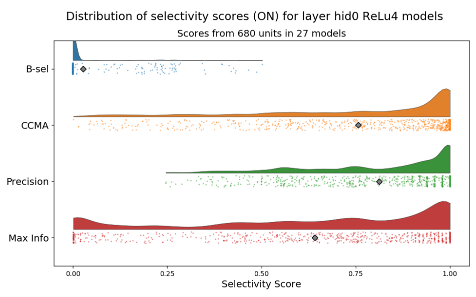
Table 3.5 shows the selectivity per-layer in Relu4 models (averaged across dataset and model size). For all measures there was lower selectivity in early layers and higher selectivity in later layers. This is consistent with previous research showing higher class selectivity in later layers (e.g., appendix A2 in Morcos et al., 2018) or a greater number of high-level 'detectors' in later layers (Zhou et al., 2017).

For B-sel, a Kruskall-Wallis test found a main effect of layer depth (H(5) = 150.78, $p$ < .001); post-hoc Dunn's test with a Bonferroni corrected alpha of .008 found significant differences between the median selectivity of each layer (all $p$s < .002). The same was also true of precision which had a main effect of layer depth (H(5) = 340.78, $p$ < .001) and significant differences between the median selectivity of each layer (all $p$s < .001). For CCMA Kruskall-Wallis found an effect of layer depth (H(5) = 157.67, $p$ < .001). Post-hoc Dunn's tests found that layer 1 (median = .87) was significantly different to layers 2 (median = 0.97), 3 (median = 0.98) and 4 (median = 0.98, all $p$s < .001). There median CCMA scores were not different between layers 2, 3 and 4 (all $p$s > .08). Max-info has a main effect of layer depth (H(5) = 96.22, $p$ < .001). There was no difference in the medians for layers 1 (median = 0.73) and 2 (median = 0.82, $p$ = 0.03); or layers 2 and 3 (median = 0.87, p= 0.02). All other layer medians were significantly different (all $p$s < .001).

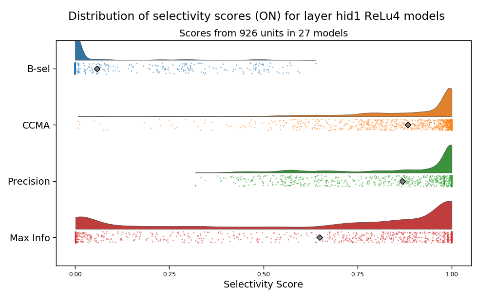
| Layer | Units | B-sel | CCMA | Prec-ision | Max Info |
|---|---|---|---|---|---|
| 1 | 680 | .03 | .76 | .81 | .64 |
| 2 | 926 | .06 | .88 | .87 | .65 |
| 3 | 914 | .07 | .89 | .91 | .68 |
| 4 | 656 | .10 | .89 | .95 | .74 |

Table 3.5: **Selectivity-per-layer for Relu4 models**. The Units column shows the number of units per layer, differences reflect the number of dead units per layer. All other values are the average selectivity of units in each layer. Results are for ON selectivity only.

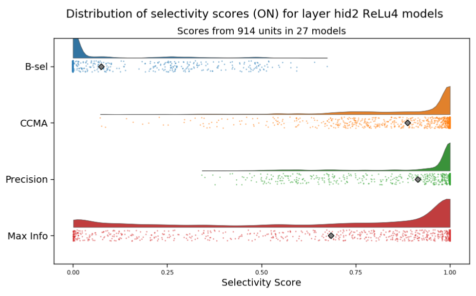
The distribution of selectivity scores per-layer is shown in figure 3.7. Units become more class selectivity an layers go from low to higher. This is consistent with previous research which showed higher selectivity in later layers (Zhou et al., 2017; Morcos et al., 2018). For CCMA and precision there were very few units with low selectivity in layer 4, but for max-info 17% of units in layer 4 had a score < .1.



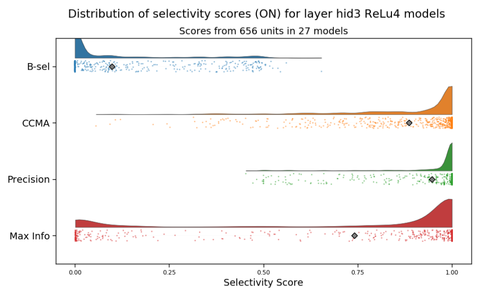
a) layer 1



b) layer 2



c) layer 3



d) layer 4

Figure 3.7: **Distributions of selectivity scores by layer in Relu4 models** Selectivity measures shown on the $y$-axis, selectivity scores are shown on the $x$-axis. A scatter-plot shows the selectivity score for each unit, with the distribution plotted above. Mean values are marked with a diamond. All plots have the same maximum width.

### 3.4.5 Interim summary

The results so far show that you get different effects of dataset, model-type and model-size depending on which selectivity measure you use. The four measures did not have the highest selectivity for the same dataset, although they did all have lowest or joint lowest for fashion. All models had higher selectivity for Relu1 models than Sig1 models, suggesting that there is an effect of activation function on selectivity with higher selectivity for relu than sigmoids. However, the measures did not all have significant differences between Relu1 and Relu4 models. Similarly the four measures did not all find an effect of model size. There was an effect of layer depth in Relu4 insofar as all measures had higher selectivity for the fourth layer than the first layer. These results shows that the effects of dataset, model type and model size on selectivity are dependent on the selectivity measure used.

### 3.4.6 Discrete measures of selectivity

The results for selectivity per dataset are presented as the averages of continuous measures. However, several authors report the selectivity of models as a count of the number of units with selectivity above a given threshold (e.g., Bowers et al., 2014; Vankov and Bowers, 2017; Zhou et al., 2015, 2017). I therefore present the selectivity results as a count of the number of units per dataset, where selectivity exceeded some threshold.

In Bowers et al. (2014) they measure the selectivity of units in recurrent models, and suggest that units with a B-sel > .5 are localist units, implying that they are not part of a distributed representation. They do not give a reason for the threshold of .5, although for units with B-sel > .5, there is more variance between groups than within groups. However, in other studies they also suggest that units might considered to be localist if scores exceed .1, .3 or .4, suggesting that the choice of threshold is arbitrary (Bowers et al., 2016; Vankov and Bowers, 2017). In chapter 2 section 2.3.4.3, I have discussed why I do not think high selectivity is grounds to claim that a unit is localist rather than part of a distributed representation. Table 3.6 shows the number of units with B-sel > .5, labelled as the number of 'Bowers' units'.

In Zhou et al. (2015) units with a precision score > .75 are classified as 'object-detectors'. In their study precision scores were based on the ratings of human participants who suggested a feature that frequently occurred in the 60 most active items. Zhou et al. (2015) do not give justification as to why they chose the value of .75 as the threshold, suggesting that .75 is an arbitrary value. Neither do they explain why they use the term 'detector' to differentiate these units from less selective units. I have my thoughts on describing these units as detectors in section 2.3.4.3 of chapter 2. As such, table 3.6 gives the number of units with precision > .75 as the number of 'Zhou units'.

I also report a third threshold, which has not been used elsewhere in the literature. This is a threshold that marks whether one class is linearly-separable from all other classes, e.g., all items from one class have higher activation (or all items have lower activation) than all other items.

Linearly-separable units will have a max-info of 1.0, and a B-sel > 0. I use this threshold in chapter 5. Selectivity above this threshold would be the minimum requirement to satisfy claims that a unit only has high activation for one class (e.g., there is a threshold where all items from class A are on one side, and all other items are on the other). As such, selectivity above threshold might represent the minimum requirement to satisfy claims that activation is similar to that of units in localist models (or grandmother cells, Bowers et al., 2014, 2016). However, as discussed in 2, I do not think that there is any level of selectivity which signifies that a unit is localist. The number of units with a B-sel > 0 is reported as the number of 'linearly-separable' units in table 3.6.

| Dataset | Models | Units | Zhou units | | Linearly-separable | | Bowers' units | |
|---|---|---|---|---|---|---|---|---|
| | | | count | prop | count | prop | count | prop |
| Iris | 36 | 4194 | 4048 | .97 | 2800 | .67 | 99 | .02 |
| MNIST | 28 | 3529 | 1403 | .40 | 6 | < .01 | 0 | 0 |
| Fashion | 14 | 1824 | 590 | .32 | 6 | < .01 | 0 | 0 |

Table 3.6: **Count of Units above thresholds** The models column shows the number of trained models (with accuracy above chance) in each condition, the units column shown the number of units in these models. Zhou-units are units with a precision > .75; Linearly separable units are units with a max-info score of 1.0 (equivalent to a B-sel > 0); Bowers-units units are have a B-sel > .5. The count columns are the number of units of each type, the prop columns give the count as a proportion of all units in models trained on that data. Results are for ON selectivity only.

Table 3.6 shows a count of the number of Zhou units, linearly-separable units and Bowers' units in each of the three datasets. All three type of unit occur most frequently in models trained on the iris dataset. The number of Zhou units was higher for MNIST than for fashion, but the number of linearly-separable and Bowers units was equal for these two datasets. Is there any justification for using these discrete measure of selectivity? The values for the thresholds are arbitrary, or at least, no theoretical reason was given in Zhou et al. (2015) or Bowers et al. (2014). I am not aware of any studies showing that there is any qualitative difference between units above these thresholds and units below. The distributions of selectivity scores shown in figures 3.3 and 3.4 do not suggest that there is anything significant about these thresholds: e.g., there is no sign of one population of 'detector's (Zhou units) with precision scores above .75, and a separate population of non-detectors with precision below .75. As such, I will treat selectivity as a continuous, rather than discrete measure.

## 3.5 Selectivity Scores for a Sample of Units

The following section will discuss the selectivity measures individually, in more depth to explore why they score units as they do, and what drives the differences between the measures. It is useful to look at visualizations of units to understand why the measures score units as they do. It is not practical to view all 9547 analysed units, so nine units have been chosen to highlight edge-cases or situations where the measures provide divergent estimates of selectivity. The first three units all have one class that is more active than the others, as such I expect them to have high selectivity for this class. The second three units have multiple classes with intermediate levels of activation, meaning selectivity will be lower than the first three, and there may be disagreement between measures as to which class is most selective. In the last three units, activations are binary-like, items either have maximum or no activation, but not intermediate activation. Units h and i are examples of saturated sigmoids, meaning there is little gradient for the model to learn from. This is due to weights being too large and is one reason why relus are often favoured over sigmoids. For visualizations of a sample of units see figure 3.8, for their selectivity scores see table 3.7. Class numbers start from 0 at the top of each plot. For example, when discussing iris which has three classes they will be classes 0, 1 and 2.

### 3.5.1 B-sel

B-sel gives the minimal distance between classes where they are linearly-separable, or a score of zero if no class is linearly separable. Only 30% of units had a B-sel > 0, which includes three of these nine units (units a, b and c). Unit a has the highest B-sel of all units in this study. In units a and b, the distance between the selective class and other classes is shown as a green region. These units both have a B-sel > .5, meaning they would be considered to be 'localist' units according to Bowers et al. (2014). For unit c, class 1 is linearly separable from other classes, but by a very small amount, meaning that the green area between classes is not clearly visible. Units a, b and c represent the range of B-sel scores from close to 1.0 in unit a, to close to zero in unit c. Units d to i all have a score of zero as no class is linearly-separable. This is a floor effect, as 70% of units had this score, and they do not all have equally low selectivity. For units a, b and c, B-sel is in agreement with the other three measures regarding the most selective class, and the other measures all score these units with 1.0, indicating that these are indeed highly selective units. B-sel was designed with the most selective units in mind, and whilst it does as excellent job in describing the level of selectivity for highly selective units, these floor effects limit the usefulness of B-sel as a selectivity measure.

### 3.5.2 CCMA

CCMA gives the mean of one class relative to the mean of the other classes. Figure 3.3 shows that CCMA scores were fairly well distributed, with some bias towards higher scores. Units a, b, c and g all have a CCMA of 1.0, which reflects ceiling effects as these units are not all maximally selective. Unit a has a CCMA of .999 which is rounded up, but units b, c and g do in fact score 1.0. This is because there are only active items from one class. CCMA is calculated as

$$\frac{\text{mean(class } A\text{) - mean(not } A\text{)}}{\text{mean(class } A\text{) + mean(not } A\text{)}}$$

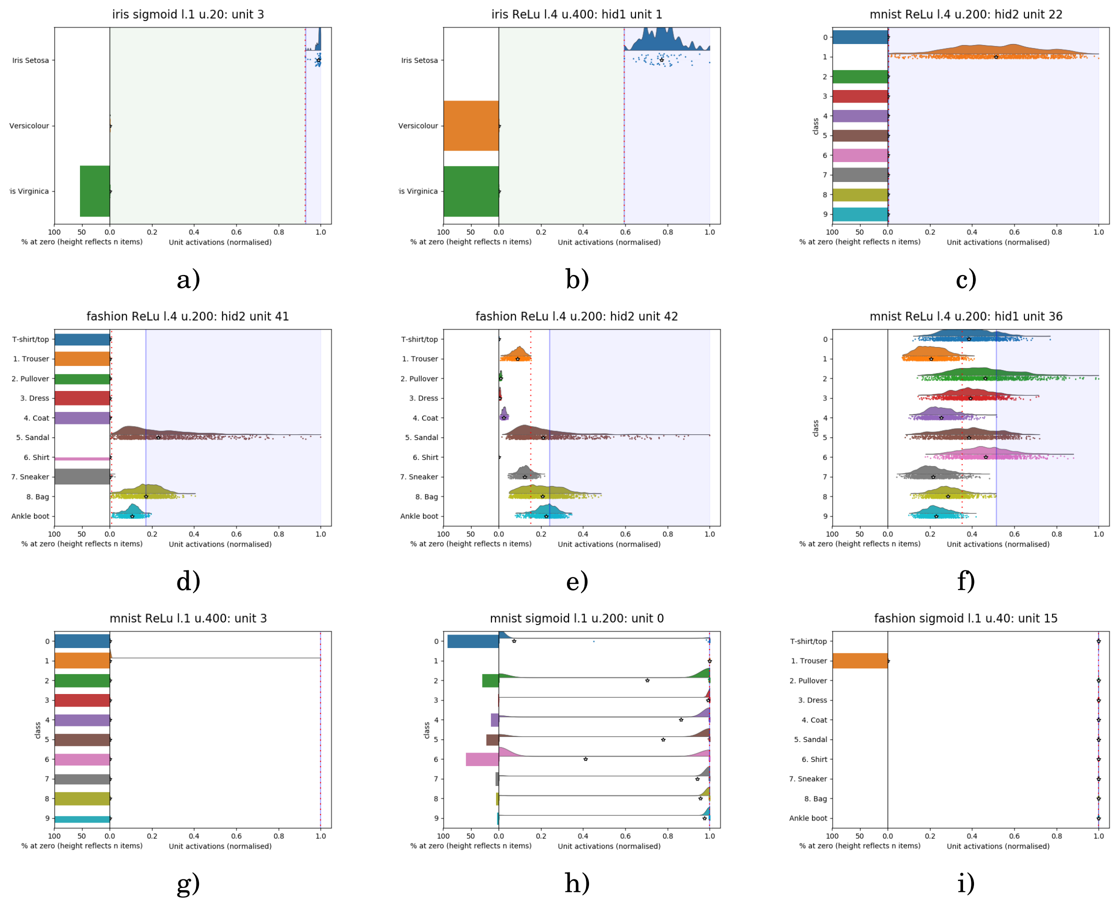


Figure 3.8: **Raincloud plots** displaying normalized activations on the $x$-axis, classes are shown on the $y$-axis. For each class a scatter plot shows active items, a dark marker shows the mean. A probability density function covers the range of active items. The proportion of each class with zero activation is shown on the left. Items considered by precision are in the blue region. A red dashed line indicates the max-info threshold. For units with B-sel > 0, the minimal distance separating the selective class is coloured green.

| Unit | Dataset | Model-Type | Size | Layer | B-sel | CCMA | Prec-ision | Max Info |
|---|---|---|---|---|---|---|---|---|
| a | Iris | Sig1 | 20 | 1 | .93 (0) | 1.0 (0) | 1.0 (0) | 1.0 (0) |
| b | Iris | Relu4 | 400 | 2 | .60 (0) | 1.0 (0) | 1.0 (0) | 1.0 (0) |
| c | MNIST | Relu4 | 200 | 3 | < .01 (1) | 1.0 (1) | 1.0 (1) | 1.0 (1) |
| d | Fashion | Relu4 | 200 | 2 | 0 | .71 (5) | .53 (8) | .74 (9) |
| e | Fashion | Relu4 | 200 | 2 | 0 | .43 (9) | .38 (9) | .75 (9) |
| f | MNIST | Relu4 | 200 | 2 | 0 | .20 (6) | .32 (2) | .54 (6) |
| g | MNIST | Relu1 | 400 | 1 | 0 | 1.0 (1) | 1.0 (1) | < .01 (1) |
| h | MNIST | Sig1 | 200 | 1 | 0 | .15 (1) | .15 (1) | .26 (1) |
| i | Fashion | Sig1 | 40 | 1 | 0 | .07 (8) | .14 (9) | .13 (8) |

Table 3.7: **Selectivity scores for the 9 sample units**. All selectivity score refer to the highest scoring class, shown in parenthesis. For units with a score of 0, no class has the highest selectivity.

When there are only active items from one class, the not-A mean is zero, which always gives a score of 1.0. Put differently, a score of 1.0 indicates high specificity (e.g., only active items from one class), regardless of the sensitivity (e.g., the proportion of active items from that class). The score of 1.0 suggests that units c and g are both equally selective for class 1, but I would argue that unit c is more selective for class 1. (This point has also been discussed with hypothetical units in Gale et al., 2019, 2020).

How selectivity measures handle sparse units is important: 20% of units had zero activation for over 90% of items, and 11% had zero activation for 99% of items. An implication of the fact that CCMA gives a high score to classes with few active items relates to Leavitt and Morcos (2020b). They inserted a CCMA term into a regularizer using which allowed them to manipulate the mean class selectivity as they trained models. They found that reducing selectivity has little impact on performance, but increasing selectivity was harmful to performance. Unit g illustrates a potential limitation of this approach: models may have been too reliant on sparse units, with high CCMA, which impaired the performance. Leavitt and Morcos (2020b) do not visualise any units so it is not clear whether this was an issue. Measures that compare sample means are often considered to be parametric tests which should only be used where assumptions about the distributions are met (e.g., data is has a normal distribution). CCMA compares sample means and as such, it might unreliable when the activations were not normally distributed, as is certainly the case for sparse units.

CCMA disagreed with precision and max-info regarding the most selective class in units d, and with precision for units f and i. For units d and f, CCMA chooses the class with the highest mean. For unit i, nine classes have a mean activation of 1.0. Differences in CCMA scores for these classes are driven by class size, with class 8 being the largest class. This means that there are fewest items *not* in class 8, so the not-A mean for class 8 is the smallest, leading to the highest CCMA for class 8.

### 3.5.3 Precision

Precision gives the proportion of the $n$ most active items that are from a given class, where $n$ equals the size of the class being analysed or the number of active items (if less than the class size). In figure 3.8, the items considered by precision are in the blue region. However, for units g, h and i, this blue region is at 1.0. For unit g this is because there is only a single active item, and for h and i the number of items with an activation of 1.0 is greater than the class size. Items below the blue region do not impact on precision scores.

Precision suffers from ceiling effects, with units a, b, c and g all scoring 1.0. In units a, b and c the score of 1.0 indicates high sensitivity (e.g., all items from the given class are above the threshold) and specificity (e.g., no items from other classes are above the threshold). However, for unit g, the high precision score reflects the fact the there was only one active item. As such, a high precision score indicates high specificity, but not high sensitivity.

Precision did not agree with CCMA and max-info about which class was most selective in units d, f and i. For units d and f precision selects the class with the highest proportion of items in the blue region. For unit i, there are 6820 items all with an activation of 1.0. I had not anticipated having so many items with an activation of 1.0, and had not balanced or sorted identical activations to account for this. Precision took a subset of these activations for each class, based on their order in the dataset. The number of items considered for each class was equal to the class size, which varied from 403 to 942 items. Based on these subsets, the highest precision score was for class 8.

In summary, precision is easy to interpret and in dense units the scores are a reliable indicator of class selectivity. However, for units where many items have an activation of zero or 1.0 it is a less reliable indicator of class selectivity.

### 3.5.4 Max-info

Max-informedness identifies the threshold with the best sensitivity and specificity (e.g., where most items from class A are above the threshold and most items not in class A are below the threshold). A score of 1.0 indicates that a class is linearly-separable (e.g., units a, b and c). However, I would argue that unit a is more selective than unit c, meaning that there are ceiling effects for max-info.

Max-info gives class 9 the highest score for unit d, whilst CCMA scores class 5 highest and class 8 has the highest precision. This is because the *least* active item from class 9 is *higher* than for classes 5 or 8. The best threshold for class 9 is at .01; *all* items from class 9 are above the threshold, giving a sensitivity of 1.0. There are 6714 items from classes 5 and 8 above this threshold, accounting for 26% of all items not in class 9, giving a specificity for class 9 of .74, and therefore a max-info of .74 (e.g., max-info = max(sensitivity + specificity) -1). At this threshold of .01, there are items from classes 5 and 8 below the threshold so they do not have a sensitivity of 1.0. The best threshold for class 5 and 8 are lower, at .004 or .002 respectively. However, at these thresholds their specificity decreases as other items from class 5, 8 or 7 are now above the threshold. All three classes have a max-info > .72.

For unit g where there is a single active item, max-info gives class 1 a score of .0009, far lower than the score of 1.0 for CCMA and precision. This demonstrates that max-info avoids inflating scores for sparse units.

For unit i, all items for all classes have an activation of 1.0, apart from class 1. When a threshold is set just below 1.0 there are 6820 items above the threshold. Max-info scores class 8 highest, because it is the largest class. There are fewer items *not* in class 8, compared to the number of items *not* in other classes. Class 1, which is below the threshold, therefore accounts for a larger proportion of the items *not* in class 8, than it does for other classes. As such, class 8 has the highest specificity.

Max-info avoids inflating the scores for sparse units, and as such, it is the most reliable indicator of class-selectivity.

## 3.6 On and OFF selectivity

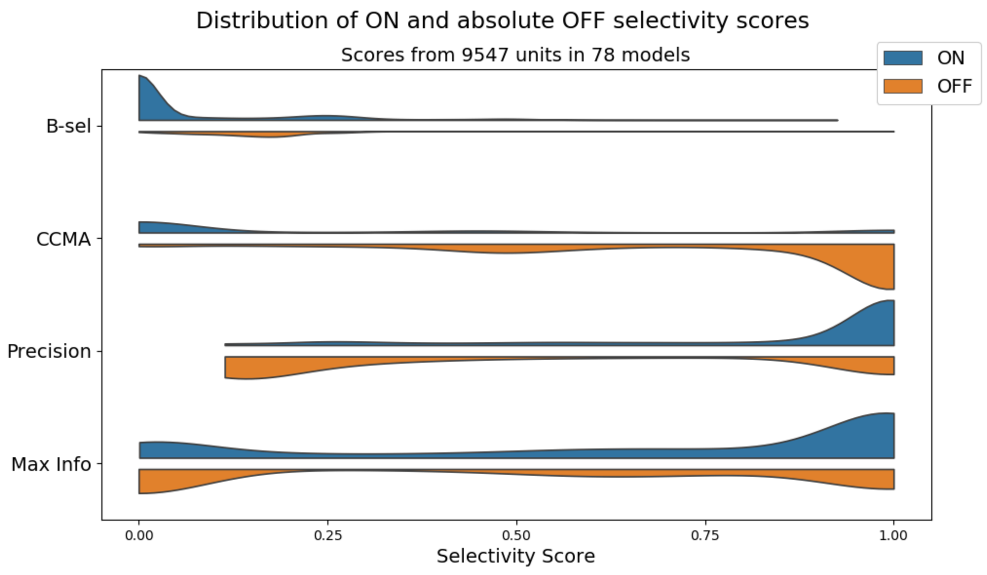


Figure 3.9: **Distribution of ON and OFF selectivity scores** across 78 simulations. Selectivity scores are shown on the $x$-axis and measures shown on the $y$-axis, with separate probability density estimates for ON (blue) and OFF (orange) selectivity. The width of the distribution reflects the number of observations.

The results so far only relate to ON selectivity: where one class has higher activation than other classes. I also tested units for OFF selectivity (e.g., one class has lower activation than other classes). I ran separate ON and OFF selectivity analysis for each unit, such that there were two scores for each class. Units are reported as being selectivity for the class with the highest of these scores. Taking the highest score of from ON and OFF selectivity analysis naturally gives us higher mean selectivity scores than when testing for ON selectivity only, as scores either increase (i.e., because there is a higher OFF selectivity score) or do not change (i.e., if the highest score if still ON selective).

Figure 3.9 shows the distributions for each measure, with ON selectivity shown in blue, and OFF selectivity in orange. The overall shape of the distributions are similar to those for ON selectivity only, but shifted slightly to the right (see figure 3.3). For B-sel there are few OFF units, although at the tail of the distribution there is an OFF unit with a B-sel of 1.0. For CCMA the majority of units are now OFF selective. For precision, many of the units with low selectivity are OFF selective, but the majority of highly selective units are still ON selective. For max-info there is a similar pattern to precision, more of the units with low selectivity are OFF than ON, but for high selectivity there are more ON than OFF units.

| | B-sel | | | CCMA | | | Precision | | | Max-info | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | OFF | Mean | Incr | OFF | Mean | Incr | OFF | Mean | Incr | OFF | Mean | Incr |
| Iris | .25 | .30 | .03 | .87 | .86 | .10 | .28 | .99 | .01 | .28 | .96 | .01 |
| MNIST | 0 | .01 | 0 | .86 | .93 | .12 | .27 | .79 | .01 | .31 | .70 | .02 |
| Fashion | .001 | .01 | .001 | .55 | .79 | .09 | .56 | .74 | .02 | .57 | .61 | .02 |
| Total | .11 | .14 | .01 | .80 | .87 | .10 | .33 | .87 | .01 | .35 | .80 | .02 |

Table 3.8: **ON & OFF selectivity by dataset** Mean gives the average selectivity when ON and OFF units are included. OFF gives the proportion of units that were selectively OFF. 'Incr' is the increase increase in selectivity when OFF units are included, compared to just ON units.

Table 3.8 gives the average selectivity for each measure when OFF units are included. It also shows the proportion of units that were selectively OFF for each measure. The 'incr' column shows the increase in selectivity scores when OFF units are included, compared to just ON units. For B-sel 11% of units were OFF, but this had a small effect of the mean selectivity, which only increased by .01. Most of this change is driven by units from model trained on iris, where 25% of units were OFF, leading to a .03 change in mean selectivity. B-sel had the lowest proportion of OFF units, which is probably because most units for B-sel had no classes that were linearly-separable, so there was no ON or OFF selectivity for these units. CCMA had the highest proportion of OFF units, with 80% of units selectively OFF. Mean CCMA was .1 higher when OFF units were included. The lowest proportion of OFF units for CCMA were for models trained on fashion, where 55% of units were OFF. A third of units were selectively OFF for precision, although this only led to a .01 increase in mean selectivity. There were around twice as many OFF units in models trained on fashion than for iris or MNIST. For max-info, 35% of units were selectively OFF leading to a .02 increase in mean selectivity. The highest proportion of OFF units was for fashion. These results show that the proportion of OFF units greatly varies depending on how selectivity is measured.

Figure 3.10 shows the the nine sample units, with their respective selectivity scores for ON and OFF selectivity shown in table 3.9. For B-sel there are only two differences compared to ON units only, units e and i. For unit e, all items from class 0 have an activation between .00003 and .0001, and the lowest activation from other classes is at 0.0003. Therefore unit e has a B-sel of .00002. At the other end of the scale is unit i, which has an OFF selectivity of 1.0. This is the only unit in this thesis with such perfect class selectivity.

For CCMA, unit b still has ON selectivity, but the other eight units are now OFF selective. Units b, c and g illustrate a limitation of applying CCMA to units with lots of activations at zero, as often occurs with relu units. This explains why 80% of units were selectively OFF for CCMA, and why mean CCMA is .1 higher when OFF units are included. When discussing CCMA for ON units I noted that units will always have a score of 1.0 if the mean of items not in class-A is zero.

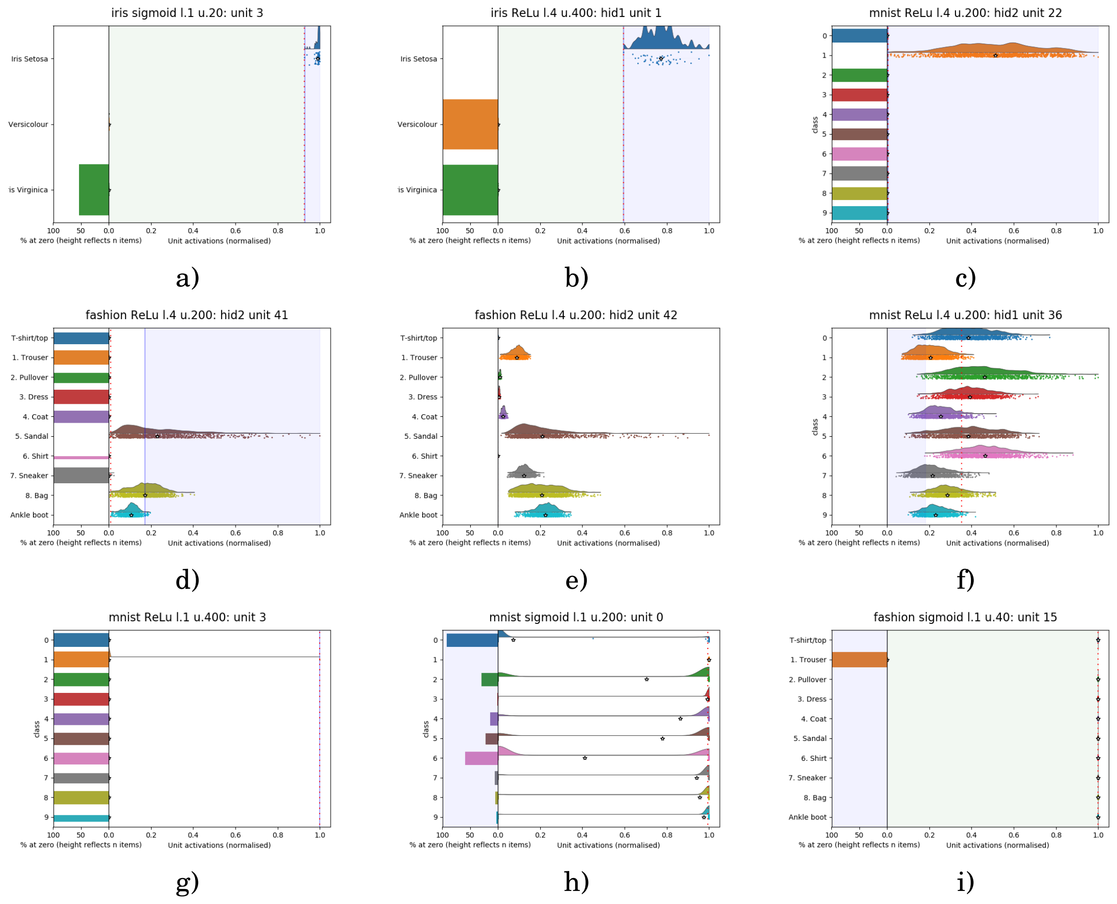


Figure 3.10: **ON & OFF Raincloud plots** $x$-axis: activation, $y$-axis: class. For each class a scatter plot shows active items, a dark marker shows the mean, probability density function covers the range of active items. The proportion of each class with zero activation is shown on the left. Items used for precision are in the blue region. A red dashed line shows the max-info threshold. For units with B-sel > 0, the distance separating the selective class is in green.

| Unit | Dataset | Model-Type | Size | Layer | B-sel | CCMA | Prec-ision | Max Info |
|---|---|---|---|---|---|---|---|---|
| a | Iris | Sig1 | 20 | 1 | .93 (0) | 1.0 (2)* | 1.0 (0) | 1.0 (0) |
| b | Iris | Relu4 | 400 | 2 | .60 (0) | 1.0 (0) | 1.0 (0) | 1.0 (0) |
| c | MNIST | Relu4 | 200 | 3 | < .01 (1) | 1.0 (0)* | 1.0 (1) | 1.0 (1) |
| d | Fashion | Relu4 | 200 | 2 | 0 | 1.0 (1)* | .53 (8) | .74 (9) |
| e | Fashion | Relu4 | 200 | 2 | < .01 (0)* | 1.0 (0)* | 1.0 (0)* | 1.0 (0)* |
| f | MNIST | Relu4 | 200 | 2 | 0 | .25 (1)* | .38 (1)* | .54 (6) |
| g | MNIST | Relu1 | 400 | 1 | 0 | 1.0 (0)* | 1.0 (1) | < .01 (1) |
| h | MNIST | Sig1 | 200 | 1 | 0 | .84 (0)* | .41 (0)* | .76 (0) |
| i | Fashion | Sig1 | 40 | 1 | 1.0 (1)* | 1.0 (1)* | 1.0 (1)* | 1.0 (1)* |

Table 3.9: **ON & OFF Selectivity for 9 sample units**. All selectivity score refer to the highest scoring class, shown in parenthesis. For units with a score of 0, no class has the highest selectivity. OFF units are marked with *

CCMA is calculated as

$$\frac{\text{mean(class } A\text{) - mean(not } A\text{)}}{\text{mean(class } A\text{) + mean(not } A\text{)}}$$

For OFF units there is a similar problem, but it extends to multiple classes. For any class where the mean activation is zero, the CCMA will be -1, regardless of whether there are other classes with lots of zero activations. For OFF selectivity the absolute value is used, giving a CCMA of 1.0. For units b, c, and g, each class has a zero in the equation, either for the class-A mean or the not-A mean, and as such, *all classes* have a selectivity of 1.0. In this case, CCMA arbitrarily selects class 0. For units e and f, where there are no items with zero activation, CCMA behaves as expected, with highest OFF selectivity for the class with the lowest mean. In general CCMA is unreliable when applied to OFF selectivity.

Four of the nine units are selectively OFF when measured with precision. For units e and f, there are no non-active items, so precision tests the least active items (highlighted in blue in figure 3.10. For unit h, precision considers all items with zero activation and has highest selectivity for class 0. For unit i, all items from class 1 have an activation of zero, and all other items are active.

For max-info there are only two units that are selectively OFF. Units c and i have a single class with no active items, which scores and MI of 1.0.

In summary, testing for OFF selectivity gives sensible results with B-sel, precision and max-info, but is unreliable for CCMA, especially where there are lots of items with zero activation. There are some units which are better described as selectively OFF than selectively ON (e.g., unit i), but for the most part the additional testing for OFF selectivity had a very small impact on the mean selectivity scores or their distribution. As such, estimates of selectivity from studies that only tested for ON selectivity are probably fairly reliable (e.g., Morcos et al., 2018; Zhou et al., 2015, 2018b, etc.).

## 3.7 Discussion

In this chapter I have presented four measures of selectivity that have been used in previous research. I applied these measures to models of differing depths, sizes, and using different activations functions. Models were trained on three different datasets. The aim was to generate a variety of sparse and dense units to test the measures. I discussed the different measures with reference to visualizations of 9 units, which allowed me to show that all measures showed either floor or ceiling effects.

High class-selectivity was defined as having high specificity (e.g., items with high activation were mostly from one class) and high sensitivity (e.g., most items from the class in question has high activation). There were several examples of such units, demonstrating that highly selective units can be found in neural networks using distributed representations. For example there was a unit with an ON B-sel score of .93, and another unit with an OFF B-sel of 1.0. These units have very high activation for all items from one class, and very low activation for all other items (or vice versa for OFF units). These units are easily interpretable with respect to one class of inputs.

B-sel indicates the minimal distance between classes where they are linearly separable. This makes B-sel a good choice for capturing the differences between highly selective units. It does not suffer from ceiling effects as the other three measures did. However, for many units there is no linearly-separable class and B-sel has a score of zero. These floor effects make B-sel unsuitable as a general measure of selectivity.

CCMA gives the class mean in relation to the mean of other classes. It is a reliable measure of selectivity for units where most items are active, but for sparse units CCMA suffers from ceiling effects making its scores harder to interpret as a measure of class-selectivity. A CCMA score of 1.0 indicates that only one class has active items, regardless of the proportion of that class which is active. Put differently, a CCMA of 1.0 indicates perfect specificity, but gives no indication of the sensitivity. These limitations of CCMA were exacerbated when testing for OFF selectivity. Overall, these weaknesses make CCMA an unreliable measure of class selectivity.

Precision is appealing in its simplicity: it gives the proportion of the most (or least) active items that are from a given class. However, its usefulness as a measure of selectivity will depend on how it implemented. I chose to use an adaptive threshold size, equal to the size of the class being analysed. This meant that I avoided the unreliable scores associated with using too small a threshold (e.g., considering just 60 items, Gale et al., 2019). The adaptable threshold allowed us to compare precision scores between very different datasets. This also meant that the number of items analysed could be different for each class in the same unit, meaning that comparison of precision scores was rarely like-for-like. However, this adaptive threshold was of little relevance for sparse units, where precision considered all active items. This led to ceiling effects and over-estimates of class selectivity, such as a score of 1.0 for a unit with a single active item. In the appendix of Leavitt and Morcos (2020b) they use a precision measure in such a way that it gives low scores for sparse units, which might have been a better indicator of class-selectivity. I also could have implemented a better strategy to deal with units where many items have an activation of 1.0. In summary, precision, as it was implemented here proved to be an unreliable measure of selectivity. However, there may be variants of precision which are more reliable.

Max-info summarises the sensitivity and specificity of each class at the optimum threshold. The fact that it reflects both the sensitivity and specificity meant that max-info gave scores that were most consistent with my definition of class-selectivity. However, there were ceiling effects, max-info scored all linearly separable units with 1.0, meaning it did not differentiate between the most selective units.

In the first part of the study I also reported the effects of dataset, model type and model size on mean selectivity. The results show general trends although the four measures were rarely unanimous regarding the effects of these factors on selectivity. The four measures all agreed that selectivity was significantly lower for fashion than for MNIST or iris. The measures also all had higher selectivity for Relu1 models than Sig1 models. The models all had higher selectivity for the last hidden layer in Relu4 models than in the first layer. For the other effects of dataset, model type, model size, the results depend on which measure of selectivity is used.

Some researchers reported a count of the number of units with selectivity above some value, rather than the mean selectivity of models (e.g., Bowers et al., 2014, 2016; Vankov and Bowers, 2017; Zhou et al., 2015, 2017). There were multiple examples of units that would be classed as 'object-detectors' or 'localist units' in these studies. However, I note that these definitions are based on arbitrary thresholds and the distributions of selectivity results do not suggest that there are separate populations of detectors and non-detectors, or localist and distributed units. In the absence of any theoretical reason to focus only on the most selective units, I shall continue with the approach of treating selectivity as a continuous property of a unit.

I compared the results for testing for classes that are selectively ON with testing for ON and OFF selectivity. Scores were naturally higher when OFF selective units were included. For CCMA 80% of units were OFF selective and the mean selectivity increased from .77 to .87. However, this was largely due to idiosyncrasies of the way CCMA treats classes with a mean activation of zero. As such, I recommend that CCMA is only used to test for ON selective units. The other three measures gave appropriate scores for OFF selectivity. There were units which were better described with OFF selectivity than ON selectivity including one textbook example where *all* class 1 items had zero activation and *all* other items had an activations of 1.0. However, I found that including OFF selectivity had little impact on the mean selectivity scores. Compared to when I only included ON units, the mean selectivity was only .01 higher for B-sel and precision; and only .02 higher for max-info. In other words, testing for OFF selective units can give a better description of some units, but it might only have a marginal impact on mean selectivity. A unit does not have to be classed as either ON or OFF. It is the outgoing weights that give a unit an excitatory or inhibitory role, and a unit may have both positive and negative outgoing weights. As such, I will focus only on ON selectivity throughout this thesis.

This chapter highlights the importance of visual inspection of unit activations to ensure that measures are performing as expected. This is especially important since the measures all suffered from floor or ceiling effects. For example, four of the 9 units discussed in detail had a score of 1.0 from at least one measure when tested for ON selectivity, or seven units when OFF selectivity was included. These units had very different patterns of activation, but without visual inspection of the activations one might have very different assumptions about how these units contribute to representing classes.

Ideally, a selectivity measures should be able to capture the full range of possible activations, without floor or ceiling effects. For example, a measure that was the sum of max-info and B-sel scores would have scores in the range 0 to 2. Where a class was *not* linearly-separable (e.g., B-sel would give a score of zero), scores would be in the range 0 to 1 (e.g., scores would be identical to max-info). Where the class was linearly-separable (e.g., max-info would score 1.0), scores would be in the range 1 to 2, and would essentially be B-sel + 1. Whilst this is not a perfect measure, does remove the main two problems with B-sel and Max-info (namely, the floor and ceiling effects respectively). However, such a measure would still be limited, as all selectivity measures are, to describing a unit's activations with reference to just one class. Selectivity analysis is a useful tool, but its role should not be exaggerated. The fact that a unit has high selectivity for one class does not mean that the unit only contributes to one class.

As discussed earlier, the findings from this chapter have been incorporated in to published work (e.g., Gale et al., 2019, 2020). In Gale et al. (2019) they compare selectivity measures on units from AlexNet (Krizhevsky et al., 2012) pre-trained on imageNet (Deng et al., 2009). They find that in this large model B-sel suffers floor effects, and that CCMA and precision can overestimate class-selectivity for some units. They also presented hypothetical units demonstrating that CCMA can give a score of 1.0 to a maximally sparse unit. The finding that maximum-informedness may act as a more reliable measure of selectivity than B-sel, CCMA and precision, informed the decision to include max-info in Gale et al. (2020). They analysed units from AlexNet with max-info and found that it was less prone to overestimate class selectivity in AlexNet. The fact that a similar pattern of results for these selectivity measures were found in AlexNet as are reported here suggests that the limitations with these measures are robust, generalizable findings.

In conclusion, I find that none of the measures are able to capture the full range of selectivity scores, with measures either showing floor or ceiling effects. The floor effects associated with B-sel limit its usefulness as a general measure of selectivity. CCMA and precision had ceiling effects and they were likely to overestimate the class-selectivity of sparse units. Max-info also had ceiling effects, but it was not influenced by sparsity as CCMA and precision were, making it the most reliable of the four measures.

# Chapter 4

# Dataset properties and selectivity

## 4.1 Introduction

When considering tasks to which a neural network can be applied, one can consider the systematicity or arbitrariness of the task's input-output mappings. For tasks with systematic mappings, items from the same class (e.g., same outputs) will tend to have similar inputs and items from different classes (e.g., different outputs) will have dissimilar inputs. For example, in many languages, the spelling and pronunciation of words is highly systematic: words with a similar spelling (e.g., input) will have similar pronunciation (e.g., output). On the other-hand, for tasks with arbitrary mappings, items with similar inputs do not necessarily have similar outputs and items with different inputs do not necessarily have different outputs. For example, when matching faces (input) to names (output), the fact the two people have faces that look similar does not suggest that they will have similar names, and people with the same name are not assumed to have similar appearance. In reality the picture is more complex, input-output mapping can be placed along a continuum with highly arbitrary at one end and highly systematic at the other. For example, the correspondence between spelling and pronunciation in written English is quite systematic but there are many words with irregular pronunciation (e.g., have, pint, etc). The mappings from faces to names is quite arbitrary, but not completely: when introduced to 'Oliver' and 'Olivia', one can often use facial appearance to predict who is who.

The similarity between items need not be explicitly present in the input data: part of the appeal of neural networks is that through abstraction they can discover structure that is only implied in the input data. For example, Hinton (1986) trained a model to discover relationships between members of two analogous families. When given the input code for the Italian family that signifies 'Emilio' and 'is uncle of...' it returns the correct output 'Alfonso'. When given the input from an English family 'Arthur' and 'is uncle of...' it correctly return 'Colin'. It is not the similarity between the codes for 'Emilio' and 'Arthur', or 'Alfonso' and 'Colin' that allowed the model to discover relationships. Instead, the model had identified useful underlying structure such as the nationality or generation of individuals. These higher-level features were not explicitly represented by the input units, but were be captured by the principle components of the model's representations (McClelland, 1994; Plaut and McClelland, 2000). In other words, similarity between items may be apparent in the similarity of their hidden layer representations (Rumelhart et al., 1986a; Goodfellow et al., 2016).

## 4.2 Arbitrariness and population-sparsity

It has been claimed that dense representations (i.e., many simultaneously active units) are well suited to representing knowledge in systematic tasks, where items with similar inputs will map to the same or similar outputs (Plaut and McClelland, 2000, 2010). Such tasks allow a rich similarity space to form, which can be exploited by the network in the form of automatic generalization of learning about one item, to other similar items. To understand how dense, overlapping representations and generalization go together, consider a relatively systematic task where images of people are mapped into different age groups (e.g, child, teenager, young adult, middle-aged etc). The images can differ along many different dimensions. In dense representations hidden-units represent directions which capture lots of the relevant variance. For example, hair-brightness, skin-wrinkliness, and posture-quality could all be predictive of age, so there might be a hidden-unit for each. Such a system could support generalization to new examples. If the model has learned to associate teenagers with activation of the hidden-unit for bright-hair, but not the units for wrinkles or good posture, then when presented with a novel image that produces a strongly activates the hidden unit for bright-hair,, but not wrinkles and good-posture; the model will infer that the image is of a teenager. Another example of generalization applies to learning new features. For example, there might be an image of a teenager which is repeatedly miss-classified as a child or young-adult. To learn to correctly label the image as a teenager, the model discovers a new distinguishing feature: dental-braces. By learning to represent dental-braces in the hidden layer, not only does the model now get this item correct, but it can use the presence of dental-braces as a useful cue of 'teenagerness' for other images.

In systematic tasks, the generalization that comes with distributed representations is beneficial to performance. Similarity in the input-pattern or hidden-layers suggests a similar output. However, in arbitrary tasks, such generalization could be harmful: there are many sources of similarity in the input which are not predictive of the output. As discussed earlier, when learning to associate faces with names, it would be a mistake to assume that people with similar names will look more alike than people with dissimilar names. A model trained to map images to identities that represented all the ways in which faces can be similar would be susceptible to catastrophic-interference. For example, in learning to associate ‘John’ with activation of the units for wrinkles and good posture units, but not colourful hair; this might interfere with a previously learned ‘John’, who has grey hair. Similarly, when presented with a new item ‘Jameelia’, who has wrinkles and good posture, the model might be assume it is ‘John’.

The question of whether arbitrary tasks could be solved by neural networks that used distributed representations, was investigated by Hinton et al. (1986). They gave models the task of mapping the spelling of 20 words to their respective meanings. The meaning of a word was defined as a pattern of activation over a set of ‘sememe’ output units. For example, given the input pattern for cat (e.g., ‘c/1’, ‘a/2’, ‘t/3’) a model should activate the relevant output units (e.g., ‘animate’, ‘fluffy’, ‘pet’ etc). They note, that for a distributed hidden layer, where each unit fires for multiple items, there would be a risk of interference. For example, some sememe units relating to the word ‘cat’ might be incorrectly activated by words with similar input patterns (e.g., ‘bat’, ‘cot’, or ‘car’). This situation could be avoided by using a localist hidden layer, e.g., separate hidden-units for ‘bat’, ‘cat’ and ‘car’ etc, each with links to the relevant orthographic input units and semantic output units. However, Hinton suggests that a hidden layer has the capacity to avoid unwanted generalization, and can use distributed representations (e.g., each hidden unit fires to many items) as long as they are relatively sparse (e.g., ‘many items’ was just a small fraction of the dataset). Hinton et al. (1986) trained a multi-layer perceptron (MLP) to map the spelling of 20 words to a unique pattern across the sememe units representing the word’s meaning. They recorded the model accuracy after lesioning each unit in turn. They found that lesioning a hidden unit led the model to fail on multiple items, suggesting that each hidden unit was involved in representing many items, and as such, the model did not use a localist coding scheme. This study demonstrates that distributed representations can solve arbitrary tasks, and suggests that they do so by using sparse, but not localist (e.g., maximally sparse) hidden layer representations.

The reason that there is less chance of catastrophic-interference with sparse representations compared to dense one, is that there is less overlap between representations (Hinton et al., 1986). If representations are dense such that the average item activates 50% of hidden-units, the chance that any particular unit is involved in representing item A and B items is .5. That means that changes to the unit as result of learning about item A, will have a .5 chance of impacting that aspect of the representation for item B. There will be very few items that do not share any hidden-units (i.e., have no overlap). For a sparse representation where the average item activates only 10% of hidden-units, there are many pairs of representations that have no overlap. For a unit involved in representing item A, there is only a .1 chance that it also fires for item B. This means that changes to a unit after learning about item A will only impact a small subset of other items.

The appropriate level of sparsity for a task is not stipulated by the modeller, but rather, the task determines the appropriate level of sparseness, which models themselves discover (Plaut and McClelland, 2000, 2010). Plaut and McClelland (2010) suggests that the principle that *a unit that fires to x will also fire for similar items*, is less strong for models trained on arbitrary tasks than systematic ones, as models learn larger weight values to overcome the effects of similarity (Plaut et al., 1996).

A compelling demonstration of the power of neural networks to solve arbitrary tasks comes from Zhang et al. (2016). They compared the performance of deep convolutional neural networks (CNNs) trained on a regular version of imageNet with those trained on arbitrary versions. In the arbitrary versions, either the input data (pixels) or output data (class labels) were shuffled. This shuffling breaks any systematicity between inputs and outputs, increasing the arbitrariness and making the task more difficult as the model must *ignore* similarity. For example, for data with shuffled labels, the models must learn that even though two items both *look* like dogs (e.g., similar inputs), they should be classified according to their new, shuffled labels (e.g., car and horse). Zhang found that although models took longer to learn the arbitrary datasets, they eventually reached accuracy > .99, thus demonstrating that neural networks can learn highly arbitrary tasks with almost perfect accuracy. Zhang et al. (2016) proposed that the models had a large enough capacity to simply 'memorise' the labels for each individual item in the dataset. The model could not identify class-members based on a small number of generalizable features as it would for a systematic task (e.g., use units dedicated for car parts to identify members of automobile classes). They suggest that the vast capacity of modern CNNs is large enough for them to learn to recognise the items individually for arbitrary tasks. Whilst Zhang et al. (2016) noted the performance of these models, they did not investigate the internal representations.

## 4.3 Arbitrariness and selectivity (lifetime-sparsity)

The claim that sparse representations are better suited for arbitrary tasks than dense ones, raises the question about the relationship between arbitrariness and class-selectivity. At first glance, the literature seems to suggest that highly selective representations are well suited for representing arbitrary relationships (e.g., Kumaran et al., 2016; Coltheart et al., 2001). However, on closer inspection, these claims may in fact relate to sparsity rather than to 'class-selectivity', which is the focus on this thesis. I shall return to this point once I have discussed the relevant claims.

Building on earlier ideas from Marr et al. (1991), the complementary-learning-systems hypothesis suggested that the brain uses dense distributed representations in the cortex to capture systematic regularities; but that the hippocampus treats episodic data as somewhat arbitrary and uses sparse representations (McClelland et al., 1995; Kumaran et al., 2016). This way, new learning in the hippocampus can not catastrophically interfere with existing knowledge in the cortex. Similar to the complementary-learning-systems hypothesis, Coltheart et al. (2001) proposed a two-systems approach to mapping the spelling of words to their pronunciation. The dual-route cascade model of reading uses a rule-based approach to name words with systematic mappings between spelling and pronunciation (e.g., hose, pose and rose; or hint, mint and tint); but a separate system with localist (e.g., maximally sparse and selective) representations to handle irregular words with a more arbitrary relationship between spelling and pronunciation (e.g., lose, pint etc). In both cases, the complementary-learning-systems hypothesis and dual route model propose that arbitrary information is handled by a more selective system than systematic information. To understand why these claims do *not* relate relate to class-selectivity, I shall first introduce some new terminology relating to a task's mappings, which can be one-to-one or many-to-one.

In a one-to-one task, each unique input pattern (e.g., item) is mapped to its own unique output. For example, mapping images of 100 people to 100 different names: each item (e.g., input pattern) is a unique identity with its own name (e.g., output pattern). For many-to-one tasks, there are several inputs that are mapped to each output. For example, mapping 100 images of ten people to their correct identity: there are ten images for each name (e.g., ten items per class). For many-to-one tasks, 'selectivity' typically refers to class-selectivity, as many-to-one tasks require recognizing that items belong to a class. A unit with maximum class-selectivity would be one that exclusively fired for the ten images of one identity. One can also discuss item-selectivity, a unit with maximum item-selectivity would be one that fired only to a single item. This item would have low class-selectivity.

For a one-to-one task, items do *not* need to be grouped into classes, so 'selectivity' typically refers to item-selectivity (e.g., the proportion of items that a unit responds to). For example, in table 2.1 in chapter 2 taken from Földiák (2009), the sparsity and item-selectivity are given for a set of representations. Item-selectivity is also sometimes referred to as lifetime-sparsity. Földiák (2009), demonstrates that the mean population-sparsity of a model is typically equivalent to the mean lifetime-sparsity of a model. For example, if an average of 50% of units fire in response to each item, then the average unit fires to 50% of items. As such, the claim that models use sparse representations for arbitrary data in one-to-one tasks, applies to both to the number of simultaneously active units (population-sparsity) and to the number of items that activate a given unit (lifetime-sparsity).

In Hinton et al. (1986), the task of mapping unique relationships or the spelling-to-meaning of individual words is one-to-one. Similarly, in Coltheart et al. (2001), the task of mapping the spelling of individual words to their pronunciation is one-to-one. Therefore increased selectivity refers to item-selectivity. The complementary-learning-systems papers (McClelland et al., 1995; Kumaran et al., 2016), claim that firing is more sparse and selectivity in the hippocampal system compared to surrounding regions. This also appears to relate to learning one-to-one mappings. This can be seen for in the fact that they suggest that the highest possible selectivity is a 'grandmother cell' that responds to a single entity alongside references to 'lifetime-sparsity' (Kumaran et al., 2016). As such, these studies predict that representations with high population-sparsity and item-selectivity (or lifetime-sparsity) are well suited for arbitrary one-to-one tasks. This raises the question of the effect of arbitrariness on representations for many-to-one classification tasks.

## 4.4 Arbitrariness and class-selectivity

There is little in the literature that directly investigates the effect of arbitrariness on the class-selectivity of units. Farah and Wallace (1992) discussed the model from Hinton et al. (1986), in relation to semantically-bounded-anomias, and suggested that brain uses highly sparse and class-selective representations between phonology and semantics. Semantically-bounded-anomias are neuropsychological disorders in object naming where impairments are restricted to specific categories such as fruits and vegetables. Farah and Wallace (1992) postulate that there are populations of neurons with high selectivity for classes of semantically related items (e.g., fruits and vegetables), and that damage to such neurons leads to impairments in naming these categories. They describe such units as ‘partially distributed representations’: a neural population that is selective for a particular class of objects, e.g., neurons that selectively fire for faces. They differentiate between these and *grandmother cells* that would only fire for one particular face (e.g., one’s grandmother). They suggest that high class-selectivity is appropriate for these neurons as the mappings between phonology and semantics are relatively arbitrary. Their argument suggests that the high selectivity for arbitrary tasks can be applied to many-to-one tasks.

Morcos et al. (2018) investigated the impact of manipulating the arbitrariness of the input-output mappings on the representations learned by models in many-to-one tasks. They do not directly investigate the effects of arbitrariness on sparsity or selectivity, although they do suggest an association between arbitrariness and unit importance, and between unit importance and selectivity. They used imageNet and CIFAR-10 to compare the performance of models trained on the unmodified versions with those trained on arbitrary versions where some or all of the labels had been shuffled. They used cumulative ablation to assessed the impact the arbitrariness of the dataset on the types of representations learned. That is, they measured the drop in accuracy as they progressively lesioned a greater number of units until all units are removed (or until accuracy reached zero). They describe the rate at which accuracy dropped as the network’s ‘reliance on single directions’. They did not define ‘reliance on single-directions’, and it appears to be a misnomer: they performed cumulative ablations, so any drop in accuracy was due to the loss of an increasing number of directions. However, the drop in overall accuracy, or ‘reliance on single-directions’ can be seen as a measure of the importance of hidden-units to overall accuracy. The reasoning is that lesioning an important unit will cause a greater drop in accuracy than lesioning an unimportant unit. The cumulative ablation showed that the greater the proportion of labels that had been shuffled, the more quickly the accuracy dropped. In other words, the more arbitrary the data was, the more each unit contributed to overall accuracy. The means that each unit was individually, more important when trained on arbitrary data than for systematic data; hence the claim that these models were reliant on a single-directions.

In a second study Morcos et al. (2018) trained 200 models from different initialization on unmodified (systematic) data and found that the 5 models with the best generalization performance (i.e., test set accuracy) were also the most robust to cumulative ablations. They took this to suggest that the best representations for generalization will also be the most robust to lesioning, and therefore, that strong reliance on single directions may be harmful for generalization. This finding also suggests that models where units typically have low importance are better for generalization than models where units have high importance. This leaves us to interpret the significance of the finding that arbitrary datasets are associated with increased reliance on single directions. One way to interpret this finding is that for models trained on systematic data, there are fewer directions required to describe the data (e.g., the data can be projected onto lower dimensional manifolds), whereas for arbitrary data, there may be important variance in more directions. Since the network is the same size for both datasets, the model that represents systematic data will have more redundant or overlapping representations (i.e., multiple units representing the same thing). For arbitrary data, more unique directions are needed to describe each class, so there is less redundancy. Therefore, lesioning the systematic model is more likely to remove a redundant unit, with little impact on accuracy, whereas in the arbitrary model lesioning is more likely to remove a unique contribution or facet of the representation, leading to a drop in accuracy.

In a third study, Morcos et al. (2018) investigated the association between class-selectivity and unit importance by lesioning individual units in three models trained on unmodified datasets. For two models they find no significant relationship, and a negative correlation in the third, meaning that decreases in selectivity are associated with increases in importance. Whilst this study does not investigate arbitrariness, the findings may be relevant for interpreting their conclusions. For the experiments where they did manipulate arbitrariness, they did not did not report levels of selectivity or demonstrate any direct association between selectivity and reliance on single directions. Still, they do allude to a connection in the abstract: “networks which generalize well minimize their dependence on individual units by reducing their selectivity” (pg. 1, Morcos et al., 2018). At face value, this claim could be taken to suggest that selectivity will be lower for systematic tasks (e.g., good generalization and low reliance on single directions) than for arbitrary tasks (e.g., where generalization is poor and models are reliant on single-directions). However, their claim appears to contradict their own findings. They find a negative association between unit importance and selectivity (or no correlation), which suggests that the low importance of units in models that generalise well should be associated with higher selectivity (or be of no relevance to selectivity). In summary, although not directly tested, they appear to predict increased selectivity for arbitrary tasks compared to systematic tasks.

Vankov and Bowers (2017) investigated the effect of arbitrary input-output mappings on selectivity across three simulations using single layer MLPs with sigmoid units. They found no effect of arbitrariness on selectivity. They analysed units with Bowers' selectivity measure (B-sel, Bowers et al., 2014), which gives the minimal distance between one class and other classes (for details see 3, section 3.3.1). They characterise units as 'localist' if the absolute B-sel exceeds some threshold, which is initially set at .5, then later reduced to .3. In the first simulation, they trained MLPs on a one-to-one task (e.g., the 500 items in the dataset were mapped onto 500 output categories) using randomly generated binary data for the input. Output data did not use 1-hot encoding, but instead used randomly generated binary vectors with 50% zeros. Since the input and output were randomly generated, the task was highly arbitrary. They repeated the simulation ten times, with different data, and reported finding no units with an absolute B-sel > .3.

In a second experiment they trained models on many-to-one (classification) tasks with datasets of varying arbitrariness. Datasets had 50 classes with 10 items per class. Input data were bi-polar (composed of values of -1 and 1) vectors. They semi-randomly generated data whilst controlling the between and within-class similarity of the datasets, measured with cosine-similarity. First a prototype vector was made for each class. These 50 prototypes either had a cosine-similarity of .2 or .8, relating to high or low between-class similarity of the data. For these prototypes they then generated ten 'off-spring'. For each class the offspring either had a mean cosine-similarity of .2 or .8, giving high or low within-class similarity. This gave them four dataset: (a) high within-class similarity and high between-class similarity (HWHB), (b) high within-class similarity and low between-class similarity (HWLB), (c) low within-class similarity and high between-class similarity (LWHB), (d) low within-class similarity and high between-class similarity (LWLB). Again, output data were randomly generated binary vectors. They ran the experiment ten times, with different data each time, but they found no units with an absolute B-sel > .3.

In simulations 3a and 3b, MLPs were trained to categorize 400 photographs of 40 human faces (10 images per person, from different views). The images were grey-scale, with pixel intensities represented with values between 0 and 1. In the one-to-one condition (3a), each face was associated with with a randomly generated binary vector. This task was considered arbitrary in the sense that the similarity of the output vectors did not correspond to similarity in the input space. In the many-to-one condition (3b), the ten images for each identity were all associated with the same random binary vector. This task is less arbitrary, in that images of the same identity were mapped to the same output, although since output patterns were randomly generated, any similarly between individuals was not related to similarity in the output patterns. The experiment was run ten times, with different output data each time. For simulation 3a (one-to-one condition), they found no units with selectivity > .3. However, in 3b (many-to-one condition), they did find selective units. They do not report the average number, but they show a figure for one of the runs where 16 of the 500 units have a B-sel > .3; and nine of which had B-sel > .5 (See figure 1a in Vankov and Bowers, 2017).

In summary, Vankov and Bowers (2017) manipulated the arbitrariness of input-output mappings in simulation 2, but their measure found no selective units in any condition and therefore, no effect of arbitrariness on selectivity. In simulations 1 and 3, they did not control the arbitrariness of the mappings. They found selective units only occurred for simulation 3b in the many-to-one condition. They attribute the differences between selectivity in these studies to differences in the input data type, which was discrete in simulations 1 and 2, but continuous in simulation 3. However, the lack of selectivity for simulation 3a suggests that other factors are at play. An alternative explanation of their findings is that there were effects of arbitrariness on selectivity, which were not detected by their measure, or were not reported as they were still below their selectivity threshold. Finally, in 3b, there are 10 identities represented by the 16 selective units shown in Vankov and Bowers (2017), with four units showing high selectivity for person 13, and three units for person 33. This raises the question of why these classes were more likely to be represented with selective units than others. This could be due to differences between the inputs for identities (e.g., hair-style, glasses or skin-tone could make some identities more distinct). Equally, the use of randomly generated, distributed output vectors means that some outputs will share more similar than others.

In summary, the literature clearly suggests that increased arbitrariness is associated with increased sparsity, however, the literature is less clear about the effects on class-selectivity. Several authors predict increased selectivity for one-to-one tasks (e.g., lifetime-sparsity), although it is not clear how this prediction would apply to class-selectivity in a many-to-one task (Hinton et al., 1986; McClelland et al., 1995; Kumaran et al., 2016; Coltheart et al., 2001). Farah and Wallace (1992) do extend the findings from a one-to-one task in Hinton et al. (1986), to propose that there are neural populations with high class-selectivity which handle arbitrary mappings. Morcos et al. (2018) does not directly investigate any link between arbitrariness and selectivity, although he alludes to a possible association, and appears to predict increased selectivity for arbitrary tasks compared to systematic tasks. Vankov and Bowers (2017) did measure selectivity of models trained on arbitrary tasks, and found highly selective units. However, they found no highly-selective units in experiments where they manipulated the arbitrariness, making it difficult to draw any conclusions regarding the effect of arbitrariness. None of the literature surveyed predicted that class selectivity would decrease for arbitrary tasks, relative to systematic tasks.

The following studies will directly address the question of how the arbitrariness of input-output mappings relates to class-selectivity in neural networks. Understanding the effects of arbitrariness might help explain why some datasets typically have higher selectivity than other datasets (e.g., iris and MNIST in chapter 3); or why in some studies selective units are found for some classes but not others (e.g., Bowers et al., 2016; Vankov and Bowers, 2017; Zhou et al., 2017). I quantify the arbitrariness of datasets and classes using the within-class and between-class cosine-similarity. Selectivity is measures with maximum-informedness. I find a clear effect:

arbitrary input-output mappings are associated with reduced class-selectivity. Experiments 1 and 2 are based on simulation 2 from Vankov and Bowers (2017). In experiment 1 I generate datasets of varying systematicity and arbitrariness by controlling the within and between-class cosine-similarity. I find that selectivity is highest for the most systematic datasets, and is lowest for arbitrary datasets. In simulation 2 I generate datasets containing classes of varying arbitrariness. I find that classes with systematic mappings have higher selectivity than those with arbitrary mappings. In these studies I show that the level of selectivity is predicted well by the difference in within and between-class selectivity. In these studies I also find strong correlations between unit selectivity and unit sparseness. Experiments 3 manipulates the arbitrariness of input-output mappings of classes in the MNIST dataset by shuffling the class labels for some classes. I find that selectivity is reduced for arbitrary classes, and that datasets with the most arbitrary classes have the lowest selectivity. In Experiment 4 I increase the systematicity of MNIST classes by adding a predictive feature. Here I find that classes with increased systematicity have higher selectivity than classes without. I conclude that arbitrary input-output mappings are associated with lower selectivity and systematic ones with higher selectivity. Although the surveyed literature was not conclusive about the effects of arbitrariness on class-selectivity, I note that none of the literature predicted the *reduced* class-selectivity selectivity found in these experiments.

## 4.5 Methods

### 4.5.1 Cosine-similarity

Throughout this chapter I quantify the arbitrariness (or systematicity) of data using cosine-similarity. Cosine-similarity is a measure of the orientation of two vectors, and is equivalent to the angle between them. All input values in the datasets are positive, and as such cosine-similarity scores are in the range 0 to 1. For example, if vector A is [0, 1, 1], the cosine-similarity with identical vector B ([0, 1, 1]) is 1.0. The similarity with vector C ([1, 1, 1]) is .82; for vector D ([1, 0, 1]) it is .5. For vector E ([1, 0, 0]), which is orthogonal to A, the cosine-similarity is 0.0. All cosine-similarity calculations were performed by the sklearn cosine-similarity package (Pedregosa et al., 2011), which computes similarity as the normalised dot-product between two vectors.

For each class in a given dataset, I calculate the average within-class similarity, which is the mean of all pairwise similarity comparisons of items in that class. I also compute the average between-class similarity for each class. The average cosine-similarity between two classes is the average of all pairwise comparisons between items in those classes. The mean between-class similarity for a class is the average of the between-class similarities between one class and all other classes. For each class, I therefore have an average within-class similarity and between-class similarity score. The mean of these gives us the dataset's mean within-class and between-class similarity scores.

**Similarity-difference** Similarity-difference aims to represent the continuum of input-output mappings from highly arbitrary to highly systematic. For each dataset, and for each class within a dataset, similarity-difference summarised the difference in the mean within-class similarity and mean between-class similarity and it is given as:

$$\text{Similarity-difference} = \text{mean within-class similarity} - \text{mean between-class similarity} \tag{4.1}$$

This gives values in the range -1 to 1, with positive scores indicating that within-class similarity is higher than between. Scores close to zero indicate that these is little difference in within and between-class similarity.

### 4.5.2 Selectivity

To record the hidden unit activations, a dataset was presented to the trained model (with the weights frozen), and the network's prediction for each item was recorded. Items to which the network gave an incorrect response were then removed from the dataset. Correct items were then run through the network again and the activation value for each hidden layer unit was recorded and saved. For relu units, the activations were normalized to the range 0-1 prior to calculating the selectivity. All analysis was performed on these activation values.

Selectivity was measured with maximum informedness (Max-info), which is a multi-class version of Youden's J statistic (Powers, 2011). For details, see equations 3.4 to 3.7 in chapter 3.

Informedness scores are in the range -1 to 1. A high score indicates good discriminative power for that class as most of the given class are above the threshold and most items from other classes are below the threshold. Second, I get the maximum-informedness score for the class, which is simply the highest of these informedness scores, giving us the best possible discrimination score for that class. Finally, for each unit I report a single max-info score, which is the class with the highest max-info score. This tells us the unit's best performance as a discriminator for one of the classes. I also record the identity of the class with the highest max-info score.

Where mean selectivity is reported for a model, this the average of the unit max-info scores. This involves one max-info score per unit (the class with the highest max-info), rather than the average of all class' max-info scores. Where selectivity of a model is reported on a per-class basis I use two measures. The first is Selective Class Frequency (Sel. Class Freq.), which is a count of the frequency with which each class is most selective. That is, for how many units in the model does class-$x$ have the highest selectivity of all classes? The second measure is the Selective Class Mean (Sel. Class Mean), which gives the average selectivity of all units where class-$x$ is most selective. For example, if there were only three units in the model where class-$x$ had the highest selectivity, then Sel. Class Mean would be the average of these three selectivity scores.

One limitation of max-info is that is has ceiling effects: it gives a maximum score of 1.0 when the most active class is linearly-separable from all other classes (e.g., the least active item from class-$x$ is more active than the most active item from any other class). This typically is not a problem as in most units, max-info scores are below 1.0. However, it means that max-info is not able to differentiate between a unit where all items from the most active class have strong activation whilst all other items have zero activation; and a unit where all items from the most active class are only slightly more active than other items. A different selectivity measure, B-sel, suffers from the opposite problem to max-info, it scores a unit with 0.0 if it is *not* linearly separable (e.g., if no there is no class where *all* activations are higher than any other class). For examples of this ceiling effect see chapter 3, section 3.5.4. Where a class *is* linearly separable (e.g., where max-info is 1.0), B-sel gives the minimal distance between one class (e.g., the most active) and all other classes. For the visualizations of unit activations (experiment 1 and 4), I do have units with a max-info score of 1.0, so in this case I will report the max-info score and also B-sel.

$$\text{B-sel} = min(\text{class-}x \text{ activations}) - max(\text{not class-}x \text{ activations}) \tag{4.2}$$

For both max-info and B-sel, I only consider ON selectivity (e.g., class-$x$ has higher activation than other classes).

### 4.5.3 Unit density / sparsity

The other unit property that I report is sparsity, (e.g., 'lifetime-sparsity'), a measure of the proportion of items with activation greater than zero. This gives values in the range 0 to 1. A score of 1.0 indicates that *all* items in the dataset drive the unit (e.g., there are no items with an activation of zero), and as such, the representation is *dense*. A score close to zero indicates that that majority of items had zero activation, and as such, the representation is *sparse*. Since high scores indicate low-sparseness/high-density, I shall refer to this as a measure of unit density rather than sparsity.

### 4.5.4 Correlations

To investigate the relationship selectivity and similarity-difference, I ran Spearman rank order correlations. This relationship was investigated at the level of classes and also datasets. For class-level correlations I used the similarity-difference for each class, which was compared to the Sel. Class Mean for each class (e.g., the mean selectivity of all units that are selective for class-$x$). For dataset-level comparisons I used the mean similarity-difference for each dataset, and the mean selectivity for all model trained on that dataset. All correlations were run using the 'stats' package from scipy (Millman and Aivazis, 2011).

### 4.5.5 Kruskal-Wallis H-test for independent samples

To compare results between different groups I used the Kruskal-Wallis test by ranks. This is a rank-based, non-parametric alternative to one-way ANOVA that can be used to determine whether there are significant differences between groups, where the group values are not normally distributed (Kruskal and Wallis, 1952). It tests the null hypothesis that the samples are drawn from the same distribution. If the test is significant, the this indicates that at least one sample is different to the others. In this case, post-hoc pairwise test for multiple comparisons of mean rank sums is used to determine which groups are different (e.g., 'Dunn's test', Dunn, 1964).

## 4.6 Experiment 1: Dataset Similarity

### 4.6.1 Introduction

In this simulation, models will be trained to categorise semi-randomly generated datasets that vary in terms of their systematicity (or arbitrariness), measured with between-class and within-class cosine-similarity. A systematic dataset would have high within-class similarity (items with the same output will have similar inputs) and low between-class similarity (items from different classes will have different inputs). This simulation is based on simulation 2 from Vankov and Bowers (2017). However, some details have been changed to make the methodology more generalizable to other tasks. For example, Vankov and Bowers (2017) used bi-polar data, the reason for this is not given, but it might relate to the fact that the cosine-similarity for two bi-polar vectors is in the range -1-1 whereas for binary vectors it is in the range 0-1. However, I will use binary datasets and continuously valued datasets in the range 0-1, so the cosine-similarity of these two types of data is more easily comparable than would be the case for bi-polar and continuous data. Vankov suggested that the use of discrete input data, rather than continuous values, contributed to the difference in selectivity between their simulations 2 and 3b. To test this, I will use both binary and continuous data for these simulations. Another differences from their study is that Vankov and Bowers (2017) used 50 classes with 10 items each, whereas our simulations will use 10 classes with 50 items. This is partly to make the dataset comparable with other common classification datasets with ten classes such as MNIST and CIFAR-10, and also to make the units easier to visualise as their are fewer classes along the y-axis. Finally, Vankov and Bowers (2017) used randomly generated, distributed output vectors for each class. This means that some output representations may be more similar to each other. This is problematic to the extent that 'items with similar input will have similar outputs, and items with dissimilar inputs will have dissimilar outputs'. For example, the output representations for classes A and B might be very dissimilar, sharing few units, whereas class C might have a similar output vector to class A (e.g., sharing many units). The models will therefore be biased to assume that something that looks very different to an A is more likely to be from class B (i.e., items with dissimilar inputs will have dissimilar outputs). The fact that these outputs were randomly generated means that this similarity is not controlled. While there are situations where distributed output representations may be beneficial, I will use 1-hot output representations, where all output patterns are equally dissimilar. Any similarity that exists in the data will come from the input representations only.

### 4.6.2 Methods

#### 4.6.2.1 Datasets

For this experiment, all datasets consists of input vectors of 500 elements. There are 500 items in each dataset, with 50 items in each of 10 classes. The datasets were constructed with semi-randomly generated data to have between-class and within-class similarity that was either low, medium or high. All cosine-similarity values are in the range 0 to 1. Low similarity was defined as having a cosine-similarity of ~.5, medium as ~.73 and high similarity as ~.85. Since it was not possible to generate datasets where the between-class similarity was higher on average than the within-class similarity this leaves six possible dataset conditions:

- HBHW: High between-class similarity and high within-class similarity
- MBHW: Medium between-class similarity and high within-class similarity
- MBMW: Medium between-class similarity and medium within-class similarity
- LBHW: Low between-class similarity and high within-class similarity
- LBMW: Low between-class similarity and medium within-class similarity
- LBLW: Low between-class similarity and low within-class similarity

The datasets construction process consists of two stages: 1. Generate ten 'prototype' vectors, from which the classes will be constructed. 2. From these prototypes, 50 items are generated by making unique changes to copies of the prototype.

**Generating the prototypes** I begin by constructing 'initial' vectors. I start with a vector of 250 zeros, to which I add 250 non-zero elements. For binary datasets these are all ones; for continuous datasets, these were sampled from a continuous uniform distribution between zero and one. These values were then shuffled with the zeros to create the 'initial' vector of 500 elements. The class prototypes are generated from these initial vectors. The similarity of these prototype vectors to each other will ultimately inform the between-class similarity of the dataset. For datasets with low between-class similarity I generated 10 'initial' vectors which were used as the class 'prototypes'. For datasets with high or medium between-class similarity, the 10 prototypes were constructed from a single initial vector. From the initial vector, a small (10) or medium (50) number of elements were selected at random and changed to make the fist prototype. The between-class similarity will be medium where a medium number of elements were changed, and will be high if a small number of elements were changed. For binary data, these changes involved flipping a one to a zero or vice versa. For continuous data, zeros were changed to a random float and vice versa. This was repeated for all ten class prototypes, with the same number of changes being made to the initial vector, but the locations of the changes selected at random each time. The original initial vector was not used as a prototype.

**Generate items for each class from the prototypes** From each prototype, I generate 50 items by copying the prototype and changing some elements. The original prototypes do not appear as items in the dataset. The elements to be changed are selected at random for each new item. The similarity of these items to the prototype will inform the within-class similarity. The within-class similarity was controlled with one of the following processes:

- **Change-proportion** The proportion of elements where the items differ from the prototype controls the similarity. For high within-class similarity, a small proportion of elements differ from the prototype; for low within-class similarity, a high proportion of elements differ from the prototype. This method was used to generate binary and continuous data. To generate datasets where classes have high within-class similarity, a small proportion of the prototype vector (35 elements) was randomly selected and flipped to produce each item. For medium within-class similarity dataset, a medium proportion (70 elements) was flipped and a high proportion was flipped (175 elements) to generated datasets with low within-class similarity. For example, to construct a class with high within-class similarity, each item is made by flipping 35 of the elements (e.g., zero to non-zero and vice-versa).
- **Change-distance** This process was only used to generate continuous datasets. The same proportion (250 elements) of the prototype is changed for each item, but the size of the change is varied. For each of these elements, I sampled a change-size value from a distribution with a maximum that was low (.25), medium (.45) or high (.9) to give high, medium or low within-class similarity, respectively. The change-size value was generated from a uniform distributed between $\frac{\text{change-size}}{2}$ and change-size. Then a one or zero was randomly selected to determine whether the change-size should be added or subtracted from the original element to make the new value. If the new value was below zero, I just set the new value to zero. If the new value is greater than one, I used $1 -$ new value. For example, to construct a class with high within-class similarity, each item will differ from the prototype in 250 of the 500 elements. For each element, the difference will be a value between .125 and .25, that is either added or subtracted from the value for that element in the prototype.

I created two versions of each dataset category using the three processes described above, such that I had 2 binary change-proportion, 2 continuous change-proportion and 2 continuous change-distance datasets for each of the six categories, giving 36 dataset in total. For average similarity details for each dataset version, see figure 4.1. This experiment did not use separate training and test data; the data used for training was also used for testing and analysis of hidden unit activations.

#### 4.6.2.2 Models and Training

All models were trained using Keras neural network API (Chollet and Others, 2015) and the Theano library (Al-Rfou et al., 2016). Models for this experiment were multi-layer-perceptrons

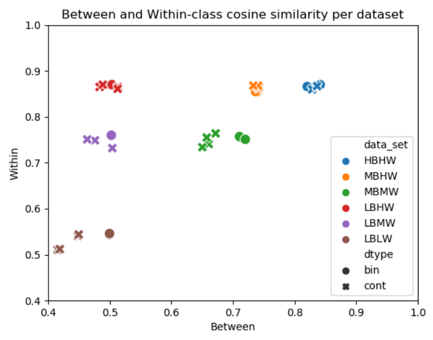


Figure 4.1: **Between and within-class cosine-similarity per dataset**. Plot showing the average within and between-class cosine-similarity per dataset. Between-class cosine-similarity is shown on the $x$-axis, within-class cosine-similarity on the $y$-axis. Binary data is marked with circles, continuous data with crosses. There are two versions of each binary dataset and four versions of each continuous dataset.

(MLPs) with a single hidden-layer of 10, 25, 50, 100 or 500 units, using sigmoid or relu activation functions. All weights were initialized with Glorot uniform initializer. Networks were trained using Stochastic Gradient Descent (SGD) with a batch size of 1, the learning rate was .001 and the maximum amount of training was 500 epochs. I used no explicit regularization, although there was an early stopping rule to prevent over-fitting: training was stopped if the loss was below .01 or if the loss did not improve by more than .001 over 10 epochs. I ran each dataset version three times from different weight initializations on each model, giving a total of 1080 simulation (36 dataset versions x 2 activation function x 5 model sizes x 3 runs). All simulations reached an accuracy of 1.0 within the 500 epochs of training. However, the early stopping rule meant that some conditions trained for less time than others. Across all models there were 3130 dead units which were not included in the analysis.

### 4.6.3 Results

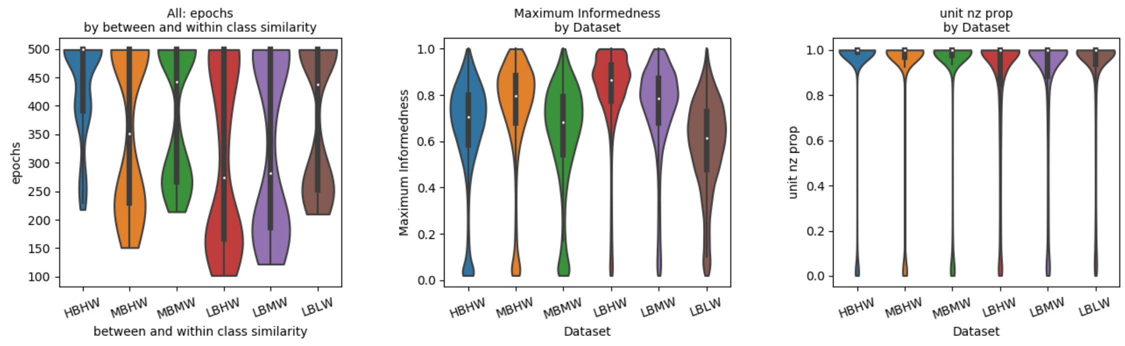
#### 4.6.3.1 Datasets

For training and selectivity mean descriptive statistics, see table 4.1 and for distributions see figure 4.2. All datasets achieved 100% accuracy. LBHW datasets took least time to train, then LBMW and MBHW with mean training times of 306, 323 and 356 epochs. The three datasets with equal between and within-class similarity took longer with LBLW, MBMW and HBHW taking 377, 386 and 435 epochs respectively. Together these results are consistent with the idea that HBHW was the most difficult dataset and LBHW was the easiest. The mean max-info scores, from highest to lowest are LBHW (.81), LBMW (.74), MBHW (.73), HBHW (.65), MBMW (.63) and LBLW (.58). There appeared to be little variation in the density scores, so a Kruskal-Wallis H-test was used, which showed that arbitrariness significantly affected unit density $H(5) = 2399$,

| Dataset | Simul-ations | Epochs | Training accuracy | Max Info | Unit Density |
|---|---|---|---|---|---|
| HBHW | 180 | 435 | 1.0 | .65 | .89 |
| MBHW | 180 | 356 | 1.0 | .73 | .87 |
| MBMW | 180 | 386 | 1.0 | .63 | .87 |
| LBHW | 180 | 306 | 1.0 | .81 | .83 |
| LBMW | 180 | 323 | 1.0 | .74 | .84 |
| LBLW | 180 | 377 | 1.0 | .58 | .87 |

Table 4.1: **Mean descriptive statistics for experiment 1**. All results are the average of 180 conditions, 6 runs x 3 data types (binary, continuous-change-dist, continuous change-proportion), x 2 activation function (sigmoid, relu), on 5 model sizes (10, 25, 50, 100, 500 units). Unit Density is the proportion of items with an activation > 0. Scores for max-info and Density are the average of all units in those condition, rather than the average of all conditions.

$p < .001$. Post-hoc Dunn tests (with a Bonferroni adjusted alpha of .003) were used to compare all pairs of groups. There were no significant difference between LBHW and LBMW or between MBHW and MBMW, but all other pairs showed significant differences (all $p$s < .001). Density was lowest for LBHW and LBMW (e.g., most sparse), followed by MBMW and MBHW, then LBLW and highest for HBHW.

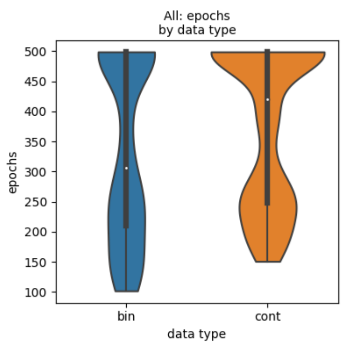


a) Training b) Selectivity c) Density

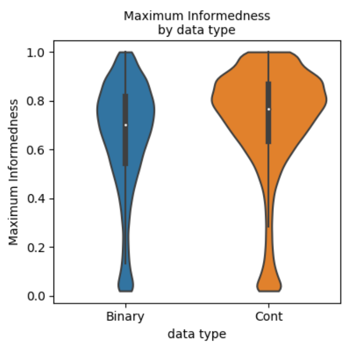
Figure 4.2: **All data types and activation functions by dataset.** For each category the plot displays the median (white dot), interquartile range (black bar) and a kernel density estimate of the underlying distribution (coloured area) such that the width is scaled by the number of observations. Plots show a) number of training epochs for each simulation; b) maximum informedness of most selective class for each unit for all items; c) Unit density: proportion of items with activation > 0.

#### 4.6.3.2 Data type

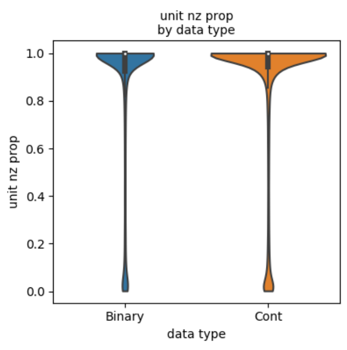
Figure 4.3 shows that the median number of epoch required to reach 100% accuracy was lower for models trained on binary data than for continuous data. Selectivity was slightly higher for continuous data and there was no significant effect of data type on density.



a) Training



b) Selectivity



c) Density

Figure 4.3: **All activation functions by data type.** For each category the plot displays the median (white dot), interquartile range (black bar) and a kernel density estimate of the underlying distribution (coloured area) such that the width is scaled by the number of observations. Plots show a) number of training epochs for each simulation; b) maximum informedness of most selective class for each unit for all items; c) Unit density: proportion of items with activation > 0.

#### 4.6.3.3 Similarity-difference

The relationship between between and within-class similarity for each dataset can be summarised as the difference between these values (e.g., within-class similarity minus between-class similarity). Figure 4.4 plots the within-between difference on the $x-axis$ and the mean selectivity of models trained on each dataset on the $y$-axis. This figure suggests that much of the differences in mean selectivity for these datasets can be predicted by the difference in within and between-class similarity.

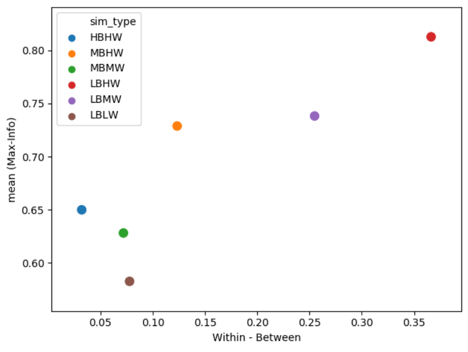


Figure 4.4: **Plot showing the similarity difference Vs mean selectivity**. The $x$-axis shows the similarity-difference, which is the dataset's average between-class minus the average within-class cosine-similarity. The $y$-axis shows the mean selectivity for each dataset. The plot shows that levels of selectivity are higher for datasets where the within-class similarity is higher than the between-class similarity. Data-points represent the average similarity-difference per class for the 6 versions of each datasets used in this study.

### 4.6.4 Discussion

In order to investigate the effect of arbitrariness on selectivity, I trained single layer neural networks to solve datasets with one of six between and within-class similarity profiles. Selectivity was higher for systematic datasets, where the within-class similarity is higher than between-class similarity (LBHW, LBMW and MBHW). The dataset with the highest selectivity was LBHW, which has the largest difference between the within and between-class similarities. Selectivity was lowest for arbitrary datasets where the within and between-class similarity are the same (HBHW, MBMW, LBLW). In these datasets, two items from the same class are no more similar than two items from different classes. The lowest average selectivity was for LBLW. I find that the effects of arbitrariness on selectivity are well predicted by the similarity-difference, which is the dataset mean within-class cosine-similarity minus the mean between-class cosine-similarity. My results contrast with (Vankov and Bowers, 2017) simulation 2, where they found no effect of arbitrariness on selectivity. One likely reason why they found no effect is that their selectivity measure (B-sel) has floor effects: units only receive a score greater than zero where classes are linearly separable. Further, they only reported the selectivity of units with a B-sel > .3. In their simulation 2, they found no units with this level of selectivity, so they were unable to compare the respective selectivity of the various conditions.

The effects of arbitrariness on training times follow a similar pattern to the effects on selectivity and suggest that the easiest datasets are those where the within-class similarity is higher than the between-class similarity (i.e., LBHW, LBMW, MBHW). Datasets where the within and between-class similarity are the same (i.e., LBLW, MBMW and HBHW) took longer to train, confirming that they were more difficult.

I found significant effects of arbitrariness on unit density: although the pattern of results is not as neat as for selectivity or training times, I still find that the most systematic dataset (LBHW) had the lowest density, whilst the highest density was for HBHW, which is highly arbitrary. My results are broadly consistent with the claim that models use sparser representations to solve arbitrary tasks, and more dense representations for systematic tasks (e.g., Plaut and McClelland, 2000, 2010).

Although (Vankov and Bowers, 2017) found no selective units in their simulations 1 and 2, they did find selective units in the many-to-one condition of their simulation 3. They attributed the difference in selectivity between these simulations to differences in the input data type – which was discrete in the first two simulations and continuous for simulation 3. Consistent with their intuition I find that selectivity is slightly higher for continuous data than binary data, although there are no differences in the pattern of results between data types.

In experiment 2, I will further examine the effects of between and within-class similarity. However, rather than comparing the effects between datasets with various levels of similarity, experiment 2 will focus on differences between classes within a dataset.

## 4.7 Experiment 2: Class Similarity

### 4.7.1 Introduction

This study will address the question of why selectivity is higher for some classes than for others. For example, in Vankov and Bowers (2017) simulation 3b (many-to-one), they ran models with 500 units on a classification tasks with 40 classes. They found 16 units which had high selectivity, representing ten of the classes. This means that 30 of the classes did not have any highly selective units, whilst classes 13, 33 and 19 had four, three and two selective units each respectively. One possible explanation for the higher selectivity for these classes is that their input-output mappings were more systematic than other classes. This experiment aims to investigate this.

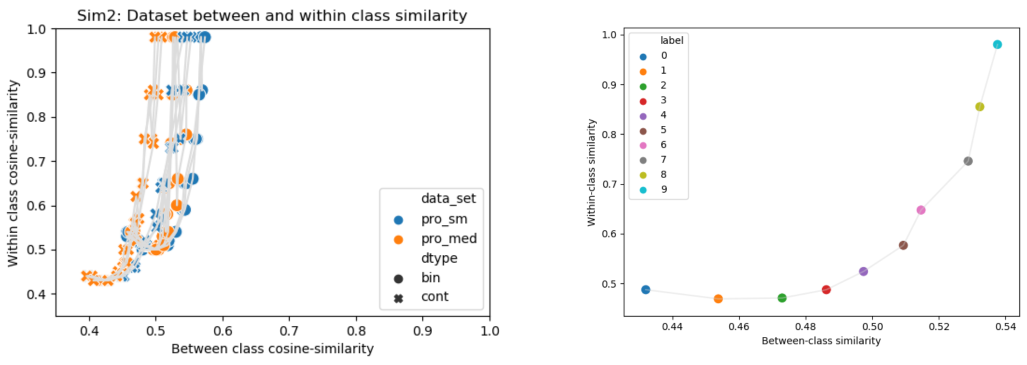
### 4.7.2 Methods

#### 4.7.2.1 Datasets

Datasets in this experiment were generated such that the between and within-class similarity varied for classes within the same dataset. The aim was that some classes should have higher within-class similarity than others, and some classes should have higher between-class similarity than others.

I started with a single initial vector, as was the case in experiment 1 for generating high or medium between-class similarity datasets, using the 'change-proportion' method (see section 4.6.2.1). However, rather than changing the same proportion of elements for each class, the number of elements to change was incrementally smaller for each class. In other words, I changed lots of elements to make the first prototype, such that it was very different to the initial vector, and by the time I got to the last class I changed few elements so this prototype was very similar to the initial vector. I tried various routines for this and ended up using two which gave us datasets with the properties I wanted. In the small-changes routine, the number of elements where each prototypes differed from the initial vector was 95, 85, 75, 65, 55, 45, 35, 25, 15 and 5. For the medium-changes routine the number of different elements were 185, 165, 145, 125, 105, 85, 65, 45, 25 and 5. These methods both gave us 10 prototypes of varying similarity to each other, e.g., Prototype 0 was very different to the other prototypes and class 9 was very similar to the others. I made three versions from different start vectors of the small and medium changes prototypes with binary and continuous data, giving 12 sets of prototypes in all.

To generate items for each class from these prototypes I used a similar method of varying the proportion of elements changed for each class. For class 0, I generated items by changing 320 randomly selected elements of the prototype for each item. This mean that items in class 0 had little in common with each other so class 0 would have low within-class similarity. I decreased the number of elements to change by 35 for each class, such that in class 9, all items only differed in 5 positions from their prototype, so were all highly similar. There was little difference in the final datasets between those generated with the small-changes and medium-changes routines. As such, they are treated 12 versions of the same dataset. For details of the within and between-class similarity for each class, and the averages of the 12 versions, see figure 4.5. The plots show that for all versions, the first classes have low within and between-class cosine-similarity. The between-class similarity slowly increases with each class but remains low. The within-class similarity has no change or a small decrease for the first couple of classes, then it increased until the later classes, which have high within-class similarity.



a) all versions b) averaged values

Figure 4.5: **Between and within-class cosine-similarity per class**. *Left*: Plot showing the between and within-class similarity for each class in each of the 12 dataset versions used in the experiment. Classes in each datasets are joined with a grey line from Class 0 in the lower left to class 9 at the top. Pro-sm are datasets with the small-changes prototype method and pro-med denotes medium-changes. *Right*: Plot showing the average similarity per class for the 12 datasets used in this study. The axis are truncated in this plot.

#### 4.7.2.2 Models and Training

Models and training were the same as for experiment 1 (see section 4.6.2.2), except that all models had relu units. Each of the 12 dataset versions was trained once on each of the 5 models sizes, giving a total of 60 simulations. All simulations trained to an accuracy of 1.0. There were 14 dead units which were not included in the analysis. The same data used for training was also used for all testing and analysis of hidden unit activations.

### 4.7.3 Results

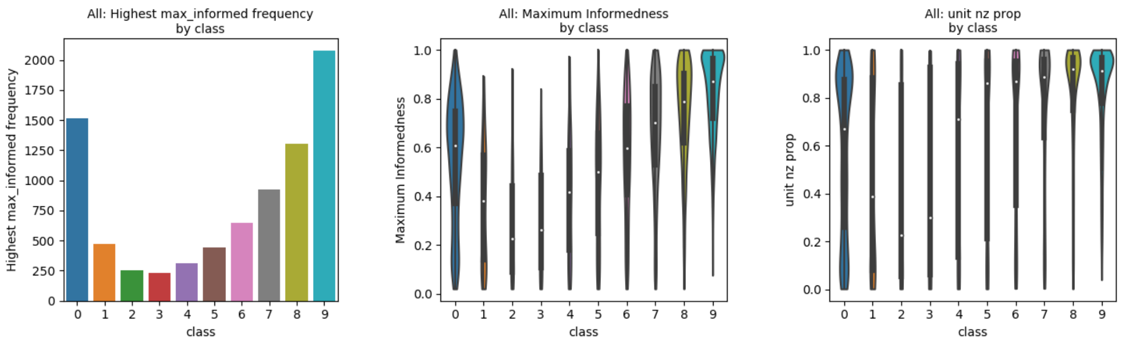
#### 4.7.3.1 Dataset

Table 4.2 shows the mean descriptive statistics for experiment 2. There were no significant differences between data generated from the small changes and medium changes prototype conditions so for the rest of the analysis, they will be collapsed into one dataset with the properties shown in the Total row of table 4.2.

| Dataset | sims | Epochs | Training accuracy | Max Info | Unit Density |
|---|---|---|---|---|---|
| small changes | 30 | 311 | 1.0 | .63 | .72 |
| med changes | 30 | 334 | 1.0 | .62 | .75 |
| **Total** | **60** | **322** | **1.0** | **.63** | **.74** |

Table 4.2: **Mean descriptive statistics for experiment 2.** Dataset results are the average of 30 conditions, 3 dataset iterations x 2 data types (binary, continuous change-proportion), x 5 model sizes (10, 25, 50, 100, 500 units). Unit density is the proportion of items with an activation > 0. Total is the averages of all datasets. Scores for max-info and density are the average of all units in those condition, rather than the average of all conditions.

#### 4.7.3.2 Classes



a) Sel. Class Freq. b) Sel. Class Mean c) Sel. Class Density

Figure 4.6: **Selectivity per class for each dataset.** For each unit, one class has the highest selectivity score. a) Shows how frequently each class was the most selective. b) Shows the mean Max. Info for units where the given class had the highest selectivity. c) Shows the density of units where the given class was the most selective. There were a total of 6500 hidden-units across the 30 models for each dataset.

Figure 4.6 shows the selectivity and density of units on a per-class basis. Panel a shows the frequency with which each class was the most selective across all models in this experiment. Class 9, which had the highest within-class similarity, was the most selective class for over 25% of all units. The next most frequent was class 0, which had the lowest between-class selectivity. Panel b shows the mean selectivity of each class in the units where that class is most selective. It follows a similar pattern to panel a, with an skewed U-shape. Class 9 has the highest mean selectivity score, scores then decrease to class 2 which has the lowest average. Class 1 is ranked 8th most selective and class 0 is 5th. Panel c shows the density of the units where each class is most selective. Again there is a skewed U-shape, class 2 and 3 are the least dense, classes 0, 1, 4 and 5 are in the middle, and classes 6 to 9 are very dense. For units where classes 6 to 9 are most selective, over 80% of all items have activation above zero. For units where class 2 is most selective, less than 25% of items are active. This is consistent with the claims that increased arbitrariness is associated with increased sparsity (e.g., reduced density, Plaut and McClelland, 2000, 2010)

The skewed U-shape pattern of results seen in figure 4.6 can be understood with comparison to figure 4.5 which shows the between and within-class similarity scores for the six datasets. This plot also has an offset U-shape. Class 9 from all datasets is shown at the top of the figure. From classes 9 to 5, there are small incremental decreases in between-class similarity. But much more importantly, there are large decreases in the within-class similarity. So differences in selectivity and density for classes 5, 6, 7, 8 and 9 can be understood as changes in their within-class similarity. For example, the reason why selectivity was highest for class 9 is because the items in class 9 were all very similar. If one item from class 9 strongly activated a unit, it is likely that many other items from class 9 will have high activation at that unit. At the other end of the figure, classes 0, 1, 2 and 3 all have similar within-class similarity, and vary in their between-class similarity. Differences in the selectivity and density for these classes can be attributed to differences in their between-class similarity. For example, the number of units that are most selective for class 0 is higher than for class 2 because the class 0 has lower between-class similarity than class 2, as such the extent to which two items from the same class are more alike than two items from different classes is stronger for class 0 than for class 2. If a unit fires strongly to one item from class 0, it is more likely that it will fire more strongly for another item from class 0 than for an item from a different class, due to the relative similarity of these items to the first item from class 0. This will result in many strongly activated items from class 0, and high selectivity for class 0. If a unit fires strongly to an item from class 2, it is equally likely to fire strongly to a second item from class 2 or an item from a different class, since items from class 2 are no more alike than an item from class 2 is to a item from a different class. This results in low selectivity for class 2. Figure 4.7 plots the similarity-difference (within minus between class similarity) against the mean selectivity for units where each class was most selective. There is a significant strong positive correlation between class similarity-difference and average class selectivity (Spearman $r(8) = .99, p < .001$). Increases in the difference score for a class are associated with increases in the mean selectivity of units where that class is most selective.

#### 4.7.3.3 Units of interest

Figure 4.8 shows all nine active units from one of the models with 10 relu units (unit 4 was dead and is not included). In this model I see that units that are most selective for systematic classes have higher selectivity and higher density than units that are most selective for arbitrary classes. It is also worth noting that only 6 of the ten classes are most selective in one of these units, there are no units which are most most selective for classes 2, 4, 5 and 7. Looking more closely at class 2, as well as never being most active, I also see that it is never the the least active class (e.g., not selectively OFF), it always has intermediate activation values. (In units 1 and 7, class 2 has a

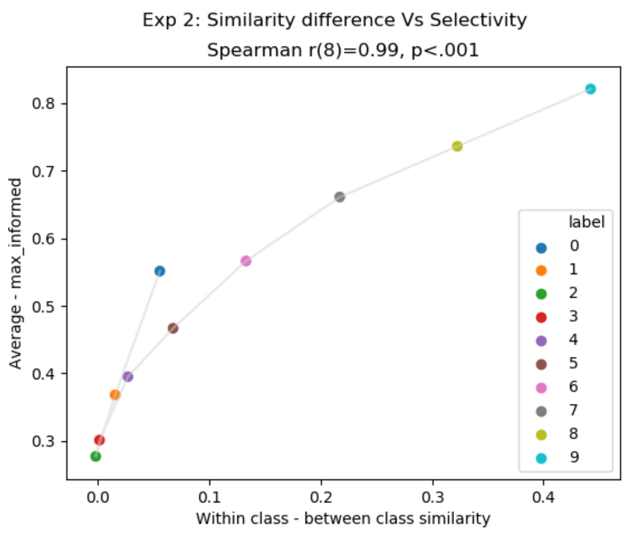
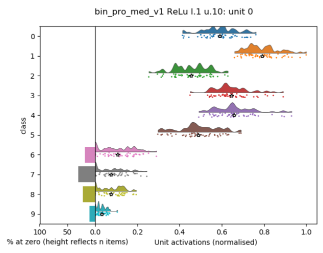


Figure 4.7: **Plot showing the similarity-difference Vs mean selectivity per class**. The *x*-axis shows the similarity-difference (between-class similarity minus within-class similarity per class). The *y*-axis shows the mean selectivity for each class in units where that class is most selective. The plot shows that levels of selectivity is higher for classes where the within-class similarity is higher than the between-class similarity. Datasets are joined with a grey line from Class 0 in the middle left to class 9 at the top right. Datapoints represent the average values per class for the 12 datasets used in this study.

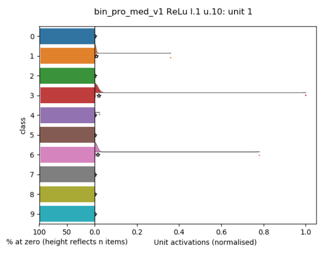
many items with no activation, but so do many other classes, class 2 is not unique in this unit). The model scored 100% accuracy on all classes, including class 2, 4, 5 and 7. This highlights the fact that a model does not need to have a selective unit for each class in order to correctly classify it. Items from class 2 were presumably correctly identified based on the combined intermediate activation values from these units. Put differently, class 2 was adequately represented in the model by units that were most selective for classes other than class 2.

### 4.7.4 Discussion

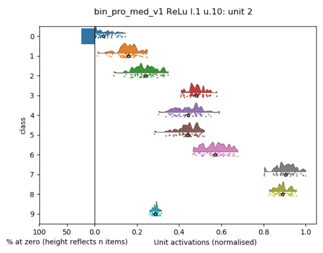
In simulation 2 I investigated the effects of arbitrariness on similarity using datasets where the within and between-class similarity varied between classes. Across 12 datasets I varied the within-class similarity of classes 0 to 9 from a mean cosine-similarity of .45 to .95. The between-class similarity also varied from around .45 to .55. The systematicity or arbitrariness of the classes is summarized as the similarity-difference. Class 9 was the most systematic (highest within-class similarity relative to between-class similarity) and class 2 was most arbitrary (least difference in the within and between-class similarity scores). I find that the selectivity scores associated with each class is predicted well by the similarity-difference scores for each class. Selectivity was highest in general for class 9: it was the most frequently the most selective class and selectivity scores were higher for these units than for units where a different class was most



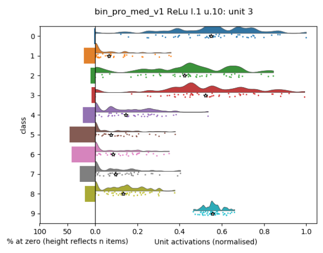
a) Unit 0
MI = .89 (1)
B-sel = 0.0



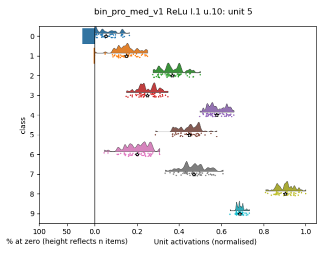
Unit 1
MI = .02 (3)
B-sel = 0.0



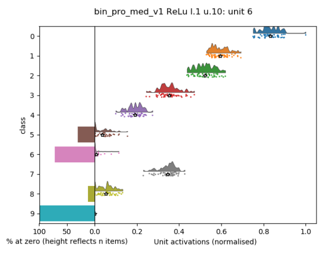
Unit 2
MI = .90 (8)
B-sel = 0.0



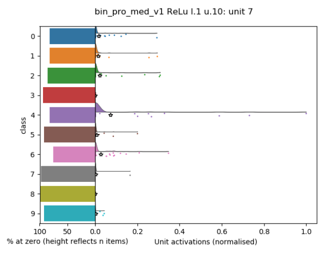
a) Unit 3
MI = .81 (9)
B-sel = 0.0



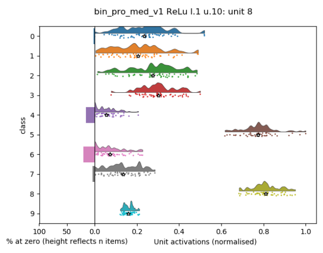
Unit 5
MI = 1.0 (8)
B-sel = .08



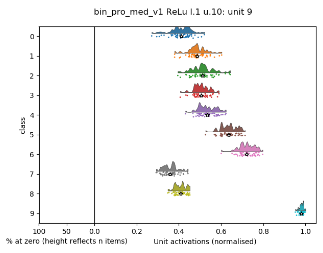
Unit 6
MI = 1.0 (0)
B-sel = .06



a) Unit 7
MI = .17 (6)
B-sel = 0.0



Unit 8
MI = .90 (8)
B-sel = 0.0



Unit 9
MI = 1.0 (9)
B-sel = .15

Figure 4.8: **Visualisation of units from experiment 2** All active units from: run: 0; model: 10 relu units; data type: binary; prototype method: medium changes, dataset version: 2. Note: unit 4 was dead (e.g., no active items) and was excluded from the analysis. MI denote the max-info score, the class which the score refers to is in parenthesis. B-sel scores are given for units where max-info was 1.0.

selective. The reason being that if a one item from class 9 is strongly activated by a unit, then other items from class 9, being similar to the first, are also likely to also cause strong activation. As such many class 9 items will have similarly high activation, giving class 9 higher selectivity than classes where only a few items are strongly activated. For a unit that is highly activated by an item from an arbitrary class (e.g., class 2), items from a different class is just as likely to cause strong activation as other items from class 2, since they are equally similar. As a result, units with many active items from class 2 will also have many active items from other classes, and so class 2 would not have a high selectivity score. However, this is not an accurate characterization of units which are most selective for arbitrary classes, I shall return to this point in a moment.

In simulation 2 I found that units that were most selective for systematic classes, were denser than units that were most selective for an arbitrary class. In other words, I find that units increase their sparsity to deal with arbitrariness in the data, consistent with earlier claims (Plaut and McClelland, 2000, 2010). For units that are most most selective for an arbitrary class (e.g., class 2), I suggested earlier that the selectivity was low because there were many items from other classes with strong activations alongside the items from class 2. In fact, units that fire strongly for one item from class 2, reduce the likelihood that they will fire to other items, regardless of their class. As such, there are few active items, most of which are for class 2. The level of selectivity is low, because the unit only fires to a small section of items from class 2 (e.g., unit has a low hit rate for class 2). I plotted the 9 active units from a 10 unit model and show that not all classes have a unit that is most selective for them. For example, no unit was most selective for class 2, there was no unit where class 2 was least active either, meaning that it always had intermediate values. This demonstrates that selective units for a class are not required for good performance on that class. This also demonstrates that selective units contribute to the representation of items from classes other than the one the unit is most selective to.

In summary, in experiment 1 I show that mean levels of selectivity for a dataset can be predicted by the dataset's mean similarity difference. In experiment 2 I extend this to show that the relative levels of selectivity associated with each class in a dataset can be predicted by the class' mean similarity difference. In experiment 2 I also demonstrate that units may contribute to the representation of items from many classes, not just the class that they are most selective for. This is especially the case for arbitrary classes, which are represented by the co-activation of units which themselves are most selective for other classes. My finding that selectivity is lower for arbitrary tasks compared to systematic tasks runs contrary to predictions that class-selectivity would be higher (e.g., Morcos et al., 2018; Farah and Wallace, 1992) or the finding of no effect of arbitrariness on selectivity (Vankov and Bowers, 2017).

## 4.8 Arbitrary and Systematic Mappings in MNIST: Experiments 3 and 4

## 4.9 Introduction

In the previous two experiments, I demonstrated that the levels of selectivity associated with a dataset (experiment 1) or class (experiment 2) can be manipulated by controlling the within and between-class similarity of classes when generating the data. However, these experiments used semi-randomly generated data with no meaningful (to humans) content or structure. The input features that were changed to influence similarity were selected at random. Here I turn my attention to MNIST, a well established digit recognition task within the machine-learning community. I investigate whether the mean levels of selectivity can be manipulated by changing the arbitrariness and systematicity. In experiment 3, I increase the arbitrariness of the dataset by shuffling the class-labels, as was done in previous research (Zhang et al., 2016; Morcos et al., 2018). In experiment 4, I increase the systematicity of the dataset by adding a predictive feature to each class.

## 4.10 Arbitrary and Systematic Mappings in MNIST Methods

### 4.10.1 Datasets

The following experiments will used datasets that are based on MNIST (LeCun et al., 1998). MNIST consists of images of handwritten digits (0 to 9). The dataset is grey scale, with white digits centred on a black background. Images are 28 x 28 pixels. The dataset consists of 60000 images in the training set and 10000 images in the test set. Standard data pre-processing methods were applied to the data (Karpathy, 2016). Original data was in the range 0 to 255, which was changed to the range 0 to 1 by dividing values by 255. Values were then zero-centred pixel-wise by subtracting the pixel mean activation, and normalised by dividing the pixel standard deviation.

### 4.10.2 Models and training

All models in experiments 3 and 4 were MLPs with four hidden layers of relu units. There were three model sizes (48, 100 or 500 hidden-units), with an equal number of units per layer. The learning rate was .01, with a batch-size of 32. The early stopping rule would stop training if the loss was below 0.01 or if the loss did not decrease by at least .001 over 100 epochs. All simulations stopped training before they reached the maximum of 5000 epochs. Models did not use any validation data during training. Models were only included in the final analysis if their mean test accuracy was greater than .15, and the per-class accuracy was greater than zero for all classes. Each condition was run 10 times, with a different weight initialization (with 'He normal' initializer, He et al., 2015). Therefore, there are 30 simulations for each dataset (3 model sizes x 10 runs).

## 4.11 Experiment 3: Arbitrary Classes

### 4.11.1 Introduction

This experiment will measure the effect on selectivity of increasing the arbitrariness of input-output mappings on a proportion of classes within a dataset. I used the MNIST dataset, along with three variations of MNIST which have increasing levels of arbitrariness. For arb 2, arb 5 and arb 8, the class-labels are randomly assigned for 2, 5 and 8 of the classes.

### 4.11.2 Methods

#### 4.11.2.1 Datasets

There were four datasets used in this experiment. The first is MNIST, all items in this dataset have their original class labels (for MNIST details, see 4.10.1). In the second dataset, a version of MNIST called 'arb 2', items from class 0 and class 1 are randomly assigned to either class 0 or 1 (with equal probability). As such, the mapping for two of the classes (classes 0 and 1) are arbitrary; all items from classes 2 to 9 retain their original labels. The next dataset is arb 5, in which items from the first five classes are randomly labelled with 0, 1, 2, 3 or 4; whilst classes 5 to 9 keep their original labels. For arb 8, the first 8 classes are randomly assigned to a class and classes 8 and 9 are unmodified. All datasets are based on the 60000 item MNIST training set. Models and training details for this experiment are as described in section 4.10.2.

### 4.11.3 Results

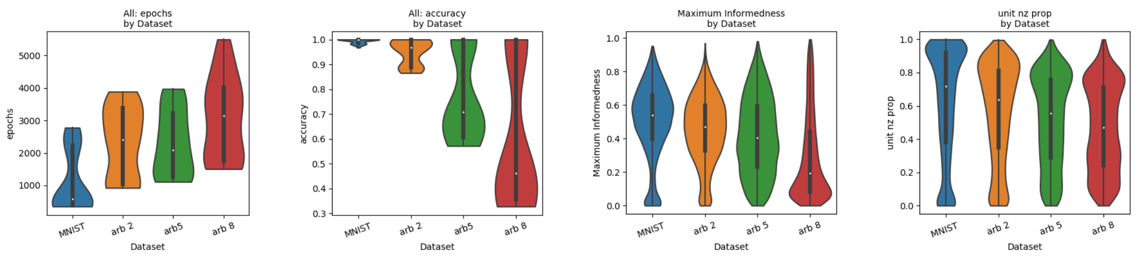
#### 4.11.3.1 Dataset

| Dataset | Sims | Epochs | Train accuracy | Max Info | Unit Density |
|---|---|---|---|---|---|
| MNIST | 30 | 1110 | 1.0 | .50 | .62 |
| arb 2 | 30 | 2334 | .95 | .45 | .57 |
| arb 5 | 30 | 2304 | .77 | .42 | .52 |
| arb 8 | 30 | 3063 | .60 | .29 | .47 |

Table 4.3: **Mean descriptive statistics for experiment 3.** All results are the average of 30 conditions. Unit density is the proportion of items with an activation > 0.

The descriptive statistics for experiment 3 are shown in table 4.3. Across all models there were 841 dead units which were excluded from the analysis. MNIST has the shortest training time and highest accuracy, and arb 8 has the longest training time and lowest accuracy. The training times for arb 2 and arb 5 were similar, although arb 5 stopped training after fewer epochs, in that time, arb 2 reached a higher accuracy than arb 5. Selectivity and density were highest for MNIST, and progressively decreased as additional classes were shuffled. As such, I again find that selectivity is highest for more systematic datasets and increasing the arbitrariness reduces the selectivity.

Accuracy for my models drops relative to the number of shuffled classes, with arb 8 reaching an accuracy of .6 after 3063 epochs. This appears inconsistent with Zhang et al. (2016), who trained models with shuffled labels to 100% accuracy. However, Zhang's smallest model has 512 units, whereas my largest has 500 and the smallest has 48 units. Average accuracy for arb 8 was 100% for models with 500 units, falling to 45% for 100 units and just 34% for 48 unit models. As such, the lower accuracy reported here is due to my use of smaller models.



a) Training b) Accuracy c) Selectivity d) Density

Figure 4.9: **Training and Selectivity results for Experiment 3 by dataset.** For each category the plot displays the median (white dot), interquartile range (black bar) and a kernel density estimate of the underlying distribution (coloured area) such that the width is scaled by the number of observations. Plots show a) number of training epochs for each simulation; b) Accuracy on the training data; c) maximum informedness of most selective class for each unit; d) Unit density: proportion of items with activation > 0.

#### 4.11.3.2 Classes

Figure 4.10 breaks down the results by class for each dataset. For MNIST I see that the scores are highest for classes 0 and 1 than for the other classes for all three measures. For arb 2, where classes 0 and 1 are shuffled together, these two classes not have the lowest scores for the three measures. For arb 5, where classes 0 to 4 are shuffled together, I see that scores are very low for classes 0 to 4. For arb 8, where classes 0 to 7 are shuffled, the scores are low for these first eight classes and as such, classes 8 and 9 dominate. In summary, increasing the arbitrariness of classes leads to reduced selectivity for those classes.

#### 4.11.3.3 Similarity-difference

Figure 4.11 shows plots of the difference in within and between class similarity for each class, plotted against the mean selectivity for units where that class was most selective. Panel a shows the MNIST training data, and in the top right of the figure I can see that class 1 (orange marker) has the largest similarity-difference. This might explain why the highest proportion of selective units were for class 1 (see top row of figure 4.10). Similarly, class 5 (brown marker) had the lowest similarity-difference and the fewest selective units. For MNIST classes, there is a significant strong positive correlation between similarity difference and mean selectivity ($r(8) = .79$, $p = .006$). However, for the other three datasets the correlations are not significant. This suggests that differences in selectivity can not be attributed to the difference in within and between class similarity. This is somewhat surprising, in that the manipulation (shuffling items from particular classes) has worked: shuffled classes have lower selectivity than unshuffled classes. Further more, the more classes that are shuffled together, the lower the selectivity is for each of those classes.

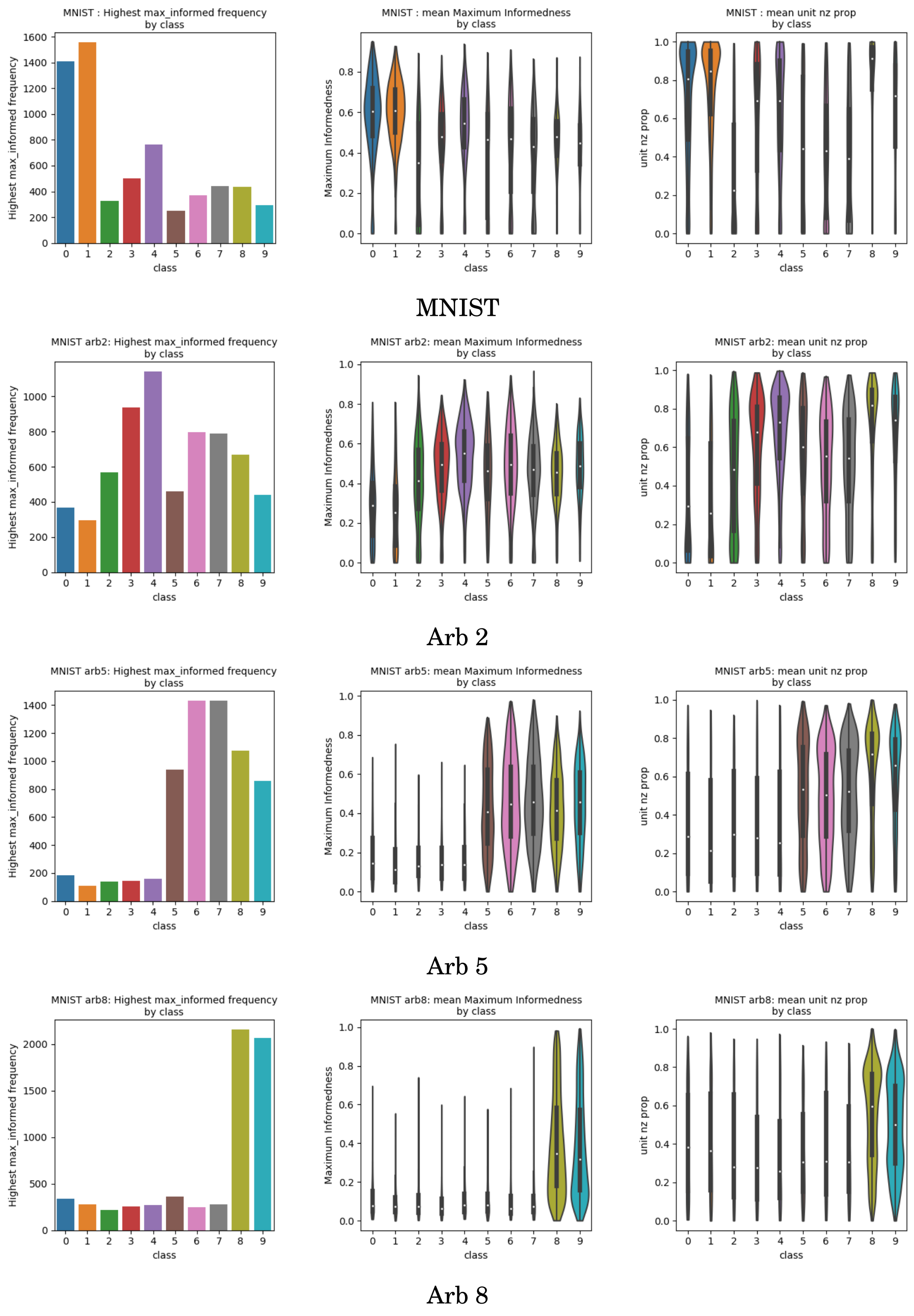
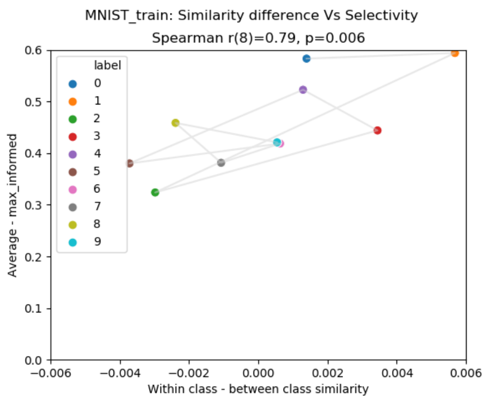
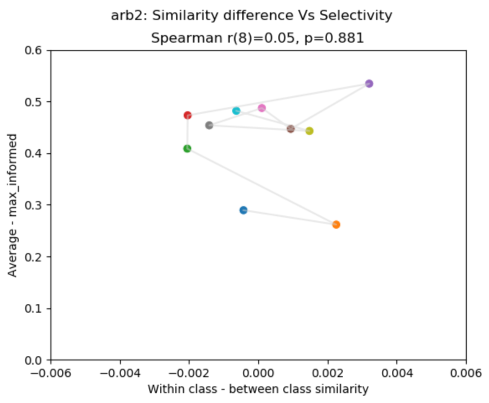


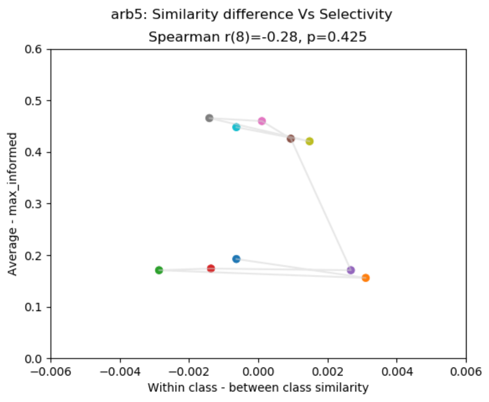
Figure 4.10: **Selectivity per-class for each dataset.** For each unit, one class has the highest selectivity score. Each row shows the mean results for each dataset. a) How frequently each class was the most selective. b) Average Max. Info for units where the given class is most selective. c) Proportion of items with activation > 0 in units where the given class was the most selective. There were 38159 hidden-units in the 30 models for each dataset.



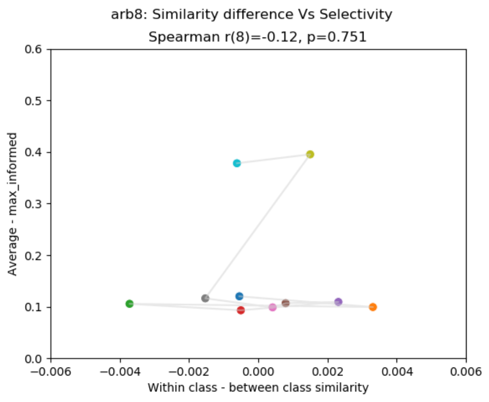
a) MNIST (train)



b) Arb 2



c) Arb 5



d) Arb 8

Figure 4.11: **Plots showing the similarity-difference and mean selectivity for each class** The $x$-axis shows the between-class similarity minus within-class similarity per class. The $y$-axis shows the mean selectivity for each class in units where that class is most selective. Datasets are joined in order with a grey line. Data-points represent the mean selectivity of all units where the class was most selective.

### 4.11.4 Discussion

In this study I find that increasing the arbitrariness of a class's input-output mappings reduces the levels of selectivity for that class. As a result, the most selective class is most frequently an unmanipulated class, and there are few units where arbitrary classes are most selective. The mean selectivity of the units where arbitrary classes are most selective is lower than for unmodified classes. For example, of the 6500 units in all models that were trained on the arb 8 dataset, about 4000 were most selective for class 8 or 9, which were the most systematic classes. Does this mean that these models devotes most of their resources to identifying the easiest (most systematic) classes? No it does not. The fact that a unit is most selective for one class does not mean that it does not represent important information about other items. I find that density was highest for the units that were most selective for classes 8 and 9, meaning that items from many classes were being represented by these units. As discussed in experiment 2, a class does not need to have units where it is most selective in order to have good accuracy. Items can be classified based on the activation of units that are most selective for other classes. Therefore, the fact that the majority of units are most selective for systematic classes does not mean that the model is more focused on those systematic classes. The dense representations in these units means that they contribute to the representation of items from many classes, not just the class with the highest selectivity.

I do find differences between this experiment and experiments 1 and 2, namely that similarity-difference does not predict selectivity as strongly in experiment 3. In experiment 1 and 2, the datasets (and classes) were designed to have a large amount of variance in within-class similarity, class means ranged from .47 to .98, and the difference scores were from -.002 to .44. In this experiment there was very little variance in similarity and difference scores. For the MNIST train set, all similarity scores for both between and within-class similarity are in the range .39 to .41, with difference scores between -.004 and .006. This means MNIST has even lower similarity scores than the LBLW dataset from experiment 1. Why is the within-class similarity so low in MNIST? One possible reason is that MNIST is a much larger dataset than those used in experiments 1 and 2. In experiment 2, the input data was vectors of 500 elements, whereas here there are 784 elements in the input of MNIST. This means that there are more ways in which two vectors can differ. Secondly, there are around 6000 items-per-class in MNIST, compared to only 50 items-per-class in experiments 1 and 2. Perhaps any apparent similarity between items is averaged-out in the mountain of pairwise comparisons that occur. For the unmodified MNIST, I do have a significant association between the similarity-differences for each class and selectivity. But since the within-class similarity is already so small, shuffling classes to make the new 'arb' datasets does not have the effect of further reducing the within-class similarity in any meaningful way. As such, I appear to have reached the limits of the precision that pairwise cosine-similarity comparisons can give us. Even though similarity-difference was not as good a predictor in these experiments, the experimental manipulations clearly had an impact on the models.

In experiment 3 I find that increasing the arbitrariness of classes leads to reduced selectivity for those classes. This raises the question of whether increasing the systematicity of classes will leads to higher levels of selectivity. The following experiments shall address this question.

## 4.12 Experiment 4: Systematic Classes

### 4.12.1 Introduction

Having established in experiment 3 that manipulations to decrease the systematicity of a dataset lead to reduced levels of selectivity, I now ask whether increasing the systematicity of a dataset will lead to increased selectivity? This is done by adding a predictive feature to some or all classes. Research has shown that when available, models will utilize a predictive feature that has been added to a dataset, at the expense of shape features (Malhotra et al., 2019). As such, my manipulation is likely to distract the model away from solving the task based on shape. Whilst this would not be desirable if I aimed to build a reliable digit classifier, my aim here is to asses the impact of systematicity on selectivity.

### 4.12.2 Methods

#### 4.12.2.1 Datasets

Five datasets will be used in the experiment. The first is the original, unmodified MNIST dataset. The the other four datasets are variations of MNIST, where the systematicity of input-output mapping is manipulated on a per-class bases so that there are systematic input-output mappings for none, some or all of the classes. The MNIST dataset is already systematic, in that similar looking inputs are typically mapped to the same output class and images that appear very different are likely to belong to different classes. However, this systematicity is far from perfect: for example, some fours look like nines and some sevens look like twos. The manipulation applied here to increase systematicity will not reduce the variance in how recognizable the digits are. Instead, it will add an additional feature that will perfectly predict class membership. This invariant, predictive feature consists of a set of 4 pixels which appear in an unique location for each class. To do this, forty of the 784 input channels, which correspond to pixels at the top edge of the image (and which do not contribute to representing any part of the digit displayed in any image) were used as locations for these features. Each class is associated with four of the 40 inputs, which are set a value of 1.0 for all items from their class, and are not modified for any items not from their class (prior to the manipulation, these pixels had values close to the dataset mean of around .25). These pixels will therefore perfectly predict whether a given image belongs to a given class. This feature can be said to increase the systematicity of the class, to the extent that images with the same feature belong to the same class and images with different features are from different classes. See figure 4.12 for examples of items with a predictive feature.

The four datasets containing the predictive features are as follows. For ‘pred 2’ the features are applied to all items from classes 0 and 1, whilst classes 2-9 are unchanged from the original dataset. For ‘pred 5’ the features are applied to each of the first five classes only. For ‘pred 8’ the feature appears for classes 0 to 7; and for ‘pred all’ each class had a predictive feature. Adding these predictive features to the inputs may well have the effect of stopping the dataset from learning to identify the items based on their ‘true’ representations (e.g., the handwritten digit displayed in the MNIST images). This is not of concern for the present purposes.

All datasets were variations on the MNIST ‘test-set’, of 10000 items. Testing and recording hidden unit activations was done with the same data used for training each model. For details of models and training for this experiment, see section 4.10.2.

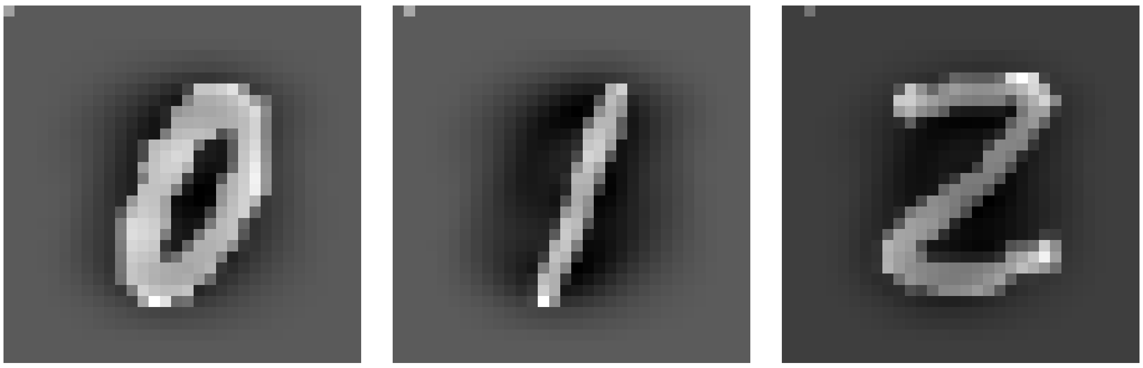


Figure 4.12: **Examples of the ‘pred all’ version of MNIST**, where four pixels appear in a unique location for each class has. For class 0 the pixels are in the top-left corner. For class 1, they are on the top row, in from the left corner, for class 2 they are again in a unique location on the top row, but further to the right. Pixels appeared in the same location for all items from each class.

### 4.12.3 Results

#### 4.12.3.1 Dataset

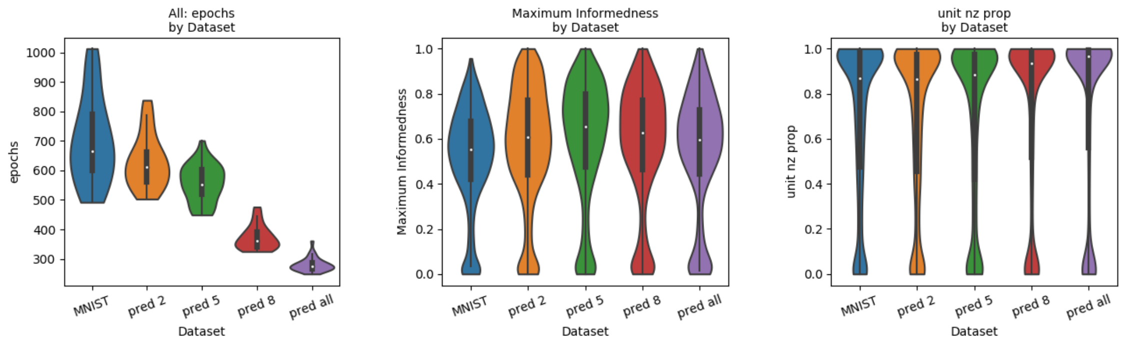
Descriptive statistics for experiment 4 are shown in table 4.4. All datasets reached 100% accuracy. There were 2269 dead units that were not included in the analysis. The amount of training required to reach 100% decreased as additional predictive classes were added. This confirms that fact that models were utilizing the predictive features where they were available, and that these features made the tasks easier.

Selectivity is lowest for the original MNIST dataset, but the increase in selectivity for dataset with predictive features was modest. To test whether there were there were significant differences in selectivity between datasets, I ran a Kruskal-Wallis test which found a significant effect of systematicity on selectivity ($H(4) = 585, p < .001$). Post-hoc Dunn tests (with a Bonferroni adjusted alpha of .005) found no significant difference between pred 2 and pred 8, but the selectivity of all

| Dataset | sims | Epochs | Training accuracy | Max Info | Unit Density |
|---|---|---|---|---|---|
| MNIST | 30 | 701 | 1.0 | .52 | .70 |
| pred 2 | 30 | 631 | 1.0 | .57 | .69 |
| pred 5 | 30 | 557 | 1.0 | .60 | .69 |
| pred 8 | 30 | 373 | 1.0 | .57 | .72 |
| pred all | 30 | 281 | 1.0 | .55 | .74 |

Table 4.4: **Mean descriptive statistics for experiment 4**. All results are the average of 30 conditions, 10 runs x 3 model sizes (10, 100, 500 units). Unit density is the proportion of items with an activation > 0. Scores for max-info and density are the average of all units in those condition, rather than the average of all conditions.

other pairs of datasets were significantly different (all tests *p*s < .001). It is interesting that the highest mean selectivity in this experiment is for pred 5, rather than for pred all. This is unlike experiment 3, where selectivity decreased as the number of arbitrary classes was increased. Pred all has the second lowest selectivity, despite having the highest systematicity. I also ran Kruskal-Wallis tests on density scores and found significant differences between the datasets ($H(4) = 460$, $p < .001$). Post-hoc Dunn tests (with a Bonferroni adjusted alpha of .005) found no significant differences between MNIST, pred 2 and pred 5. There were significant differences in density for pred 8 and all other datasets, and between pred all and all other datasets (all *p*s < .001).

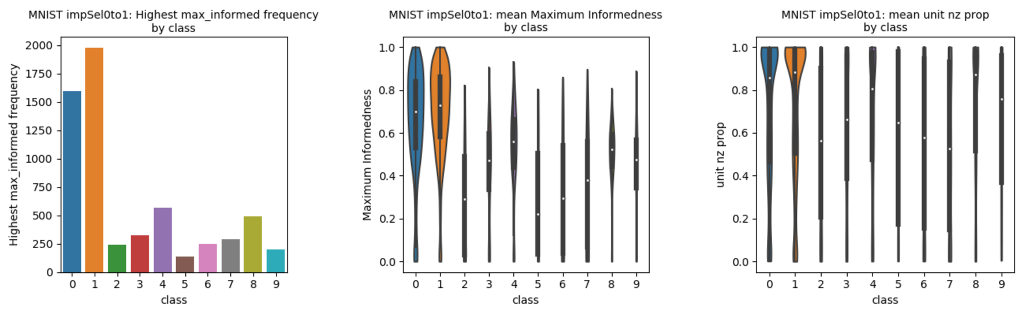


a) Training b) Selectivity c) Density

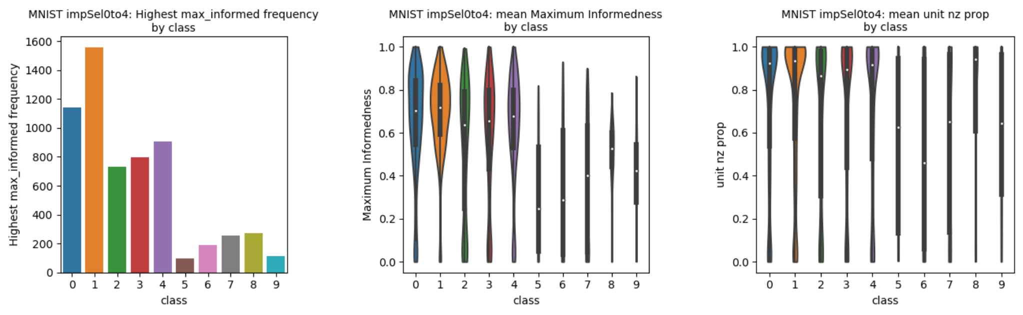
Figure 4.13: **Training and Selectivity results for Experiment 4 by dataset.** For each category the plot displays the median (white dot), interquartile range (black bar) and a kernel density estimate of the underlying distribution (coloured area) such that the width is scaled by the number of observations. Plots show a) number of training epochs for each simulation; b) maximum informedness of most selective class for each unit; c) Unit density: proportion of items with activation > 0.

#### 4.12.3.2 Classes

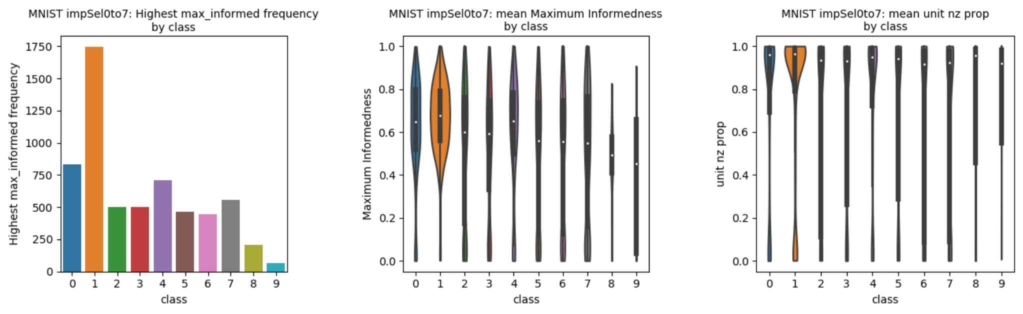
Figure 4.14 shows the results broken down by class for each dataset. Plots show that compared to unmodified classes, classes with a predictive feature have higher selectivity and the units where they are more selective are more dense than for classes without the predictive feature.



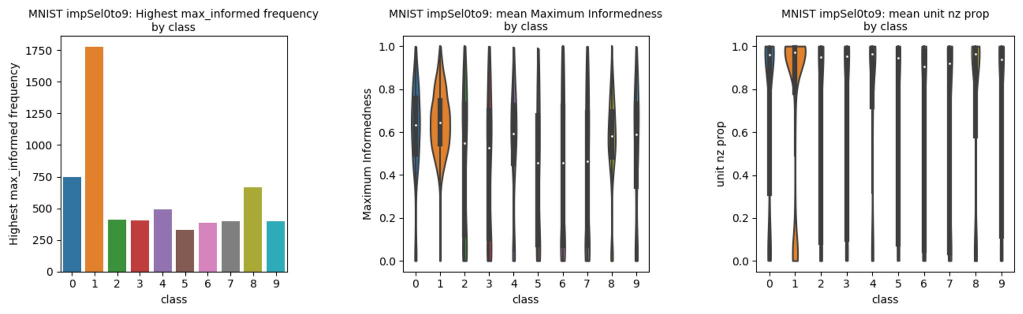
Pred 2



Pred 5



Pred 8

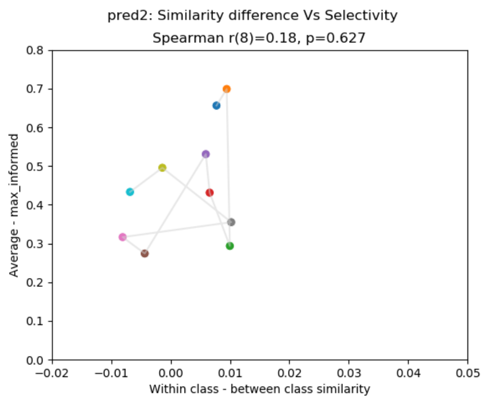


Pred all

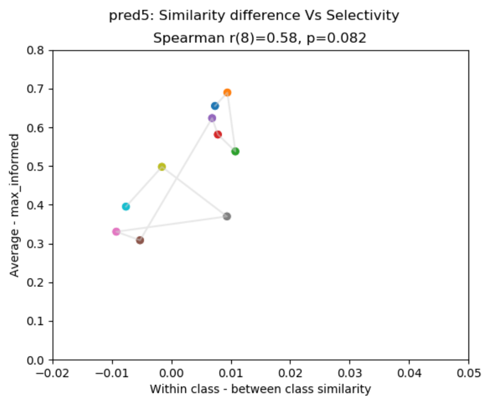
Figure 4.14: **Selectivity per class for each dataset.** For each unit, one class has the highest selectivity score. a) How frequently each class was most selective. b) Average Max. Info for units where the given class was most selective. c) Proportion of items with activation > 0 in units where the given class was most selective. Unmodified MNIST not shown, but values were similar to top row of figure 4.10.

#### 4.12.3.3 Similarity-difference

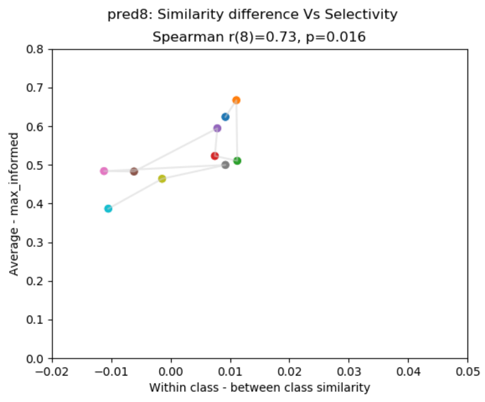
Figure 4.15 shows plots of the similarity-difference for each class plotted against the mean selectivity of unit that were most selective for that class. I see that although adding a predictive feature increases a class' selectivity (e.g., classes appears higher on the *y*-axis); the predictive feature does not have any meaningful effect on the difference in within and between class similarity (e.g., classes do not move to the right along the *x*-axis). In other words, cosine-similarity in not sensitive to the predictive features. It is only as a small change to the datasets, insofar as it involves only 40 out of 784 pixels, yet the models utilize the feature, which impacts training time, selectivity and density.



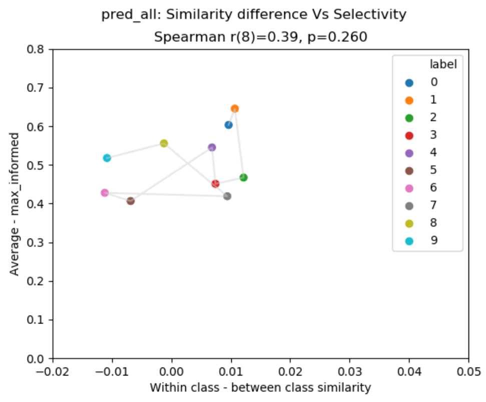
a) pred 2



b) pred 5



c) pred 8



d) pred all

Figure 4.15: **Similarity-difference and mean selectivity for classes in Experiment 4** The *x*-axis shows the similarity-difference per class. The *y*-axis shows the mean selectivity of all units where the given class is most selective. Datasets are joined in order with a grey line.

In figure 4.16, I plot the similarity difference against the mean selectivity for all datasets in this chapter. Similarity difference scores are the based on the dataset averages, and not on a per-class basis. There is a strong positive correlation ($r(15) = .94$ , $p < .001$), which means that increases in the similarity-difference are associated with increased average selectivity for that dataset. The generated datasets used in experiments 1 and 2 have the highest mean similarity-difference and mean selectivity and they appear at the top and right of the figure. The fact that these datasets are spread across the $x$-axis shows that my data generation process allowed us to move the datasets around in 'similarity-space'. However, despite the fact that there were large differences between these datasets in terms of their mean similarity-difference, the differences between these datasets in terms of selectivity are relatively small. On the other hand, the MNIST-based datasets used in experiments 3 and 4 have much lower mean similarity-difference and slightly lower selectivity and are grouped in the lower left. I can see that the manipulations made to MNIST (e.g., shuffling classes or adding an invariant cue) had a very small impact on similarity-difference, but a large impact on selectivity. Whilst I find that similarity-difference is a good predictor of selectivity in these experiments, the low variance for MNIST-based dataset could suggest that similarity-difference does not scale well and alternative methods for measuring similarity are more appropriate for predicting the selectivity for 'big-data'.

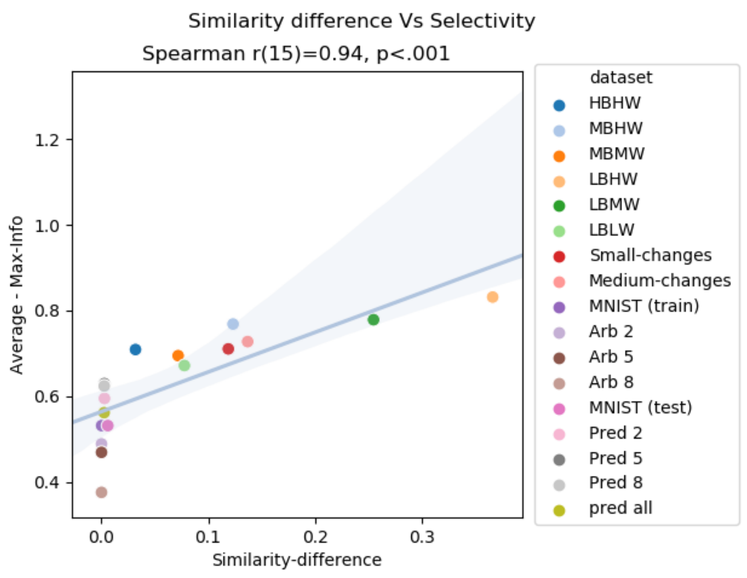
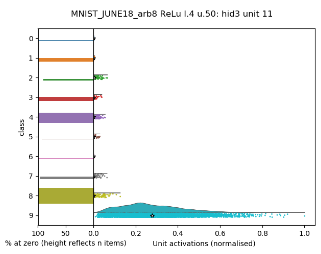


Figure 4.16: **Plot showing the relationship between the similarity-difference and selectivity for all datasets in this chapter**. The $x$-axis shows the similarity-difference (within-class-similarity minus between-class similarity). The mean maximum informedness is shown on the $y$-axis. Position on the $y$-axis is the average of the mean selectivity for each model trained on a particular dataset. Spearman correlation $r(15) = .94$ , $p < .001$.

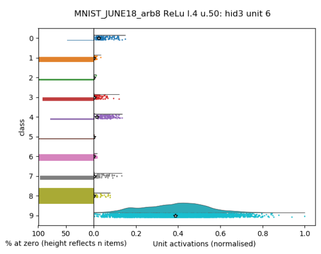
#### 4.12.3.4 Units of Interest

Figure 4.17 shows the three units with the highest selectivity for experiments 3 and 4. In experiment 3, the most selective units are all from arb 8, and are most selective for class 8 or 9, which were the classes that were not shuffled. In other words, the most selective units were for the most systematic classes. The units have very strong selectivity, with max-info scores of

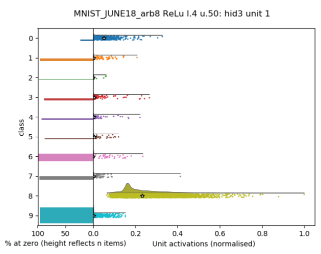
.98 or .99, yet they are not linearly separable. However, in Experiment 4 there were 12 units that had a max-info score of 1.0 (e.g., were linearly separable). As such, max-info is at ceiling, however, the selectivity of these 12 units can be well quantified with B-sel - which gives the minimal normalised distance between the classes *when they are linearly separable*. In these units, the most selective class is a class where the predictive feature has been added, again the most systematic classes have the highest selectivity.



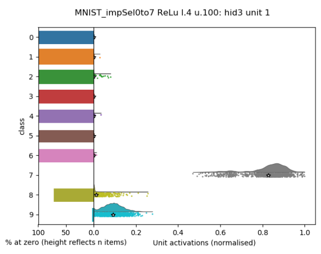
a) Max-info = .99



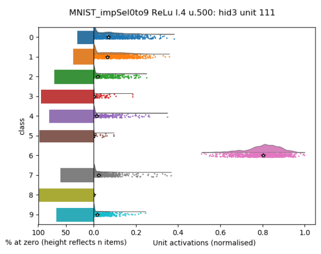
b) Max-info = .98



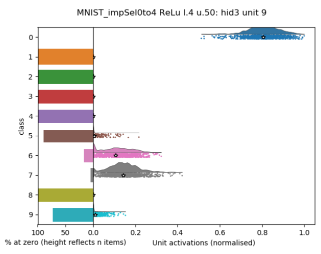
c) Max-info = .99



d) Max-info = 1.0
B-sel = .19



e) Max-info = 1.0
B-sel = .13



f) Max-info = 1.0
B-sel = .09

Figure 4.17: **Visualisation of units from experiments 3 and 4** *Top row*: Units with the highest selectivity for experiment 3. a, b & c) All three are from models with 50 units and the arb 8 dataset, most selective for classes 8 or 9, which has not been shuffled. *Bottom row*: Unit with highest selectivity from experiment 4. All units are most selective for a class which has had the predictive feature added. B-sel gives the minimal normalised distance between the most selective class and all other classes. d) from 100 unit model with pred 8 dataset, most selective for class 7. e) from a 500 unit model with pred all dataset, most selective for class 6. f) From a 50 unit model with pred 5 dataset, most selective for class 0. There were a total of 12 units out of 30231 that were linearly-separable (e.g., max-info = 1.0 or B-sel > 0).

## 4.12.4 Discussion

In this experiment I find that increasing the systematicity of a class leads to increased selectivity. However, I find that increasing the systematicity of multiple classes only increases the dataset's mean selectivity up to a certain point. Datasets with the highest selectivity had predictive features for 5 classes; selectivity was lower for datasets with features on 8 or all classes, but still higher than the unmodified dataset. This appears to be inconsistent with the findings from experiment 3, where selectivity progressively decreased as more classes were shuffled. However, in experiment 3, the principle that *items from the same class will have similar inputs* was only true for unshuffled classes. This meant that the more systematic classes dominated the arbitrary classes in terms of the frequency with which they were most selective. Here in experiment 4, this principle applied to all classes, whilst adding the invariant feature increased the systematicity, classes without the feature still had internal structure. In other words, the manipulation applied to classes in experiment 4 (e.g., a predictive feature) was far more subtle than the manipulation used in experiment 3 (e.g., shuffling classes).

In summary, experiment 4 found that selectivity was higher for datasets where the systematicity of classes was increased compared to the unmodified dataset. However, it is unclear why highest mean selectivity was for pred 5 rather than for pred all.

## 4.13 General Discussion

In a series of four experiments I investigated the effects of arbitrariness on selectivity. In experiment 1 I used a semi-random data generation process to construct datasets with different levels of mean within and between cosine-similarity. I found that arbitrariness predicts selectivity, with higher systematicity associated with higher selectivity and and high arbitrariness associated with low selectivity. I also found similar effects of arbitrariness on task difficulty (e.g., training time) and density. That is, the most systematic tasks trained for less time and had more dense representations, whilst the most arbitrary took longer to train and had sparser representations. This finding is in contrast to Vankov and Bowers (2017), where they found no effect of arbitrariness on selectivity.

In experiment 2 I investigated the effects of differing levels of within and between-class similarity for classes in the same dataset. I find that the most systematic classes are the most selective class with high frequency. The levels of selectivity and density are higher in these units than for units that are most selective for a more arbitrary class. I also show that it is not necessary for each class to have at least one unit where it is most selective. This highlights the fact that although a unit might be most selective for a particular class, it still can represent information about items from other classes. In other words, the statement that 'a unit is selective for class-$x$', should not be taken to imply any exclusivity in representing information about class-$x$, rather that class-$x$ has higher selectivity than other classes. Units which are most selective for a systematic class are denser than units that are most selective for an arbitrary class. Therefore, it is in these dense units where important information about many items from arbitrary classes is represented.

In experiment 3 I move from randomly generated data to MNIST, a well-studied digit recognition dataset. I compared the selectivity of the unmodified dataset with the selectivity of more arbitrary versions where a number of classes had been shuffled together. I again found that increased arbitrariness was associated with reduced selectivity. Although the manipulation was effective, I found that similarity-difference, my measure of arbitrariness, did not capture the effects of shuffling the classes. Put differently, for the arb datasets, there was no significant association between similarity-difference and mean selectivity. As can be seen in the lower-left of figure 4.16, similarity-difference scores are almost identical for all of the datasets used in experiments 3 and 4. This could be because these datasets have considerably more items than the datasets from experiments 1 and 2. The high number of pairwise comparisons may have averaged out any meaningful difference in their within-class and between-class cosine-similarity score.

In experiment 4 I again used versions of MNIST, this time with an invariant, predictive feature added to some or all classes. Again I found that increased systematicity was associated with increased selectivity in that all manipulated datasets had higher selectivity than the unmodified MNIST. However, adding the predictive feature to progressively more classes did not result in progressively higher selectivity: the highest selectivity was for pred 5, and pred all was second lowest. The effects on density were less strong than in experiment 3. There was no difference in density between MNIST, pred or pred 5. However, density was significantly higher in pred 8 and pred all than for MNIST.

The finding that increased arbitrariness is associated with reduced levels of class-selectivity runs contrary to predictions that class-selectivity would be higher (e.g., Morcos et al., 2018; Farah and Wallace, 1992) or the finding of no effect (Vankov and Bowers, 2017). It also demonstrates that claims regarding the suitability of highly selective representations for arbitrary one-to-one tasks (e.g., Hinton et al., 1986; McClelland et al., 1995; Kumaran et al., 2016; Coltheart et al., 2001), can not necessarily be be extended to many-to-one tasks.

The story that selectivity is higher for systematic datasets is perhaps one way of saying that selectivity is highest for the *easiest* datasets. That is, datasets where each class is distinct and not easily confusable (e.g., low between-class similarity); and where there is little variation within classes (e.g., high within-class similarity). Empirically, I see that in the higher accuracy and lower training time for systematic data. In chapter 3, the dataset with the highest selectivity was the Iris dataset, which is small, both in terms of number of items (150) and number of classes (3). It might therefore be tempting to suggest that it is the small size which makes the task simple to solve and have high selectivity. Here I have changed the levels of selectivity associated with a dataset (MNIST) by changing the systematicity, and without changing the number of items or classes. It therefore could be the systematicity of the Iris dataset, rather than (or as well as ) its size, which is a key factor.

The findings from these studies also address the question of why some classes of a dataset have higher selectivity (or more highly selective units) than others. My results suggest that the most systematic classes will be more selective and have more selective units, whilst the most arbitrary classes will be least selective. Many studies have reported finding highly selective units for a subset of features or classes (e.g., Bowers et al., 2014, 2016; Radford et al., 2017;

Berkeley et al., 1995; Vankov and Bowers, 2017; Karpathy et al., 2016; Zhou et al., 2017). However, arbitrariness is unlikely to the only factor. For example, the frequency with which an item appears in the dataset might affect selectivity. Famously, models trained on imageNet have many units which are highly selective for dogs or associated features (e.g., parts of dogs, see Zhou et al., 2015, 2017; Olah et al., 2020). In imageNet, 200 of the 1000 classes are breeds of dogs, which makes the dataset a good test of a model's fine-grained classification abilities. The prevalence of dogs in the dataset could contribute to their high selectivity (Zhou et al., 2017). A different interpretation is that selectivity is not driven by the prevalence of dog units, but rather, the importance of dog features to many of the output classes. An illustrative example comes from Karpathy et al. (2016). They trained recurrent neural networks to predict the next character which were trained either on a novel (War and Peace) or on computer code (the Linux kernel). They found units with high sensitivity to 'if-statements' in models trained on computer code, but not the novel. The word 'if' is an operator in the computer code, with specific syntax associated with it (e.g., 'if is typically preceded by an open ‘', a new line and an indentation. If statements are followed by a new line, an indentation and a ‘'). In the novel, the word 'if' is less predictive of a specific sequence of characters, and as such there was no dedicated unit for this feature.

In conclusion, these experiments provide robust evidence demonstrating that systematic datasets or classes, will tend to have higher selectivity than arbitrary datasets or classes.

# Chapter 5

# Recurrent models of Short-term memory

## 5.1 Introduction

This chapter investigates the selectivity of hidden units in models trained on a short-term memory task. Bowers et al. (2014) found that recurrent neural networks (RNNs) learn a greater number of highly selective units in circumstances where this helps them overcome the 'superposition-catastrophe'; caused by multiple overlapping distributed representations forming ambiguous patterns. Bowers et al. (2014) claimed that their findings showed that that pressure to avoid the superposition-catastrophe meant their models used localist representations, suggesting that distributed representations are only suitable in limited circumstances. In this chapter I fail to replicate the finding from Bowers et al. (2014) that RNNs increase their use of highly selective units to ensure they avoid the superposition-catastrophe.

### 5.1.1 The Superposition Catastrophe

One factor that might affect the selectivity of units in a distributed representation is the pressure to avoid the *superposition-catastrophe* (Rosenblatt, 1961; Von Der Malsburg, 1986). This refers to the fact that simultaneously activating multiple distributed representations can lead to an ambiguous blend, from which individual representations can not be easily disentangled. Rosenblatt (1961) gives the example of a neural network with four hidden units with the following representation system: if a triangle is present, activate the first unit; if a square is present, activate the second unit; if the shape is in the upper visual field, activate the third unit; if the shape is in the lower visual field, activate the fourth unit. Using this system the model can represent the identity and location of a single shape. However, if all four units are activated, there is a superposition-catastrophe as there is no way to know whether there is a triangle at the top and a square at the bottom, or a square at the top and a triangle at the bottom.

### 5.1.2 Bowers, Vankov, Damian and Davis, 2014

Bowers et al. (2014) demonstrated a task in which RNNs increased the number of highly selective units as a way to avoid the *superposition-catastrophe*. The models they trained were based on a model of short-term memory by Botvinick and Plaut (2006), which featured a single hidden layer of simple recurrent units. In Bowers et al. (2014), models were presented with a list of words, one item at a time, and following the last item, the model had to simultaneously activate the output units corresponding to each word. The items were three letter words, composed from an 'alphabet' of 30 letters: 10 onset consonants; 10 'vowels' (including 'y, aa, ea, ou, oo', etc.) and 10 coda consonants. From the possible 1000 available words, a vocabulary of 30 words was used in simulation 1, and a vocabulary of 300 words were chosen for simulation 2.

These words were presented to the model as binary vectors, across 31 input units. Thirty of the units were used to represent the input data, and the thirty-first unit was activated at the end of each sequence as a cue to produce the output. There were three conditions for the format of the input data used to train each model in simulation 1, which varied in how dense the input representation were. For *local-words* input-coding, each of the 30 words was represented by a single active unit out of thirty units. For *local-letters*, each of the thirty letters was represented by its own dedicated unit, so each word was represented by the activation of three input units. For *distributed-letters* each letter was represented by the activation of three units, meaning 9 units were active for each word. With this coding there was high overlap between different input patterns, which may share units even if the words share no letters. The amount of overlap in input representations may increase the likelihood of a superposition-catastrophe occurring.

All models had a single hidden layer with 200 simple recurrent units using a sigmoid activation function. The output layer had a dedicated unit for each item in the vocabulary (e.g., 30 output units in simulation 1 and 300 in simulation 2). Models were run on LENS (Rohde, 1999) and were trained with a backpropagation through time algorithm and a variant of stochastic-gradient-descent (SGD) specific to LENS with a learning rate of 0.01 and momentum of 0.9. Models were trained exclusively on lists of either 1, 3, 5 or 7 words.

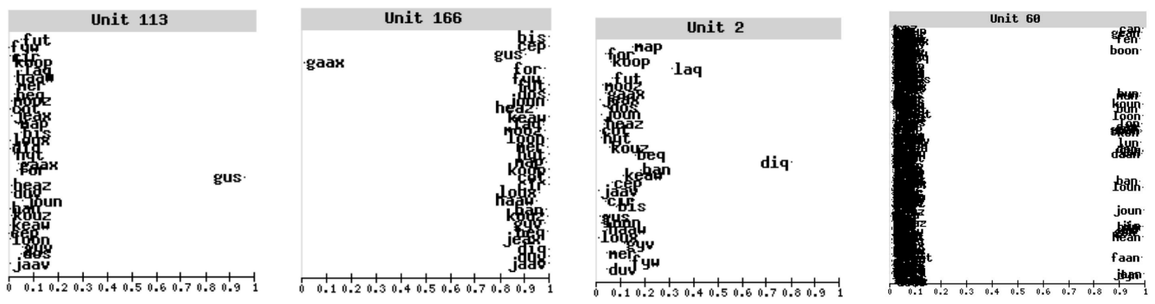
All models in simulation 1 reached 100% accuracy from one million training trials. In simulation 2, models were trained for 20 million trials, with final accuracy for lists of 1, 3, 5 and 7 words of 100%, 95%, 80% and 20%, respectively. Based on the final accuracy and the time taken to reach their highest accuracy, models found the local-words input scheme easiest and distributed-letters hardest; models found recalling short lists easier than recalling long lists; and the smaller vocabulary was easier than the larger vocabulary.

They performed selectivity analysis on one model from each condition in simulation 1 and 2. Analysis of trained models was performed by recording the hidden activations in response to single items (e.g., lists of length: one) for all models. The analysis looked to identify units that were selective for a single *word* in simulation 1 and selective for an individual *letter* in simulation 2. Bowers et al. (2014) proposed a selectivity measure (B-sel), which gives the minimal distance between a given item or feature (e.g., word or letter), and all other items. For example, when testing for item-selectivity if the most active word has activation of .8 and the second most active word was at .25, then the B-sel would be .55 (e.g., .8 - .25). When testing for feature-selectivity in experiment 2 each letter appeared in 30 words, and if the highest activation for a word containing 'b' was at .4, and the lowest activation for a word *not* containing 'b' was at .7, then the B-sel would be -.3 (e.g., .4 - .7). They considered a unit to be 'localist' (or a 'local code'; e.g., highly selective) if the *absolute* B-sel was greater than .5.

In simulation 1, Bowers et al. (2014) identified many word-selective localist units (for examples see panels a and b in figure 5.1). They found that use of local units increased with task difficulty. That is, there were fewest local codes for the local-words input scheme and most for distributed-letters input scheme, and for all input schemes, the number of local codes increased with list length. This is consistent with the claim that local units emerge to help the model avoid the *superposition-catastrophe*. Of the 100 models trained on lists of seven words with a *distributed-letter* input (the condition with the most local codes), the average number of words that were selectively coded by individual units was 21 out of 30 words.

In simulation 2, they increased the vocabulary size to 300 items, and all models were trained with *distributed-letters* input. Models could not avoid the superposition-catastrophe by relying on word-selective units, as there were more words than units. Indeed, they found no word-selective units, but in models trained on lists of three, five or seven words they found many letter-selective localist units (for examples see panel d in figure 5.1). Across five runs for each condition they found no *letter-units* for models trained on single words; an average of six units for models trained on lists of 3 words; and 28 units for models trained on lists of five and seven words.

Bowers et al. (2014) suggested that highly selective units emerge for the sake of avoiding the *superposition-catastrophe*, with a greater number of highly selective units found when there was the increased chance of ambiguous, overlapping representations. The three variable (input coding scheme, vocabulary size and list-length) all contributed to the superposition-constraints in different ways. The different input-coding schemes made words more or less distinct (i.e., confusable). It is easier for the hidden layer to construct distinct representations for each word when the inputs do not overlap. Where the input codes overlap, the hidden layer representations may also overlap, increasing the chance that multiple representations will form an ambiguous blend. The vocabulary size relates to the number of unique items that the model needs to represent across 200 units. In the smaller vocabulary, it is easier for the model to find representations that are distinct. For the larger vocabulary, there is increased chance that the representation will overlap, increasing the chance that representing multiple items will result in an ambiguous blend. They found that models succeeded in the smaller vocabulary by using word-selective units. For the larger vocabulary the model learned letter-selective representations. This is consistent with



a) b) c) d)

Figure 5.1: **Jitterplots of units from Bowers et al. (2014)**. The level of activation for a word is shown by its position along the x-axis. The position along the y-axis is pseudo-random to keep items visible. a) Unit 113 from simulation 1 with selectivity of .9 for the word ‘gus’. b) unit 166 from simulation 1 with selectivity of -.89 for the word ‘gaax’. c) unit 2 from simulation 1 with selectivity of .49 for the word ‘diq’. This unit is not considered to be a localist unit as the selectivity is below .5. d) Unit 60 from simulation 2 with selectivity of .8 to the letter ‘n’. Figure taken from Bowers et al. (2014)

Plaut and McClelland (2000), who claimed that the type of representations learned by a model emerge in response to task demands. The variable that most directly relates to the superposition-catastrophe is list-length: the number of items that must be represented before recall. For lists of one item, there is no superposition-constraint, representations will not form an ambiguous blend as a single representations is not blended with anything. But where models have to recall multiple items, it has to disambiguate these items from the blended pattern caused by their co-activation. As such, these blended patterns must be distinct. The superposition-catastrophe occurs if the combined representation of two or more items result in a blended pattern of activation that is indistinguishable from the blended pattern caused by a different list of items. There are over 24000 permutations of three words from the vocabulary of 30 words. To avoid the superposition-catastrophe, the model must ensure that none of those blended permutations can be confused with a different blended permutation of three words. For lists of five words from vocab 30 there are 17 million permutations and for seven words there are billions of possible permutations, greatly increasing the chance that a blended pattern of seven words will be indistinguishable from the blended pattern of a different seven words.

In Bowers et al. (2014), models were trained to recall lists of 1, 3, 5 or 7 words, yet they were only analysed on lists of *one* word. This is equivalent to only analysing responses to the first item in longer lists. Whilst this does not impact on the claim that the proportion of selective units increased in response to superposition constraints, it is relevant to how they describe these units. For example, in describing a unit that is highly selective (at the first timestep) for words containing the letter ‘n’, they state: “this unit appears to be a localist detector for the letter $n$." (Bowers et al., 2014, pg. 265). In only analysing the responses to single items, they are unable to show that the unit codes for ’one thing’ (e.g., high selectivity for the letter ‘n’ across all list positions), which is a defining feature of localist representations according to Bowers (2009). Such a unit could plausibly be selective for letter 'n' at the first timestep, but have low selectivity or even be selective for a different letter at other timesteps.

Another limitation of Bowers et al. (2014) is that accuracy was low in the critical conditions where the superposition constraint was highest. In simulation 2, for models trained on seven words from the 300 word vocabulary, accuracy was around only 20%, meaning that many of the items in the analysis were incorrectly classified. Bowers et al. (2014) suggests that in these conditions, models still suffer from the superposition-catastrophe, even with these highly selective units, resulting in their low accuracy.

In Bowers et al. (2014) they contrast their 'localist' units with distributed-representations, as such they claim that their models learn both distributed-representations and localist representations. For example: "Nevertheless, the results highlight the pressure to learn localist codes in response to the superposition constraint and indicate that localist letters provided a better solution than purely distributed coding" (pg.257, Bowers et al., 2014). However, Bowers et al. (2014) do not give details of how a model with mixed localist and distributed coding would work. For example, in figure 5d of Bowers et al. (2014) they show the word-selective localist units for a model trained to recall lists seven words from simulation 1. There are two localist units each for the words 'jeax' and 'joun' (similarly, figure 9d shows two localist units for the letter 'a'). Whilst localist models and 'grandmother cell' theories do allow for redundant representation or clones (e.g., Davis, 2001; Bowers, 2009), the different B-sel scores for these pairs of localist units show that they are not identical copies of each other. This leaves open the possibility that that two 'jeax' units form part of a distributed representation for the word 'jeax'. Even where there is only a single localist unit for a word or letter, Bowers et al. (2014) do not show that this highly selective unit is not part of a distributed representation. Bowers himself has argued that a highly-selective unit is not localist (or a 'grandmother cell') if it acts as part of a distributed-representation, (e.g., pg 93, discussion of neuron 2, Bowers, 2011). One way to show that these selective units are *not* part of a distributed representation would be through a lesioning study.

In a subsequent paper, Bowers et al. (2016) ran a similar study and lesioning the most selective units. This study investigated the ability of RNNs trained to recall lists of words to generalize learning to novel items. This study used a similar dataset to Bowers et al. (2014), all input data used *local-letter* coding (described above), but with distributed output representations which also used local-letter coding. When presented with a list of words, the model had to recreate the input pattern for each word, one-at-a-time, in the correct order. Models had one hidden layer of 200 simple recurrent units and were trained on all list lengths from 1 to 9 words. Selectivity was measures with B-sel, and units were described as 'localist' if the absolute selectivity was > .1. Unlike Bowers et al. (2014), models trained on the small vocabulary did not learn any localist units (i.e., for words or letters), demonstrating that highly selective units were not necessary to avoid the superposition-catastrophe. Models trained on the large vocabulary had 33 letter-selective localist units. Generalization performance was much better in models trained on the large vocabulary than the the small vocabulary, which they suggest was due to the use of localist units in models trained on the large vocabulary.

Bowers et al. (2016) performed lesioning on the 33 localist units and found that lesioning led to a mean drop in accuracy of 25% for items containing the letter that the unit was selective for. This partial drop in accuracy is closer to the 'graceful-degradation' associated with distributed representations (Rogers and Mcclelland, 2014), rather than the 'graceless degradation' associated with localist models (Page, 2000; Coltheart, 2017) as it suggests that other parts of the model (e.g., less selective units) contributed to the representation of the letter in question. In other words, that the highly selective units in Bowers et al. (2016) acted as part of distributed representations for the given letters. Rather than postulating two types of representation (e.g., localist and distributed), the results in Bowers et al. (2014) are consistent with the view that models increase their use of highly selective units in response to superposition-constraints; but representations remain distributed - with no qualitative change.

Across these two studies, Bowers et al. (2014, 2016) demonstrated that in some circumstances, pressure to avoid the superposition-catastrophe can lead RNNs to utilize a greater number highly selective units . Since RNNs in general are required to represent multiple pieces of information across the same units, one might suppose that pressure to avoid the superposition-catastrophe is ever-present, and that therefore, highly selective units are common, or at least more common in RNNs than in feed-forward models. But it is worth noting that the factors highlighted by Bowers et al. (2014) are all in relation to the size of the model, which was held constant across their simulations. For example, increasing the hidden-layer size may have mitigated the superposition-effects of increasing the vocabulary-size, list-length or input-density. The lack of localist units in some conditions suggests that these models only utilise highly selective units where necessary.

There might be properties of the datasets used in Bowers et al. (2014) that made this task well suited to the use of highly selective representations. In the datasets used by Bowers et al. (2014), all words and letters were highly regular. Words only varied in terms of the letters that composed them, and all instances of letters were identical with no variance. All inputs were binary, so a unit was either on or off. The only difference between the 'a' in 'bat' and in 'cap' is the surrounding letters. This might make it easier for models to learn a selective and invariant representation for the letter 'a' because there is no variation of how 'a' is presented. However, real-world data often varies along multiple dimensions, instances of the same category can differ. For example, the MNIST handwritten digits dataset, there are well formed number '9's and there are also unusual, irregular and borderline '9's that look like '4's. This information (or 'dark-knowledge', Hinton et al., 2015) can be used useful; for example, in classifying digits in the MNIST dataset, it could be useful to know that a 4 looks like a 9 more often than a 9 looks like a 4. Similarly, in the recall tasks used by Bowers et al. (2014), although there was similarity between word items in terms of their shared input pattern, there was no meaningful structure in terms of semantics or syntax. All words were equally likely to occur in all positions of the lists (although they could only occur once per list). This can be contrasted with a language modelling task where knowing the start of a phrase constrains predictions for the next item. This raises the possibility that learning highly-selective units might have come at a low cost in Bowers' studies, as models did not need to devote resources to capturing the similarity between different instances of a word; and did not need to learn the statistics of word frequency or order. Learning a selective unit for a word, or even breaking a word down to represent it across selective letter units did not sacrifice any important word-level statistics. On the other hand, perhaps in a language modelling task, the need to represent richly structured relationships would increase increases the number of things that the model needs to represents, making a superposition-catastrophe more likely and as such increase the need for selective units.

### 5.1.3 Selectivity in RNNs

Several studies have reported finding highly selective units in RNNs (e.g., Karpathy et al., 2016; Kementchedjhieva and Lopez, 2018; Qian et al., 2016; Dalvi et al., 2019). The studies reviewed here do not make explicit reference to the superposition catastrophe nor do they manipulate factors relating to superposition-pressure (e.g., list-length). However, in some studies, the features that they find selective units for are features that must be tracked across a many timesteps (Karpathy et al., 2016; Lakretz et al., 2019; Radford et al., 2017). This could be seen an analogous to Bowers et al. (2014) finding that models use more selective units to track information across long-lists than short-lists. Other units are selective for short range features, which suggest factors other than the superposition-catastrophe influence selectivity (Kementchedjhieva and Lopez, 2018; Qian et al., 2016).

Karpathy et al. (2016) investigated selectivity in RNNs that were trained to predict the next character in the Linux Kernel (i.e., computer code) or Tolstoy's War and Peace. They identify several units that selectively tracked long-range dependencies: units which maintain information across many timesteps. For example, analysis of the hidden state of LSTMs identified units that appear to 'count down' until the next new line character (trained on War and Peace); or units that are activated only when inside a quotation, parenthesis or an 'if' statement (when trained on computer code). They note that models can track whether it is inside a quote, even for long quotations (~230 timesteps), despite the fact that training only propagates back through 100 timesteps. They hypothesise that the unit is first sensitive to short range dependencies, and later generalizes to longer ones. They do not give details of how they chose which features to test for, but they note that most units are uninterpretable. This study is notable for the types of feature that they found high selectivity for: not the most frequent character in the dataset or combination of characters (e.g., 'the', 'and' etc), but long-range dependencies - correlations and associations separated by many characters. The finding of high selectivity for long-range but not short-range dependencies is consistent with Bowers et al. (2014)'s finding that selectivity is higher for models trained to recall long lists than short lists of words.

Radford et al. (2017) trained LSTMs to predict the next character in a corpus of online product reviews. Their model achieved impressive performance (e.g., 92.88% on the IMDB dataset). Reviews in their datasets were classified according to their 'sentiment' as either positive or negative. They found that the overall sentiment of the review could be predicted by observing a single unit (out of 4096). That is, by fitting a threshold to this one unit, they could predict the sentiment of a review with 92.3% accuracy (chance performance would be 50%). They demonstrated a causal role for this 'sentiment unit' on a review's sentiment: by setting this unit to a high or low value, they could generate positive or negative reviews. However, others have questioned the role played by such highly selective units. Donnelly and Roegiest (2019) replicated Radford et al. (2017)'s model and finding of a highly predictive sentiment neuron. Donnelly and Roegiest (2019) notes that the sentiment of reviews could be predicted from this unit alone with high accuracy, but that lesioning the unit only led to a small decrease in total accuracy. They performed cumulative ablation of neurons, starting from the most predictive of sentiment to the least There was a minimal drop in performance at predicting sentiment, even after lesioning 4000 neurons. This means that the least predictive 100 units represent sentiment related information equivalent to the sentiment unit alone. Training 1000 classifiers on random selections of 100 neurons typically produced higher sentiment prediction accuracy than the sentiment neuron alone. This suggests that the information represented by the sentiment neuron is also represented throughout the model. In other words, the model did not represent sentiment related information locally in one unit; instead relevant information was distributed throughout the model. Together these two studies demonstrate that whilst highly selective neurons may be sufficient to represent the information required for good performance, they are not necessary (Radford et al., 2017; Donnelly and Roegiest, 2019).

Another study that is relevant to the superposition-catastrophe comes from Lakretz et al. (2019). They studied the performance of a language model (LSTM) on a number agreement task. For example, comparing predictions for the words 'greet' and 'greets' when the number of the subject or distractor noun is varied in sentences such as "The boy(s) near the car(s) greet(s) the girl". They identified highly selective units for the long-distance tracking of number agreement between subject and verb in sentences containing a distractor. Lesioning these selective 'number agreement' units causes a significant drop in performance on these sentences, but not short-range dependencies (e.g., 'the boy(s) greet(s) the girl'). This suggests that these selective units functioned only to track long-range dependencies, with short-range dependencies handled by a populations of distributed neurons. This is consistent Bowers et al. (2014)'s findings that pressure to avoid the superposition catastrophe is increased for longer sequences than shorter ones. Lakretz et al. (2019) reported two separate units which performed the task of tracking number agreement, one which was highly active when the subject was singular, and one was active when the subject was plural. A third type of highly selective unit was identified with strong connections to these two number agreement units. These units identified embedded clauses in sentences (which are distractors in this task, separating the subject from the verb), and are referred to as 'syntax units'. Activation in these units increases following the subject and remains high until the corresponding verb is reached. Ablating them impaired performance on Number-Agreement tasks containing interfering nouns. Together these three units allow the model to track long range dependencies, and the presence of the syntax units explains how the unit is able to differentiate between long-and short-range dependencies.

Other studies have reported finding selective units relating to short-range dependencies, and as such, these findings are less consistent with predictions relating to the superposition-catastrophe. Kementchedjhieva and Lopez (2018) trained a character level language LSTM model to predict the next character from the Wikipedia corpus. Through use of correlations between characters and activations and noting the inputs which caused the highest activation per unit they reported finding 40 units out of 256 with a pattern of activation that could be described as meaningful by human researchers. Three such units are discussed in the paper: The *punctuation unit* was selectively activates for punctuation marks such as full-stops and closed brackets, and reminiscient of the punctuation units identified in Karpathy et al. (2016). However, other units appear to track short range dependencies. The *latinate unit* selectively activated by word prefixes such as 'inform', 'concentra' or 'introdu' that precede the latinate suffixes '-ion', '-ation', '-ction' and '-tion'. Finally, the *word unit* was selectively active at the end of a familiar word, which either occurs at the end of a word, or in the middle of a word containing a sub-word. For example, it is strongly active for the final letter in in the words 'paper' and 'grew'. However, for the word 'accordingly' it is strongly active at the letter 'g', then remains active until the end of the word. They note that although the model is at the character level, and is therefore 'wordless', the model learned selective units to track morpheme boundaries (in the case of the latinate and word unit). Similarly, Qian et al. (2016) reported finding several selective units in character and word LSTM language models. In word models they find units selective for sequence features (e.g., line length); lexical features (e.g., determiners, pronouns, prepositions); and compositional features (e.g., discriminating between types of prepositions). In the character models they find units selective for individual characters such as white-space, which they suggest acts as a word-boundary detector. The presence of selective units for features that are tracked over few timesteps suggests that the superposition-catastrophe is not the only factor that contributes to selectivity (Qian et al., 2016; Kementchedjhieva and Lopez, 2018).

### 5.1.4 Chapter overview

This chapter will outline a series of experiments that fail to replicate the high levels of selectivity reported in Bowers et al. (2014), which they attributed to superposition effects. I present these experiments sequentially in order to clearly show the steps taken to resolve this difference. I find that when using the parameters described by Bowers et al. (2014), models fail to converge in the critical conditions (i.e., list of seven items from the vocabulary of 300 words), where the greatest number of selective units was reported by Bowers et al. (2014). In experiments 1 and 2 I use similar models to the ones used in Bowers et al. (2014), which use Simple Recurrent units. In experiment 3 I apply long short-term memory models to the tasks. In experiment 4 I investigate the representations of untrained models, focusing on the impact of weight initialization on selectivity. I do not find any units with selectivity at the levels reported by Bowers et al. (2014) demonstrating that models can avoid the superposition-catastrophe without learning any localist units. Further more, vocabulary size and list length, two factors relating to the superposition-catastrophe, do not affect selectivity in the ways predicted by Bowers et al. (2014). I find no effect of list length on selectivity and I find that my models reduce selectivity for both words and letters when trained on the larger vocabulary. Whilst I can not explain why they saw the pattern of results that they did, I demonstrate that initializing models with large weights presents a model with more selective units at the start of training than if initializing with small weights. This suggests that solutions involving highly selective units are more easily available to models initialized with large weights than small weights.

## 5.2 Methods

### 5.2.1 Task

The models were trained to perform the same task as in Bowers et al. (2014). Each model is given a list of word vectors, one per time step. The number of items in the list is fixed for each model (e.g., models are exclusively trained on lists of 1, 3, 5 or 7 words). Note that lists of 1 is equivalent to recalling a single item and requires no recurrence. Following the final word in the list, model must simultaneously activate the output word units corresponding to the words in the list. This is a *free-recall* task, in that the model is not required to recall the order in which the items were presented. For example, if presented with the list ‘ban’, ‘bop’, ‘moor’, the model should activate output units 1, 2 and 30. The same three units should be activated for the list ‘moor’, ‘bop’, ‘ban’, since the order of presentation does not matter. In this task, lists did not contain repeated items (e.g., ‘ban’, ‘ban’, ‘bop’).

### 5.2.2 Data sets

Input data for all experiments corresponded to lists of 3 letter words. The three letter words were composed of letters from a 30 letter alphabet. Each word contained: an *onset* drawn from a list of ten consonants (b, c, d, f, g, h, j, k, l, m); a *vowel* drawn from a list of ten vowel sounds (a, e, i, o ,u, y, aa, ea, ou, oo); and a *coda* drawn from a list of ten consonants that did not include any consonants used as onsets (n, p, q, r, s, t, v, w, x, z). From the 1000 possible words (10 onsets x 10 vowels x 10 codas), I selected a small dataset of 30 words (vocab 30) and a large dataset of 300 words (vocab 300). Items were chosen for each vocabulary such that each letter occurred three times in vocab 30 and thirty times in vocab 300. All words in vocab 30 were included in vocab in 300.

All experiments use what Bowers et al. (2014) refer to as *distributed-letter* coding. In this scheme, each letter is represented by three units, so each word is therefore represented by a unique pattern of nine active units out of thirty. For example, first word in the vocabularies ('ban') is represented by a binary vector with ones in positions 1, 2, 3, 11, 12, 13, 21, 22 and 23; while the second word ('bop') is represented by ones in positions 1, 2, 3, 14, 15, 16, 22, 23, 24. In this coding scheme, there can be overlap in input-space, even for words that do not share letters. For example, the words 'ban' and 'bop' share activations in positions 1, 2, 3 which correspond to the letter 'b'; and also positions 22 and 23 caused by the overlap between the representations for 'n' and'p'. Of the three input coding schemes used in Bowers et al. (2014), this scheme was associated with the highest number of selective units.

### 5.2.3 Models

The simulations in Bowers et al. (2014) used a Elman network (Elman, 1990), where hidden units are connected to a separate bank of 'context' with a one-to-one connectivity (of weight 1.0) to each hidden unit, and then learned weights feeding back to the hidden units at the next time step. This architecture should be entirely equivalent to a simple recurrent unit network with no explicit context layer, and a single recurrent connection back to itself with a learned weight. As such I run experiments 1 and 2 on simple recurrent units.

All models were trained using Keras (Chollet and Others, 2015) and Tensorflow (Abadi et al., 2016), as opposed to LENS (Rohde, 1999) used by Bowers et al. (2014). Models consisted of 30 input units; a single hidden layer with 200 simple RNN units with a sigmoid activation function (apart from experiment 3 which used LSTMs); and a sigmoid output layer with one unit per word (e.g., 30 units for vocab 30 or 300 units for vocab 300). In (Bowers et al., 2014), the input layer included an additional unit which was used to 'cue' the end of the sequence. I had no reason to include such a unit as all models are trained on lists of a fixed length, and in Keras models automatically recall items after a complete list has been presented.

It was also suggested by Ivan Vankov (personal correspondence, 2nd December 2019.), who ran the simulations in Bowers et al. (2014) that I use the 'unrolled' setting in Keras, to align it more closely with their simulations. In unrolling these recurrent models, they are treated as feed-forward models with separate layers using weight sharing for each timestep rather than as a symbolic loop through a single hidden layer.

### 5.2.4 Training

Training data was generated on the fly, rather then from a fixed training set. This is because the number of possible lists grows very large. For example, for the vocab 30 dataset, there are 30 possible lists of one item; 24360 possible lists of three items and millions of possible lists of five or seven items. All models were trained with the 'binary-crossentropy' loss function, which is used for multi-label classification (i.e., for each list, the output should activate multiple units corresponding to all items in the list).

Whilst I aimed to closely replicate (Bowers et al., 2014), I was unable to use the exact same parameters. This was either because I found that models performed poorly when I used their settings; or because their methods have fallen out of favour for theoretical reasons.

In Bowers et al. (2014), they trained all models for a fixed number of trials. They do not describe how they decided on these numbers, but the limitations of a fixed training time are three-fold. Firstly, the fixed amount of training might be insufficient, meaning some models do not converge. Figure 7 in Bowers et al. (2014) shows that for lists of seven words the accuracy and loss were still improving at the point where training stopped. Secondly, the fixed amount of training might be too long, and as such models might overfit the training data. The final reason to avoid fixed training duration is that in the case where training reaches the ceiling early on, it wastes resources (i.e., GPU availability). As such, I will set an upper limit on the amount of training allowed for each experiment, but I will use an early stopping function, which will stop training if the loss did not improve for $\frac{\text{maximum epochs}}{50}$ epochs. Models were were set to train for a maximum of 10000 epochs in experiment 1 and 10000 or 200000 epochs in experiment 2 (e.g., for the small and large vocabulary, respectively); and 10000 epochs in experiment 3. Models in experiment 4 received no training. All epochs consisted of 100 batches.

Bowers et al. (2014) used online training, meaning that weights were updated after each list. The limitation of this approach is that the gradient for one list might not be in the correct direction for the entire dataset, as such steps might wander around, heading in the right direction on average, but taking a winding path to get there. In mini-batch training, the weights are updates after a small batch of lists (e.g., 32). This allows the model to take the average of several examples, which give a better approximation of the true gradient and can allow more efficient training (Goodfellow et al., 2016). Lists were presented in batches of 32 in experiments 1 and 3; and in batches of one ('online learning') in experiment 2.

Recently, SGD or SGD with momentum have fallen out of favour and other adaptive optimizers have become more popular. Adam (Adaptive Moment Estimate, Kingma and Ba, 2014) is one that uses an adaptive learning rate, momentum and bias correction (for past gradients) which consistently performs well and is recommended as a good default optimizer (Karpathy, 2016). A variant of Adam called AMS grad (or Adam with ams) updates using the maximum of past squared gradients rather than using the exponential average, which can improve performance over Adam on small datasets (Liu et al., 2019). The Adam optimizer was used in experiments 1 and 3, and experiment 2 used Adam with AMS grad.

For recurrent models in Keras, the recurrent activation (from the hidden layer back to itself) at the first timestep is zero (i.e., there is no activation from a 'previous' timestep. In the simulations from Bowers et al. (2014), the recurrent activation (from the context layer to the hidden layer) were initialized at .5 for the first timestep. However, I failed to get models to converge when hidden-states were reset to .5, so my experiments will use the default of resetting hidden states to zero at the start of each lists.

In Bowers et al. (2014), weights were initialized with values drawn from a random uniform distribution between -1 and 1, which I refer to as 'LENS initializer'. Initializing with large weights can cause unnecessarily large weight updates, overshooting the optimal value, then oscillating; and can lead to saturated sigmoids with poor gradients (Goodfellow et al., 2016). Other weight initialization methods scale the distribution of weights according to the size of the model (e.g., based on the number of input or output units). Keras' variance scaling initializer is a generalized version of Kaiming He's initializer (He et al., 2015). Variance scaling draws weights from a truncated normal distribution with a mean of zero and a standard deviation of $\sqrt{\frac{1}{\text{input units}}}$. Variance scaling was used in experiment 1. Keras' default weight initializer, designed to avoid exploding or vanishing gradients in sigmoids is 'Glorot uniform' (or Xiavier Uniform) initializer, which draws weights from a uniform distribution with limits at $\pm\sqrt{\frac{6}{\text{input units + output units}}}$ (Glorot and Bengio, 2010). I used variance scaling in experiment 1, LENS initializer in experiment 2; Glorot Uniform in experiment 3 and all three initializers were used in experiment 4.

(Bowers et al., 2014) used a learning rate of .01 for all their models. I conducted parameter-sweeps to find the best learning rates, which were .013 for experiment 1, .0025 for experiment 2, and .005 in experiment 3.

### 5.2.5 Recording hidden activations

In Bowers et al. (2014), selectivity analysis was carried out on one model from each condition, on the basis of model responses to each word presented (individually, rather than in lists). Here I analyse the unit activations for all models in response to a test-set of lists of the same length as were used in training. The test-sets contained 30 lists for models trained on vocab 30 and 300 lists for vocab 300. Each word appeared once in each position, and was preceded and followed by different words each time. For example, for vocab 30 lists of three, the word 'ban' appears in the lists: ['ban', 'bop', 'cis'], ['moor', 'ban', 'baaq'] and ['mow', 'maat', 'ban']. Model predictions were taken as the activation of an output 'word' unit greater than .5 (which is the midpoint for sigmoid units). For example, if the first unit in the output layer had an activation greater than .5, this was taken as a prediction that the word 'ban' had appeared in the list. Test set accuracy is reported with two metrics. The first is the intersection over union (IoU) per list. This allows for scoring in cases where the model predicts the wrong number of items. The IoU accuracy was the number of correctly predicted words, divided by the true number of words in the list plus any incorrectly guessed words. For example, for a three word list consisting of 'ban', 'bop', 'mooz': the IoU would be .66 ($\frac{2}{3}$) if only the two units corresponding to 'ban', 'bop' were active above .5 . The IoU score would be .5 ($\frac{2}{4}$) if 'ban', 'bop' and 'jin' were activated . If the model activated four units corresponding to 'ban', 'bop', 'mooz' and 'jin' the score would be .75 ($\frac{3}{4}$). The IoU scores per list were used to calculate a mean IoU score. However, the main metric used to assess model performance was the proportion of lists with IoU of 1.0 (referred to as the proportion of lists correct). This was done to avoid models which for example, consistently performed poorly on the first or last timestep. Models were only included in selectivity analysis if the proportion of correct lists was greater than .1. For models where the proportion of lists correct was greater than .1, the test set was presented again. The level of activation at each unit, at each timestep was recorded.

### 5.2.6 Selectivity analysis

Selectivity analysis was carried out separately for each word in the vocabulary and for each letter in the alphabet, for all units in the model at each timestep. All models have 200 hidden units, but since each unit has separate analysis for each timestep, it means that the total number of *unit/timestep* analyses for models trained on lists of 1, 3, 5, and 7 words is 200, 600, 1000 and 1400 respectively.

Units were tested for both ‘on’ (e.g., the word or letters has higher activation than all other items) and ‘off’ (e.g., the word or letters has lower activation than all other items) selectivity using a measure based on the one used in Bowers et al. (2014), which I shall refer to as ‘Bowers’ Selectivity’ or ‘B-sel’. Bowers’ Selectivity gives the minimal differences in activations between one word (or letter) and all other words (or letters). For details of how B-sel is calculated see 3.3.1 in chapter 3.

All B-sel scores are in the range 0 to 1. (Note: this is slightly different to (Bowers et al., 2014), where ‘off’ units were scored with negative values). If the word or feature being analysed is not linearly-separable, the B-sel score is 0.

For each word or letter, the B-sel score is the maximum value of the selectively for on and off scores. For each timestep, the unit is said to be selective for whichever word has the highest B-sel, or no word if they are all zero. For example, unit 1 at timestep 1 is said to be ‘selective for the word ‘ban’ with a score of .1’, it means that this word had the highest selectivity score of all words at this unit at that timestep. Each unit and timestep is associated with a selectivity score for for both words and letters. This means that a unit can be both selective for a word (e.g., ‘ban’) and also a letter (e.g., ‘b’).

In Bowers et al. (2014), they summarised a trained model’s selectivity by reporting how many ‘localist units’ were in a trained model, defined as units with an absolute B-sel greater than .5. However, this threshold of .5 could be seen as stringent, and exclude units that appear to be selective for a given class. In these experiments I also adopt this approach of reporting the number of units with selectivity above some threshold. However, studies using B-sel in non-recurrent models found very few units with B-sel scores > .5 (e.g., see chapter 3); which suggested that a more liberal threshold might be appropriate. For example, one could consider a unit to be highly-selective if all items in a class are linearly separable from all other items (i.e., they have a B-sel > 0). For letter-selectivity, this would require than all instances of words containing a given letter have higher activation than all instances of words than do not contain the letter. However, this might be impractical for word-selectivity, because there is no variation between instances of each word, so all instances of a word are likely to produce the same activation. For word-selectivity, I therefore set the liberal threshold of .1 for a unit to be considered to be selective. A unit is defined as selective for a letter if the B-sel is > 0. I will also report how many units have a B-sel > .5 if and when the occur. Where I identify a unit as being *selective* for a letter or word, I am not claiming that it is a localist unit, or that there is any qualitative difference between a highly selective unit and a unit with low selectivity. Rather I adopt the method of setting a threshold and counting units that cross this threshold for the sake of replicating their methods and analysis.

In keeping with Bowers et al. (2014), I report the average number of selective units at the first timestep (reported as 'ts 1') and also the average number of selective units per-timestep (reported as 'ts ave'). This is the total number of selective units across all timesteps, divided by the number of timesteps. To investigate whether these units are invariant to timesteps, I record whether a unit that selective for $x$ at the first timestep is also selective for $x$ at all other timesteps. These are reported as the number of invariant units (reported as 'invar').

### 5.2.7 Compare groups

Difference in selectivity between conditions were tested with Kruskal–Wallis one-way analysis of variance (Kruskal and Wallis, 1952). This is a non-parametric test of whether samples are drawn from the same underlying distribution, which is suitable when the group values are not normally distributed. I ran separate Kruskal-Wallis to see if there was an effect of list length on the average number of word-selective or letter-selective units. If the test is significant, post-hoc Dunn's tests were used to identify significant differences in the median number of selective units for each list length (Dunn, 1964). Dunn's tests were also used to test if there was an effect of vocabulary size (e.g., 30 or 300 items) on the average number of word-selective or letter-selective units.

## 5.3 Experiment 1

In this first study I attempt to test Bowers et al. (2014) claim that models learn highly selective units to avoid the superposition catastrophe. I also aimed to extend their analysis: they only tested the selectivity of units on single items, whilst I will measure the selectivity of units, at all lists positions. This will show whether units perform the same role throughout the task (e.g., retain high selectivity for a given letter) or whether their selectivity is context dependent (e.g., selective for a given letter only if it occurs in the first word in a sequence).

I fail to replicate many of the key findings from Bowers et al. (2014): models contain no localist units (e.g., with selectivity > .5); there is no effect of list length; and for both vocabularies selectivity was higher for words than letters. I do however find examples of invariant units in vocab 30, with selectivity for the same feature across all timesteps. There are no invariant units in models trained on vocab 300.

### 5.3.1 Methods

In my initial attempts at replicating the Bowers' study, I used models with the Adam optimizer, a learning rate of .001 and weights were initialized with Keras' default 'Glorot-uniform' initializer. However, these models failed to converge, especially for lists of seven words from the vocabulary of 300 words. A such, I performed a parameter sweep of training hyper-parameters that would leader to greater training accuracy when the model was trained on lists of seven words from vocab 300 with distributed-letter input codes. The hyper-parameters I adjusted were the weight-initialization, optimizer and learning rate. The weight initialization setting I tried were Glorot-uniform, Glorot-normal, variance-scaling and orthogonal. The optimizers considered were SGD, Adagrad, Adam, Adam-with-AMS-grad, Adamax, Nadam, Adadelta and RMSprop. The learning rates were 0.01 , 0.001 and 0.0001. The first stage of the parameter sweep trained models for 100 epochs. Models were evaluated on the model loss, the proportion of correct sequences (e.g., where IoU = 1.0) and mean IoU. The best performing configurations used the Adam optimizer, with a learning rate of 0.01 (as used in Bowers et al., 2014)), and variance scaling initializer. In Bowers et al. (2014), they used an SGD optimizer, which performed poorly in all conditions here. In the second stage of the parameter-sweep I used the Adam optimizer, with learning rates of 0.005, 0.007, 0.009, 0.01, 0.011, 0.013, 0.015, 0.02. The best performing configurations were then compared after training for 1000 epochs. The best performing models, which I used in this experiment used variance scaling weight-initialization, the Adam optimizer with a learning rate of 0.013. This study trained models with mini-batches of 32 items. This means that weights were updated after 32 items and used the average gradient form all 32 items. Although training was generated on-the-fly, I consider an epoch to be 100 batches, meaning that there are 3200 items per epoch. The maximum number of epochs will be to 10000, which equates to 32 million trials (e.g., 1 epoch is 3200 trials). This is longer than the 1 million or 20 million trails used in Bowers et al. (2014). However, early stopping is set to activate only if the loss does not decrease for 1000 epochs, and as such, it is likely that models' loss will plateau and stop training prior to reaching the 10000 epoch limit. I ran each condition 5 times from different initializations.

### 5.3.2 Results

For training times and accuracy, see table 5.1. The average training times reflected tasks difficulty: training times were shorter for vocab 30 than for vocab 300, and were shorter for short lists than long lists. Training times were in the range 1641 to 9954 epochs, which equates to 5 million to 31 million trials. This is a higher amount of training than in Bowers et al. (2014). All models had more than 10% of lists correct. Levels of accuracy are similar to those in Bowers et al. (2014).

| Vocabulary size | List length | Epochs | Mean IoU | Proportion lists correct | Sims acc > .1 |
|---|---|---|---|---|---|
| 30 | 1 | 1641 | 1.0 | 1.0 | 5 |
| | 3 | 2500 | 1.0 | 1.0 | 5 |
| | 5 | 3243 | 1.0 | 1.0 | 5 |
| | 7 | 3892 | 1.0 | 1.0 | 5 |
| 300 | 1 | 2085 | 1.0 | 1.0 | 5 |
| | 3 | 6937 | .99 | .98 | 5 |
| | 5 | 9890 | .96 | .78 | 5 |
| | 7 | 9954 | .84 | .29 | 5 |

Table 5.1: **Experiment 1 results.** All values are the average of 5 runs from different initializations. Proportion lists correct is the proportion of lists where the IoU was 1.0. Sims acc > .1 gives the number of simulations (out of 5) where the proportion of correct lists was > .1, which were included in the analysis.

| Vocabulary | List | words ≥ .1 | | | letters ≥ 0 | | |
|---|---|---|---|---|---|---|---|
| size | length | ts 1 | ts ave | invariant | ts 1 | ts ave | invariant |
| 30 | 1 | 14.40 | 14.40 | - | 26.80 | 26.80 | - |
| | 3 | 17.20 | 18.73 | 7.40 | 30.40 | 28.67 | 11.60 |
| | 5 | 13.60 | 15.64 | 4.20 | 26.20 | 25.48 | 6.20 |
| | 7 | 17.60 | 21.94 | 5.60 | 25.20 | 26.00 | 5.80 |
| 300 | 1 | 0 | 0 | - | 0 | 0 | - |
| | 3 | 0 | 0.07 | 0 | 0 | 0 | 0 |
| | 5 | 0 | 0.24 | 0 | 0 | 0 | 0 |
| | 7 | 1.80 | 2.91 | 0 | 1.00 | 1.20 | 0 |

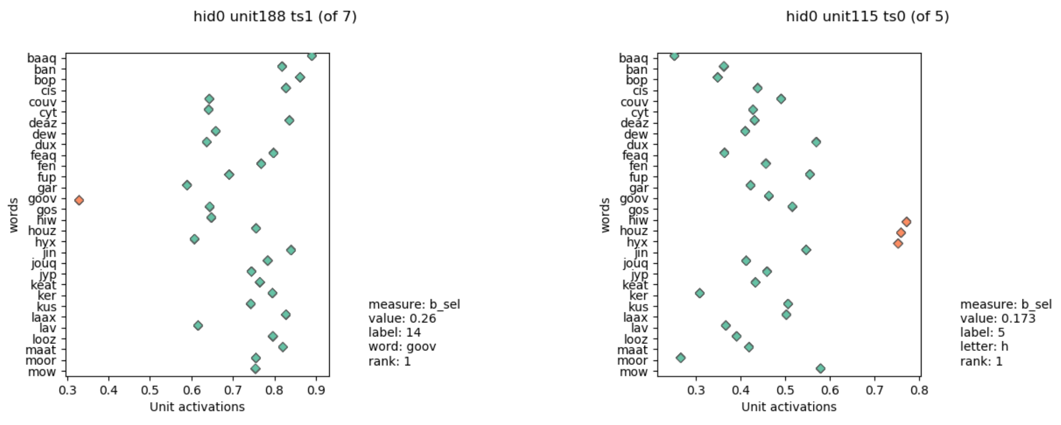
Table 5.2: **Experiment 1. A count of units with B-sel above .1 for word or 0 for letters.** ts1 denotes the number of units at the first time step only, comparable with Bowers et al. (2014). ts av is the total number of units across all time steps divided by the number of time steps. Invariant is units that are selective for the same feature across all time steps. All values are the average per 200 unit model.

For a count of selective units in each condition, see table 5.2. I found no localist units (selectivity > .5), so 'selective units' will be reported using the liberal thresholds of > 0.1 for words and > 0 for letters. The most selective unit across all models in this experiment was selective for words with a score of .26, this units was in a model trained on lists of seven words from vocab 30. The highest selectivity for letters was .17, from a model trained on lists of 5 words from vocab 30. See figure 5.2 for a visualization of these units.

There was a strong effect of vocabulary size, with more selective units in vocab 30 than in vocab 300. Dunn's test showed that there were significantly more word-selective units (e.g., B-sel > .1) in models trained on vocab 30 (median = 17.1) than vocab 300 (median = 0.1, $p < .001$). The effects was also significant for letter-selective units (v30 median = 26.0, v300 median = 0.0, p < .001). Bowers et al. (2014) only found word selective units in models trained on vocab 30, and only found letter-selective units in models trained on vocab 300. My findings are inconsistent with their pattern of results, and show that models can achieve similar accuracy on vocab 300 to Bowers et al. (2014), with very few selective units.

There was no effect of list length on the number of selective units. A Kruskal-Wallis test found no differences between models trained on lists of 1, 3, 5 or 7 items in numbers of word-selective units ($H(5) = 3.60$, $p = 0.31$) or letter-selective units ($H(5) = 0.69$, $p = 0.88$). My lack of effect is inconsistent with Bowers et al. (2014) who found that increases in list length were associated with increases in the use of localist units.

The average number of selective units per timestep (e.g., labelled ‘ts ave’ in 5.2) is similar to the average at the first timestep, suggesting that measuring selectivity at the first time step is a reliable indicator of selectivity at other timesteps. I also find that there are several ‘invariant’ units, which are selective for the same feature at all timesteps. These are found only in vocab 30, and is lower than the number of selective units at the first timestep. This shows that many units which are selective in the first timesteps, are either less selective at other timesteps or are selective for a different feature. The number of invariant units decreases with lists length. This might suggest that invariant units are less useful for long lists, or more likely, that maintaining invariant units over long sequences is more difficult than for short sequences.



a) b)

Figure 5.2: **Units with the highest selectivity for words and letters in Experiment 1** The $x$-axis shows the level of activation and the $y$-axis shows each word in the dataset. The feature that the units is selective for is shown in orange. The figure text indicates the B-sel score, labelled as ’value’, and the word or letter that the unit is selective for. Rank: 1 indicated that this was the most selective unit in that particular model. a) Unit with the highest word-selectivity, with selectivity of .26 for the word ‘goov’, from a model trained with lists of seven words from vocab 30. b) unit with the highest selectivity for a letter, with selectivity of .17 for the letter ‘h’, trained on the vocabulary of 30 items to recall five words.

### 5.3.3 Discussion

In this experiment I investigated the claim that models learn highly selective units to reduce the risk of a superposition-catastrophe. I investigated two sources of superposition-pressure identified by Bowers et al. (2014): The number of unique items the model must be capable of representing (e.g., vocabulary size) and the number of representations to be maintained before recall (e.g., list length). I trained 40 models to recall lists of 1, 3, 5 or 7 words from vocabularies of either 30 or 300 words.

These results are inconsistent with the predictions from Bowers et al. (2014). I find no localist units with selectivity > .5, although there are selective units according to the liberal thresholds of .1 for words and 0 for letters. In Bowers et al. (2014), models appeared to adopt radically different approaches to solving the small and large vocabulary: utilizing word-selective units for vocab 30 but adopting letter-selective units for vocab 300 (i.e., when there were more words than units). In my models, there is a dramatic shift in selectivity between vocabularies, but I find that numbers of both word-selective units and letter-selective units are higher for vocab 30 than in vocab 300. This means that the model was able to overcome the superposition-pressures of vocab 300 without relying on highly selective units.

I found no effect of list length. In Bowers et al. (2014), list-length provided the clearest demonstration of the pressure to avoid the superposition-catastrophe. The use of highly selective units increased with list-length, suggesting that as the model had to disambiguate an increasing number of representations from the hidden layer, it increased the selectivity of the representations. Further, they note that there is no pressure to avoid the superposition-catastrophe for lists of one item, and they rarely found localist codes in these models. I find no significant difference in how many selective units are found in models trained to recall multiple items (e.g., lists of 3, 5 or 7 items) compared to lists of one item. This suggests that the emergence of selective units can not be attributed to superposition-constraints, and I must speculate that these selective units emerged for some other reason.

I find that around a third of all selective units are ‘invariant’, e.g., selective for the same feature across all timesteps. However, this means that most of the selective units at the first timestep will either have lower selectivity at other timesteps, or become selective for a different feature. In Bowers et al. (2014) they only analysed the selectivity of units at the first timestep, but suggested that the selective units they identified could be ‘detectors’ for a feature (e.g., a localist detector for the letter n). My results suggest that most of these units do not only code for one thing as would be expected from localist units. However, my results are very different to theirs, so the tuning of units in my model might not be a reliable indicator of units in their models.

In general my pattern of results is very different to that found in Bowers et al. (2014). This can be attributed to differences in the models and training between my experiment and theirs. According to I. Vankov (personal communication, 29 November and 11 December, 2019), their findings might be dependent on the optimizer and weight initialization settings, which were not discussed in the paper. The optimizer they had used was called "Doug's Momentum', a variant of SGD with momentum specific to the LENS (Rohde, 1999), the program they used to run their simulations. Their models were initialized with weights from a random uniform distribution between -1 and 1, which I shall refer to as the LENS initializer. Bowers' models also differed in how they treated the hidden-state of units at the first timestep and in the batch-size. The following experiment will attempt to incorporate these changes into my models and training.

## 5.4 Experiment 2

In the previous experiment, there were differences between my models and those used in Bowers et al. (2014). Suggestions from I. Vankov (personal correspondence, 29 November and 11 December, 2019) led us to explore different configurations that would be closer to their original study.

In Bowers et al. (2014), the optimizer was 'Doug's Momentum'; a variant of SGD developed by the writer of the LENS program (Rohde, 1999). The momentum term means that a weighted average of the previous gradients is added to the current gradient to stop the model from oscillating around the edges of a local minima. Doug's Momentum clips the pre-momentum weight change if it exceeds 1.0.

Another difference relates to how the weights were initialized at the start of training. In Bowers et al. (2014), models were initialized with LENS initializer where weights are drawn from a random uniform distribution between -1 and 1. These initial weights are significantly larger than those in the previous experiment where I used Keras' 'variance scaling' initializer. Using large initial weights may cause units to saturate, with no gradient to guide learning, but large weights will have strong symmetry-breaking effect (Goodfellow et al., 2016).

Bowers et al. (2014)'s models and mine in experiment 1 differed in how they handle recurrent activation at the start of each list. Simple Recurrent units have a hidden state, which represents the unit's activation at previous timesteps. In my models, this hidden state was reset to zero for the start of each list. In Bowers et al. (2014), the hidden state was reset to .5 for the start of each list. I shall refer to this as using 'LENS states'. Lens states might have led their model to behave differently for the first time in each list compared to my models in the previous experiment.

Finally, in Bowers' models, weights were updated after each item (known as online learning), rather than after a mini-batch of 32 items as I used in the previous experiment. This experiment will use online learning.

The following experiment will attempt to more closely replicate Bowers et al. (2014) by incorporating the model and training parameters similar to those in LENS. However, I found that models that were most similar to Bowers et al. (2014) (e.g., using online-learning, Doug's Momentum, LENS initializer and hidden states reset to .5) would not converge, and so could not be used. I was able to train models with online-learning and LENS initialization. In these models

I see a stronger effect of lists length on selectivity than the previous experiment.

##### 5.4.0.1 Methods

I conducted a parameter sweep to find suitable training parameters for this experiment. For this, models were trained only on lists of seven items from the vocabulary of 300 items. In the first round, models were trained for 1000 epochs using Doug's Momentum and learning rates of 0.0001, 0.0005, 0.001, 0.005 or 0.01. I trained version with and without LENS states giving us ten models in total. None of the models converged, the mean IoU was 0 for all models. In the second round I repeated this process with the maximum number of epochs set to 10000, and still the models would not converge. As such, I dropped Doug's momentum. In the third round I trained models with four different optimizers: SGD, RMSprop, ADAM and Adam-ams-grad. SGD is the most similar to "Doug's Momentum". The learning rates were 0.0001, 0.0005, 0.001, 0.005 or 0.01. None of these models used LENS states. Models were trained for a maximum of 10000 epochs. None of the models using SGD or RMSprop converged. The best performing models from this round used ADAM or Adam-ams-grad with learning rates of 0.001, 0.005 or 0.01. In the final round I trained models for 10000 epochs using ADAM or Adam-ams-grad with learning rates of 0.0005, 0.00075, 0.001, 0.0025, 0.005, 0.0075, 0.01 or 0.025. None of these models used LENS states. The condition with the lowest loss and highest accuracy used Adam-ams-grad optimizer with a learning rate of .0025. All models for experiment 2 will use the LENS initializer, online learning, Adam-ams-grad optimizer with a learning rate of .0025.

For vocab 30, models were trained for a maximum of 10000 epochs, for vocab 300 the maximum number of epochs was 200000. An epoch consists of 100 trials (e.g., lists), so my maximum training times are now equivalent to Bowers et al. (2014) with 1 million trails for vocab 30 and 20 million for the large. However, early stopping will halt training if loss does not improve for $\frac{\text{maximum epochs}}{50}$.

I trained models using these setting with and without LENS states (e.g., hidden states reset to .5). However, models with LENS states failed to converge, for example, they had a mean IoU of .03 for lists of seven items from a vocab of 300. As such, all models have the hidden states rest to zero at the start of each list. I did 5 runs of each condition from different initializations.

### 5.4.1 Results

For training details from Experiment 2, see table 5.3. Accuracy is slightly higher for this experiment than the previous one, all models had accuracy above .3.

For selectivity results, see table 5.4. There is a strong effect of vocabulary size, with more selective units for vocab 30 than vocab 300. Dunn's tests showed that this effect was significant for both word-selective units (v30 median = 39.9, v300 median = 2.8, $p < .001$); and letter-selective units (v30 median = 26.83, v300 median = 0.0, $p < .001$). The finding of more word-selective units in vocab 30 than vocab 300, is generally consistent with Bowers et al. (2014), although they *only* found word-selective units in models trained on vocab 30. However, I also find more

| Vocabulary size | List length | Epochs | Mean IoU | Proportion lists correct | Sims acc > .1 |
|---|---|---|---|---|---|
| 30 | 1 | 471 | 1.0 | 1.0 | 5 |
| | 3 | 878 | 1.0 | 1.0 | 5 |
| | 5 | 1179 | 1.0 | 1.0 | 5 |
| | 7 | 1455 | 1.0 | 1.0 | 5 |
| 300 | 1 | 6052 | .99 | .99 | 5 |
| | 3 | 14797 | .98 | .95 | 5 |
| | 5 | 24305 | .97 | .87 | 5 |
| | 7 | 23343 | .86 | .36 | 5 |

Table 5.3: **Experiment 2 results** Hidden states reset to zero at the start of each sequence. All values are the average of 5 runs from different initializations. Proportion lists correct is the proportion of lists where the IoU was 1.0. Sims acc > .1 gives the number of simulations (out of 5) where the proportion of correct lists was > .1, which were included in the analysis.

| Vocabulary | List | words ≥ .1 | | | letters ≥ 0 | | |
|---|---|---|---|---|---|---|---|
| size | length | ts 1 | ts ave | invariant | ts 1 | ts ave | invariant |
| 30 | 1 | 39.00 | 40.40 | - | 27.00 | 27.00 | - |
| | 3 | 37.80 | 43.00 | 17.80 | 28.00 | 26.93 | 13.80 |
| | 5 | 42.60 | 43.88 | 16.00 | 24.60 | 24.60 | 8.80 |
| | 7 | 39.40 | 44.94 | 12.80 | 28.80 | 28.20 | 9.20 |
| 300 | 1 | 1.40 | 1.40 | - | 0 | 0 | - |
| | 3 | 2.20 | 2.87 | 1.00 | 0 | 0 | 0 |
| | 5 | 1.20 | 2.36 | 0 | 0 | 0 | 0 |
| | 7 | 5.60 | 8.00 | 0 | 0 | 0.11 | 0 |

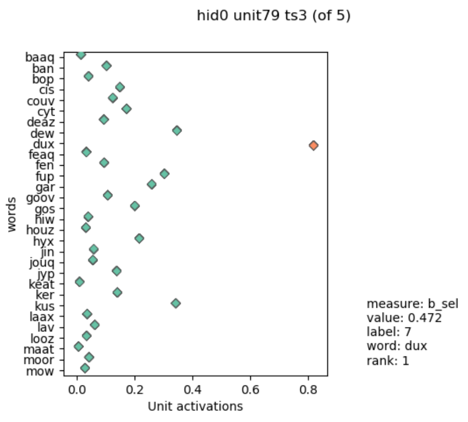
Table 5.4: **Count of units with selectivity for words (above .1) and for letter (above 0.0) from Experiment 2.** *ts1* denotes the count of selective units at the first time step only, comparable with values from Bowers et al. (2014). *ts ave* is average number of selective units per timestep. ‘Invariant’ gives the count of units that are selective for the same feature across all time steps. All values are the average per 200 unit model.

letter-selective units for models trained on vocab 30, whilst Bowers et al. (2014) only found letter-selective units in models trained on vocab 300.

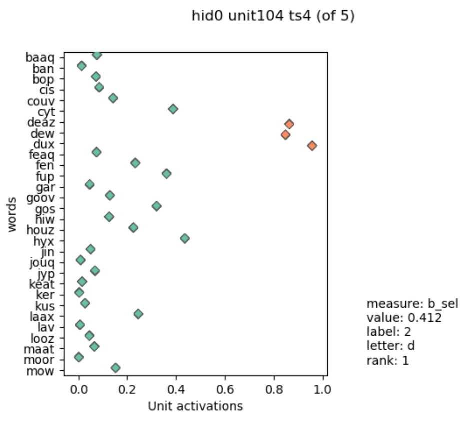
There was no effect of list length on the number of selective units. Kruskal-Wallis tests found no differences between models trained on lists of 1, 3, 5 or 7 words in terms of the median number of word-selective units ($H(5) = 2.33, p = 0.51$); or letter-selective units ($H(5) = 1.21, p = 0.75$). This is inconsistent with Bowers et al. (2014)’s finding that use of highly selective units increased with list length.

The selectivity levels are higher than in the previous experiment and approach the the .5 threshold of localist units in Bowers et al. (2014). There were no localist units (with a selectivity above .5) in any condition, however, I do find units with selectivity of .47 for words in a vocab 30 model trained on lists of five or seven words. the highest selectivity for letters is .41, in a vocab30 model trained on lists of five words (see figure 5.3 for visualizations).

The average number of selective units per timestep is similar to the average for selective units at the first timestep, suggesting that selectivity at the first timestep is a good predictor of selectivity at other timesteps. I also find a similar pattern for invariant units as in the previous experiments. The number of invariant units decreases with list length, there are none for letters in vocab 300, although I do find an invariant unit for words in vocab 300.



a)



b)

Figure 5.3: **Units with the highest selectivity for words and letters in Experiment 2** The $x$-axis shows the level of activation and the $y$-axis shows each word in the dataset. The feature the units is selective for is shown in orange. The figure text indicates the B-sel score, labelled as 'value', and the word or letter that the unit is selective for. Rank: 1 indicated that this was the most selective unit in that particular model. a) Unit with the highest selectivity for a word, with selectivity of .47 for the word 'dux', from a model trained on lists of five words from vocab 30. b) unit with the highest selectivity for a letter, with selectivity of .41 for the letter 'd', trained on the vocabulary of 30 items to recall five words.

### 5.4.2 Discussion

In experiment 2, models used a batch-size of 1 and the same (large) weight initialization as was used in Bowers et al. (2014). However, my models used a different optimizer and learning rate to their models, and the hidden states were set to zero at the start of each list rather than .5.

Selectivity was higher in this experiment than in the previous one, which can be attributed to difference in the models (e.g., online learning and different weight initialization). However, although selectivity is higher, the pattern of results is still similar to experiment 1: selectivity is higher for both words and letters when models are trained on vocab 30; and there is no effect of list length. Selectivity scores are similar for lists of one word, where there is no superposition-constraint as for longer lists, where there is a superposition-constraint. In contrast, Bowers et al. (2014) found selectivity was much reduced for lists of one, typically with less than one selective unit per model. I also find very few units selective for letters with vocab 300, which was the condition with the most letter-selective units in Bowers et al. (2014).

These findings challenges Bowers et al.'s claim that pressure to avoid the superposition catastrophe leads models to utilize highly selective units. I am disappointed not be be able to more closely replicate the models and training used in Bowers et al. (2014). Whilst they clearly worked in their simulations, I found that models with similar parameters would not converge. My parameter sweep included a model which used their initialization, optimizer, learning rate and hidden states, but after training it had a mean IoU of 0.0. This suggests that there are other differences between their study and mine which I have not yet identified, and which might allow models to converge with similar levels of selectivity to their results. Bowers et al. (2014) ran their simulations in LENS (Rohde, 1999), rather than Keras (Chollet and Others, 2015). LENS has some anachronistic or idiosyncratic processes (resetting of hidden states to .5; the custom optimization algorithm: 'Doug's momentum'; and the weight initialization strategy) and there may be other differences. The fact that I have been unable to directly replicate their study has made it hard to draw conclusions about why my results are so different to theirs. However, for now I can say that their findings do not generalise to other simple RNNs with the parameters used here.

In Bowers et al. (2014), they suggest that pressure to avoid the superposition-catastrophe will apply to most networks that co-activate multiple representations over a set of units, and that learning localist representations is the best way to avoid ambiguous representations. It could therefore be the case that the models used in experiments 1 and 2 are unusual exceptions in showing no effect of list-length on selectivity. Alternatively, it could be the case that there are many ways in which models overcome the superposition-catastrophe, and that increasing the selectivity of units in Bowers et al. (2014) is just one way. To further explore how models (that co-activate multiple representations over the same set of unit) can overcome the superposition-catastrophe, I turn to a different type of recurrent model. In the intervening years since 2014, Long-short-term memory (LSTM) units have emerged as a widely used type of recurrent unit, with gating mechanisms that allow it to store and retrieve information over longer periods of time than simple-recurrent units (Hochreiter and Schmidhuber, 1997). The following experiment will apply LSTMs to the same task and analyse their selectivity. This will show whether the pattern of results from Bowers et al. (2014) is found in other recurrent models.

## 5.5 Experiment 3: LSTMs

In the previous two experiments I have failed to replicate Bowers et al. (2014)'s finding that models learn localist units to avoid the superposition-catastrophe (Bowers et al., 2014). In these experiments, my models used simple recurrent units, as used by Bowers et al. (2014). However, there are other types of units which may be vulnerable to superposition-effects when solving sequence based tasks. Long short-term memory units (LSTMs) are a type of gated recurrent unit which allows it to accumulate information over longer durations than simple recurrent units; or gating can allow then to 'forget' information once it is not longer required (Hochreiter and Schmidhuber, 1997; Goodfellow et al., 2016). In this experiment, I will move away from simple-recurrent units and will apply LSTMs to this short-term memory task.

### 5.5.1 Methods

A parameter-sweep was conducted to find good training parameters for these models. I included 5 optimizers (Adam, Adam with AMS grad, RMSprop, SGD with momentum of .9 and SGD with momentum of .9 and gradients clipped at 1.0); 8 learning rates (.0005, .00075, .001, .0025, .005, .0075, .01, .025); and 2 batch sizes (1 or 32 lists). Models were only trained on lists of seven words from vocab 300, set to trained for a maximum of 10000 epochs. The best performing five models all used Adam optimizer with batches of 32 items. The model with the lowest loss had a learning rate of .005.

Models in experiment 3 will use a batch size of 32, Adam optimizer and a learning rate of .005. The weights were initialized with the 'Glorot-uniform' initialization and models used Keras' 'unrolled' setting. All conditions were trained for a maximum of 10000 epochs (or 1000000 trials), with early stopping if loss did not improve over 200 epochs. There were five runs of each condition, each initialized with different weights. Selectivity analysis was carried out on the output activation of each LSTM unit, rather than on their internal gates.

### 5.5.2 Results

For training time, accuracy and mean and max selectivity see table 5.5. Training time was shorter for models trained on vocab 30 than vocab 300, reflecting the difference in task difficulty. However, whilst the amount of training required increased with lists length for vocab 300 as expected, for vocab 30, training time was quickest for lists of seven items. This was driven by two of the five conditions which stopped training after 55 and 77 epochs respectively, both with .76 proportion of lists correct scores of .76. Accuracy was high in this experiment compared to Bowers et al. (2014), or experiments 1 and 2. All models achieved an accuracy above .98.

For selectivity details, see table 5.6. Selectivity was far lower than in previous experiments. The highest word-selectivity score in this experiment was 0.097, in a v30 model trained on lists of 5 words. This unit is just below the .1 threshold set to be considered a 'word-selective' unit. The highest letter-selectivity was .04, which occurred in several v30 models trained on list of 3, 5, and 7 items. For visualizations of the most selective units, see figure 5.4. The fact that there were no word-selective units word with a B-sel > .1 means that there was no effect of vocabulary size or list length on word selectivity. For letter selective units there was a clear effect of vocabulary size, with a median of 22.17 letter-selective units in vocab 30 and no letter-selective units for vocab 300. There was no effect of list length on the use of letter selective units ($H(5) = 0.94$, $p = 0.82$).

| Vocabulary size | List length | Epochs | Mean IoU | Proportion lists correct | Sims acc > .1 |
|---|---|---|---|---|---|
| 30 | 1 | 244 | 1.0 | 1.0 | 5 |
| | 3 | 252 | 1.0 | 1.0 | 5 |
| | 5 | 242 | 1.0 | 1.0 | 5 |
| | 7 | 181 | .99 | .91 | 5 |
| 300 | 1 | 4049 | 1.0 | 1.0 | 5 |
| | 3 | 4652 | 1.0 | 1.0 | 5 |
| | 5 | 6442 | 1.0 | .99 | 5 |
| | 7 | 7275 | 1.0 | .98 | 5 |

Table 5.5: **Experiment 3 results**. All values are the average of 5 runs from different initializations. Proportion lists correct is the proportion of lists where the IoU was 1.0. Sims acc > .1 gives the number of simulations (out of 5) where the proportion of correct lists was > .1, which were included in the analysis.

| Vocabulary | List | words ≥ .1 | | | letters ≥ 0 | | |
|---|---|---|---|---|---|---|---|
| size | length | ts 1 | ts ave | invariant | ts 1 | ts ave | invariant |
| 30 | 1 | 0 | 0 | - | 22.60 | 22.60 | - |
| | 3 | 0 | 0 | 0 | 31.00 | 30.20 | 7.80 |
| | 5 | 0 | 0 | 0 | 24.40 | 21.12 | 3.20 |
| | 7 | 0 | 0 | 0 | 25.60 | 23.11 | 4.25 |
| 300 | 1 | 0 | 0 | - | 0 | 0 | - |
| | 3 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 5 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 7 | 0 | 0 | 0 | 0 | 0 | 0 |

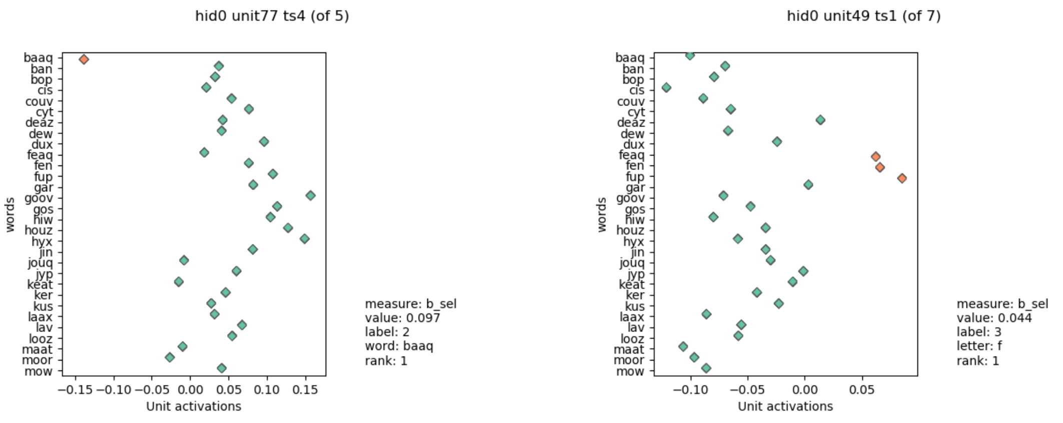
Table 5.6: **Count of units with selectivity above .1 for word or 0 for letters measures with B-sel for Experiment 3.** ts1 denotes the number of units at the first time step only, comparable with Bowers et al. (2014). ts av is the total number of units across all time steps divided by the number of time steps. Invariant is units that are selective for the same feature across all time steps. All values are the average per 200 unit model.

Again, this is inconsistent with Bowers et al. (2014)'s finding that selective units increased in number with list length.

### 5.5.3 Discussion

In this experiment I apply LSTMs to the short-term memory task from Bowers et al. (2014). LSTMs were able to learn the task more quickly than in previous experiments, and with higher accuracy, as expected from these powerful units. In fact the accuracy here is greater than for Bowers et al. (2014), where their models achieved 20% accuracy on the most difficult conditions, compared to accuracy of .98 for the LSTMs.

Unlike the previous two experiments, I find no word-selective units (e.g., with B-sel > .1). This means that the activation of the most active item, and the second most active item are



a) b)

Figure 5.4: **Units with the highest selectivity for words and letters in Experiment 3** The $x$-axis shows the level of activation and the $y$-axis shows each word in the dataset. The feature the units is selective for is shown in orange. The figure text indicates the B-sel score, labelled as 'value', the word or letter that the unit is selective for, and 'rank: 1' indicates that this was the most selective unit in this model. a) Unit with the highest selectivity for a word, with selectivity of .097 for the word 'baaq', from a model trained on five words from vocab 30. b) Unit with the highest selectivity for a letter, with selectivity of .04 for the letter 'f', trained on the vocabulary of 30 items to recall lists of seven words.

always similar, never having a gap greater than 10% of the range of activations. Although the results for word-selective units are dramatically different to the previous experiments, the count of letter-selective units is similar to the numbers in the previous experiments. This shows that none of the models needed word-selective units to solve the task.

There was an effect of vocabulary size on the number of letter-selective units, although this is again in the opposite direction to Bowers et al. (2014). They found more letter selective units for vocab 300 and suggested that this was because the model could not use word-selective units to avoid the superposition-catastrophe, as there were more words than units. Here I find that the model solves the larger vocabulary without word-selective or letter-selective units. Put differently, highly selective units are not *necessary* to avoid the superposition catastrophe. The short training times and high accuracy show that the representations that these models used were entirely sufficient.

Why are there no selective units for for models trained on vocab 300? If I allow that vocab 300 places additional constraints on the representations that the model can use to avoid the superposition-catastrophe; this implies that these models reduced their use of selective units in response to the superposition-catastrophe: the exact opposite effect of that predicted by Bowers et al. (2014). This might suggest that using highly selective units inhibited performance on vocab 300. A different explanation would be that highly selective units were neither harmful or helpful for high accuracy, but rather they were redundantly representing information that was also represented by other units. For the larger vocabulary, the model could not 'afford' to maintain these redundant representations, and instead, employed these units in a less selective capacity.

The proposed explanations for the differences in selectivity between the large and small vocabulary assume that the task demands are the main factor effecting selectivity. This is consistent with the claim from Plaut and McClelland (2000) that a model's representations (e.g., sparsity and selectivity) are determined by the structure of the task. However, there may be factors that contribute to the selectivity of representations which do not relate to the process of learning to solve a task. For example, in order for a unit to be selective for a letter, it requires that the three items with highest (or lowest) activation all contain the same feature. In vocab 30, there are 30 words, and each letter occurs three times. The probability of the three most or least active items containing a given letter by chance are:

$$\frac{3}{30} \times \frac{2}{29} \times \frac{1}{28} \times 2 = .00049$$

Therefore, I might expect a B-sel for letters $\geq 0$ to occur by chance in one out of every two-thousand units (i.e., once in ten models trained on single items, or four models if trained on lists of five items etc.). Each letter appears in thirty words in vocab 300, as such the probability the thirty most active items containing a given letter of around $1.2e^{-41}$! With these predictions, letter-selective units are far less likely in vocab 300 compared to the small one, which might explain why I consistently find more letter-selective units in vocab 30 than the large one. Importantly, Bowers et al. (2014) found more localist units (with the more stringent threshold of .5) in models trained on a large vocabulary than in models trained on the small vocab, which suggests that in their models, high selectivity was not stumbled upon by chance - but learned in response to task demands.

Regarding selectivity for words, I consider a unit to be selective for a word if the B-sel is greater than .1. This is far below the threshold of .5 used in Bowers et al. (2014), and it is also higher than the threshold of 0 used for letter selective units. This is because there is very little variance in the responses for each instance of a word, so all instances can be treated as equal (e.g., see figures 5.2 and 5.3). There is always one word with the highest activation, and by definition this will have a B-sel greater than zero. Therefore, for word-selective units the critical factor is *how much* higher is the activation of the most active item than the second highest item (or how much lower is the least active item than the second lowest). If I consider the probability of word-selective units occurring by chance, I could suggest that when sampling values between zero and one, the distance between the two highest numbers will be larger if a small sample is drawn (e.g., 30 values) than if a large sample is drawn (e.g., 300 values). I am not suggesting that selective units occurs purely by chance, but stating that it might be a factor. Alternatively, differences between the level of activation for two items at a given unit can be explained by differences in the weights connecting the inputs to that hidden unit. For example, if the word 'cat' is more highly activated than the word 'cot', I can infer that there is a stronger connection from the units involved in representing 'a' than 'o'. I can be sure that such differences are caused by the weights, not the input data, because for the datasets used in this study, all input patterns are binary (i.e., there are no differences in the level of activation of input units, they are all zero or one), and all items have the same proportion of 1s and 0s. The are many interacting factors that affect the weights of a trained model, such as the weight-initializer, optimizer, learning rate and batch-size.

My models and Bowers et al. (2014)'s models appear to be responding to the task demands differently (e.g., by increasing or decreasing their use of selective units for the larger dataset). The following experiment will pursue the idea that factors other than task demands may contribute to the selectivity of the representations. I will focus on just one of the factors related to the weights in a trained model, the weight initialization method, to see how this might contribute to differences in levels of selectivity. I will measure the selectivity of representations in *untrained* models: any differences in selectivity between models will relate to their weight-initialization, rather than to differences in their training (e.g., optimizer or learning rate) or in how the model deals with the task demands during training.

## 5.6 Experiment 4: Untrained Models

In three experiments I have attempted to replicate Bowers et al. (2014)'s finding that models use highly selective units to avoid the superposition-catastrophe. I find a very different pattern of results to that in Bowers et al. (2014). I find no units with selectivity greater than .5; no effect of list length, and lower levels of selectivity for both words and letters for models trained on the larger vocabulary. However, differences in the methods used to train the models make it hard to draw conclusions as to why they saw a very different pattern of selectivity to my models. The following experiment will analyse *untrained* models, which will remove the influence of my different training methods and allow us to highlight the impact of weight initialization on selectivity.

It is not clear whether there will be any selective units in untrained models. I predict that it is very unlikely that a letter-selective unit will emerge by chance, with long odds that the items with highest activations will all contain a particular letter. For vocab 30, the probability of this occurring by chance is around .00049, and for vocab 300 is is around $1.2e^{-41}$! I have not estimated the probability that the word with the highest activation would > .1 above the second highest word, but I expect this is more likely to occur for vocab 30 than the large one. I also predict that word-selective units will occur more often than letter-selective units. Invariant units, which are selective for the same feature across multiple timesteps also seem very unlikely to occur by chance.

Prior to training a model, its accuracy is of course very low. In previous studies, all incorrect items were removed before analysis, such that analysis only considers the items that the model correctly identifies. That strategy is inappropriate here, so instead I shall analyse the selectivity of units based on all items, the majority of which will be incorrect.

### 5.6.1 Methods

I will analyse models with three different weight initializers that have been used in these studies, variance scaling, LENS initialization and Glorot Uniform. For each weight initialization method I will initialize three models for each of the eight data conditions (e.g., large or small dataset, lists of 1, 3, 5 ,and 7 items), giving 72 models in total. Models have a single hidden layer of 200 simple recurrent units, as in experiments 1 and 2 and Bowers et al. (2014). Experiment 4a will use variance scaling, the initializer used in experiment 1. For variance scaling, weights are sampled from a truncated normal distribution with a mean of zero and a standard deviation of $\sqrt{\frac{1}{\text{input units}}}$, which is .17. Experiment 4b will that same initializer as experiment 2 (referred to as the LENS initializer); which is based on the one from the LENS software used in Bowers et al. (2014). This initializer samples weights from a random uniform distribution between -1.0 to 1.0. Experiment 4c will use Glorot-uniform, Keras' default initializer for Simple RNN and LSTM recurrent layers, as used in experiment 3. This initializer samples weights from a uniform distribution where the upper and lower bounds are equal to the square root of $\frac{6}{\text{input units + output units}}$. For the vocab of thirty items this would bound weights between $\pm.32$ and for the vocabulary of 300 items the limits would be at $\pm.13$.

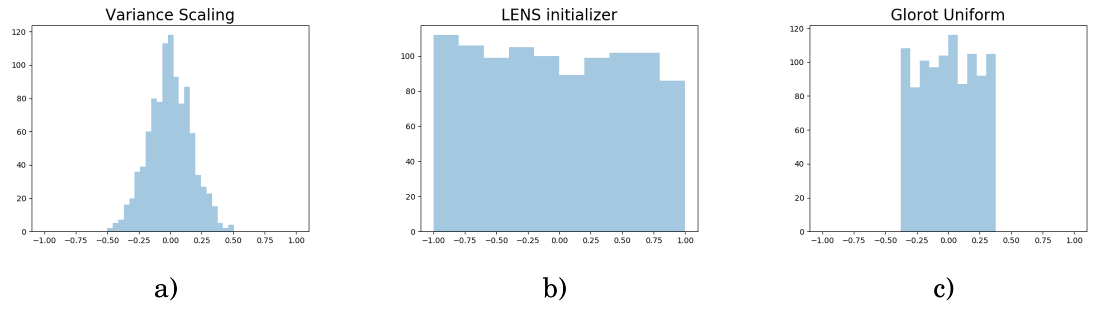


Figure 5.5: **Distribution of weights with different initializers** Plots showing examples of values sampled with the three weight initialization methods. All plots show 1000 data points, with a mean of zero. a) variance scaling: Values sampled from a random normal distribution with a standard deviation of .17. b) LENS initializer: Values are sampled from a random uniform distribution with limits at -1 and 1. c) Glorot Uniform: Values sampled from a random uniform distribution with limits at $\pm.32$. Glorot uniform includes the number of output weights in its calculation and as such, gives different values for the small and large vocabulary. Values here are for the larger vocabulary, for the smaller vocabulary, weights would be limited at $\pm.13$.

### 5.6.2 Results

For experiment 4 selectivity scores see table 5.8. All three weight initializers have similar pattern of selectivity as the trained models in previous experiments. They all shown an effect of vocabulary

| Weight | Vocab | List | B-sel(words) | | | | B-sel(letters) | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| initializer | size | length | mean | max | ts ave | invar | mean | max | ts ave | invar |
| 4a | 30 | 1 | .05 | .18 | 13.33 | 13.33 | 0 | .09 | 27.33 | 27.33 |
| Variance | | 3 | .05 | .19 | 15.78 | 6.33 | 0 | .13 | 26.56 | 12.00 |
| Scaling | | 5 | .05 | .20 | 13.13 | 2.33 | 0 | .17 | 26.33 | 6.67 |
| | | 7 | .05 | .25 | 13.81 | 1.00 | 0 | .11 | 25.57 | 4.00 |
| | 300 | 1 | .02 | .06 | 0 | 0 | 0 | .01 | 0.33 | 0.33 |
| | | 3 | .02 | .12 | 0.44 | 0 | 0 | 0 | 0 | 0 |
| | | 5 | .02 | .09 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 7 | .02 | .14 | 0.29 | 0 | 0 | 0 | 0 | 0 |
| 4b | 30 | 1 | .06 | .29 | 35.33 | 35.33 | .01 | .20 | 28.67 | 28.67 |
| LENS | | 3 | .07 | .50 | 41.67 | 23.00 | 0 | .22 | 22.44 | 10.33 |
| Initializer | | 5 | .07 | .39 | 37.87 | 13.33 | .01 | .34 | 23.60 | 10.67 |
| | | 7 | .07 | .36 | 40.38 | 13.67 | 0 | .22 | 25.19 | 7.00 |
| | 300 | 1 | .02 | .14 | 1.67 | 1.67 | 0 | 0 | 0 | 0 |
| | | 3 | .02 | .16 | 1.67 | 0 | 0 | 0 | 0 | 0 |
| | | 5 | .02 | .24 | 2.93 | 0 | 0 | 0 | 0 | 0 |
| | | 7 | .02 | .23 | 2.10 | 0.33 | 0 | 0 | 0 | 0 |
| 4c | 30 | 1 | .03 | .11 | 0.33 | 0.33 | 0 | .05 | 29.33 | 29.33 |
| Glorot | | 3 | .03 | .11 | 0.89 | 0.33 | 0 | .06 | 27.56 | 11.67 |
| Uniform | | 5 | .03 | .12 | 0.67 | 0 | 0 | .06 | 25.60 | 7.00 |
| | | 7 | .03 | .13 | 0.48 | 0 | 0 | .06 | 24.24 | 5.33 |
| | 300 | 1 | .01 | .05 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 3 | .02 | .06 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 5 | .01 | .07 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 7 | .01 | .07 | 0 | 0 | 0 | 0 | 0 | 0 |

Table 5.7: **Experiment 4 results for untrained models.** All values are the average of 3 runs from different initializations. Proportion lists correct was zero for all runs. Analysis of hidden unit activations was conducted on correct and incorrect responses.

size, with far higher levels of selectivity for the small dataset than for the large dataset (all Dunns' test *p*s < .001). The median number of selective units for the large dataset is zero, apart from word-selectivity for LENS initializer models, where the median number of word-selective units is 2. This is still far lower than the median of 38.86 word-selective units for vocab 30. For Glorot Uniform, the median number of word-selective units for the small dataset is only .38, however, this significantly different to the median of zero for vocab 300. There is no effect of list length on word or letter-selective units for any of the weight initializers. Kruskal-Wallis tests found no significant differences in the number of selective units for models trained on different length lists (all *p*s > .66).

Levels of selectivity are highest for LENS initialization and are lowest for Glorot Uniform. However, differences between in the mean selectivity scores are small, when compared to differences in the maximum selectivity. Put differently, although the average selectivity score are slightly higher with LENS initializer than for the other initializers, the range of selectivity scores is much higher when initializing models with LENS initializer.

Interestingly, I do find a unit with selectivity close to the threshold of .5, above which, Bowers et al. (2014) considers a unit to be localist. This unit had a B-sel of .498, and was in a model initialized with the LENS initializer for vocab 30 and lists of three words. This demonstrates that high levels of selectivity do not only emerge in response to task demands, but can occur by chance through initializing the model with random weights. The highest selectivity for letters is also in a LENS initialized model, for lists of 5 words from vocab 30 with a B-sel of .34.

| Exp | Weight | Trained | words B-sel | | words ≥ .1 | | letters B-sel | | letters ≥ 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Initializer | | mean | max | ts ave | invar | mean | max | ts ave | invar |
| 1 | Variance | Yes | .04 | .26 | 8.84 | 3.95 | 0 | .17 | 13.37 | 6.30 |
| 4a | Variance | No | .04 | .13 | 7.10 | 2.88 | 0 | .06 | 13.27 | 6.29 |
| 2 | LENS | Yes | .05 | .47 | 22.15 | 10.88 | 0 | .41 | 13.36 | 7.35 |
| 4b | LENS | No | .04 | .50 | 20.45 | 10.96 | 0 | .34 | 12.49 | 7.08 |
| 3 | Glorot | Yes | .01 | .10 | 0 | 0 | 0 | .04 | 12.31 | 4.87 |
| 4c | Glorot | No | .02 | .13 | 0.30 | 0.08 | 0 | .06 | 13.34 | 6.67 |

Table 5.8: **Summary table of conditions 4a, 4b and 4c, and also for comparison, showing a summary of experiments 1, 2 and 3**. Table showing the mean and max selectivity scores for words and letters and a count of units with selectivity above .1 for words or 0 for letters measures with B-sel. ts ave is the total number of units across all time steps divided by the number of time steps. Invar (invariant) is units that are selective for the same feature across all time steps. All values for untrained models are the average of 24 simulations: 2 (vocabulary) x 4 (list length) x 3 (runs). Untrained selectivity is calculated on correct and incorrect items. Values for trained models are the average of 40 simulations: 2 (vocabulary) x 4 (list length) x 5 (runs). Selectivity for trained models is calculated on correct items only.

For a comparison of the untrained models in experiment 4 with trained models from experiments 1, 2 and 3, see table 5.8. Rows are organized by weight initializer, showing the scores for trained models first, then the untrained models (e.g., top row shows results from experiment 1 with trained models initialized with variance scaling, the second row shown results from untrained models initialized with variance scaling in exp 4a). Scores here have been collapsed across vocabulary size and list length. Levels of selectivity are very similar between trained and untrained models using the same initializer. For example, the scores for experiment 4a (untrained models with variance scaling) are more similar to the scores from experiment 1, a trained model with variance scaling than they are similar to the scores from experiments 4b and 4c, which are other untrained models with different initializers. This suggests that weight initialization explains a lot of the differences in selectivity between experiments. This is remarkable considering the fact that the selectivity scores for untrained models are based on representations which do not solve the task (e.g., accuracy is close to zero). For variance scaling and LENS initializer models, selectivity is slightly higher in trained models than untrained models. For Glorot Uniform, selectivity is lower in the trained model than the untrained model. However, here I am comparing untrained simple recurrent units with trained LSTMs, which might not be a fair comparison. However, my results suggest that the weight initialization method plays a strong causal role in the end levels of selectivity.

One surprising detail that I had not previously noticed is that for models initialized with variance scaling or Glorot Uniform, there are more letter-selective units than word-selective units. This is true of both trained and untrained models. Letter and word selective units have different thresholds, selectivity must be greater than .1 for words, but only 0 for letters. As such, they are not directly comparable, but I had assumed that letter selective units would be more rare, as they require they require the co-ordination of multiple items (e.g., high selectivity for all words containing a given letter), whereas a word selective unit only requires that a single item has high activation. The fact that there are more letter-selective units than word-selective for untrained models with these two initializers shows that letter-selective units frequently occur, in the absences of training or task demands. The number of letter-selective units was lower than the number of word-selective units for models with the LENS initializer.

Another surprising feature of untrained models is how frequently invariant units were present. I had expected that for a unit to be selective for the same feature across multiple timesteps, would require training. Again I see that this is not the case, the numbers of invariant units are similar between trained and untrained models.

### 5.6.3 Discussion

In experiment 4, I compared the selectivity of untrained models that used three different weight initialization methods: variance scaling, LENS initializer and Glorot Uniform. Since these models were untrained, selectivity analysis involved analysing the activation of incorrect items.

I found a striking similar pattern of selectivity scores in untrained models to the results from trained models in experiments 1, 2 and 3. I found no effect of list-length in untrained models, as was the case for previous trained models. However, there was a significant effect of vocabulary size, with more selective units for vocab 30 than vocab 300. This effect of vocabulary size in untrained models cannot be due to the model having to avoid superposition-constraints, since the model was untrained and didn't avoid anything! This suggests that I should not seek to explain differences between selectivity for vocabulary types in trained models with reference to task demands.

I found that the pattern of selectivity scores for each of the initializer types, was very similar for untrained models to the results for trained models with the corresponding initializer type (e.g., the average number of selective units for untrained models using variance scaling, was similar to the results from trained models with variance scaling in experiment 1). This suggests that the initial weights prior to training greatly constrain the types of representation that the trained model uses. I see little evidence that models radically change the range of selectivity scores for the units. However, this contrasts with research showing that models increase the number of selective units during training (e.g., Zhou et al., 2017). I find that the selectivity scores for trained models using a particular initializer are more similar to untrained models using the same initializer than to trained models using a different initializer. This shows that the choice of initialization in these experiments had greater influence over the selectivity of the units than the actual training!

My results show that models initialized with LENS initializer are associated with higher selectivity than models initialized with variance scaling or Glorot Uniform. I suggest that rather than LENS boosting selectivity in general, this is a product of LENS initalizer sampling weights from a larger distribution, resulting in a larger range of selectivity scores.

In summary, experiment 4 provides some explanations of the differences in selectivity between experiments 1,2 and 3. I find that the weight initializer has a strong impact on the levels of selectivity. In fact the levels of selectivity are remarkably similar between untrained and trained models. I find higher levels of selectivity in untrained models initialized with LENS initializer, as used in Bowers et al. (2014), compared to variance scaling and Glorot Uniform. This finding mirrors the pattern of results seen for trained models, with highest levels of selectivity in LENS initializer models than for the other initializers. However, the choice of weight initializer is not the only factor that influences selectivity, trained models different learning rate, optimizer, training times and batch size, all of which might impact on selectivity. Whilst the results from experiment 4 explain some of the differences in selectivity between experiments 1, 2 and 3, experiment 4 does not explain the pattern of selectivity results seen in Bowers et al. (2014). I can suggest that their initialization strategy led to units with a wide range of selectivity scores, including some highly selective units. However, the differences between this study and Bowers et al. (2014) are not simply a case of different levels of selectivity. The most significant differences are that I found no effect of list length and that vocab size had a very different impact here compared to in their study. It is hard to reconcile those differences simply by appeals to the levels of selectivity. So whilst this experiment highlights the impact of weight-initialization on selectivity, it does not resolve the question of why I did not see the effects attributed to the superposition-catastrophe in Bowers et al. (2014).

## 5.7 General Discussion

The aim of this chapter was to investigate the finding that models learned highly selective units in order to avoid the superposition-catastrophe (Bowers et al., 2014). They trained models to recall lists of words, and recorded the selectivity of hidden units for words or for letters. They identified three dataset properties that impacted on the levels of selectivity: the similarity of the input representations (e.g., input-density), the number of items that the model was trained to recall (e.g., list-length) and the total number of items that the model had to represent (e.g., vocabulary size). The effects of input-density and list-length were seen in the model's use of selective units: there were more selective units for dense input data than sparse input data; and more selective units for long lists than short lists. However, the effect of vocabulary size related to the features being represented by each unit: units were selective for words when trained on vocab 30, but selective for letters when trained on vocab 300.

In Bowers et al. (2014), they only analysed the selectivity of models at the first timestep. I aimed to analyse the selectivity across all timesteps to see if selective units were invariant to time, selectively representing the same feature across all timesteps. I investigated the effects of list-length and vocabulary size in four experiments (I did not vary the input density). In the first two experiments I attempted to closely replicate their study using simple recurrent models. However, I found that models would not train when they used similar parameters to Bowers et al. (2014), as such I could not directly replicate their approach. I was able to train simple recurrent models to solve the task, using slightly different parameters. In the third experiment I switched to using LSTMs, a more powerful type of recurrent unit, which solved the task with high accuracy. In the fourth experiment I investigated the representations present in untrained models, using the three different weight initialization methods that had been used in experiments 1 2 and 3. Across all four experiments I consistently found the same pattern of results, which was very different to that found in Bowers et al. (2014). I found no effect of list length on the number of selective units in any experiment. I found an effect of vocabulary size, with higher levels of selectivity for both words and letters in vocab 30. These findings are inconsistent with the claim that models learn highly selective units to avoid the superposition-catastrophe.

The superposition-catastrophe only relates to learning sequences, and as such, it should not apply to lists of one item. I found no effect of list length in any experiment, meaning that there was no difference in the level of selectivity between models trained on lists of one item, and models trained on lists of multiple items. This suggests that the level of selectivity in my models is not related to superposition-constraints at all. One interpretation of this could be that there *were* effects of list length, but that they were so small that they did not reach significance. This might apply if my models found the task trivially easy to solve. For example, in experiment 3, all my LSTM models reached accuracy > .98, compared to an accuracy of just .2 for the most difficult conditions in Bowers et al. (2014). This might suggest that these powerful units were able to solve the task with a different approach to Bowers' models. However, the pattern of results for experiment 3 was the same as for experiment 1 and 2, where models had similar accuracy to Bowers' models, and similar to experiment 4, where models were untrained. I also see a clear effect of list length on trained models in all experiments, with longer lists taking requiring more training and having lower accuracy than short lists. In other words, list-length clear had effects on training in my models, but not on selectivity.

The effect of vocabulary size in Bowers et al. (2014) related to the information represented by selective units. They only found word-selective units in models trained on vocab 30, and only letter-selective units in models trained on vocab 300. In this study I find more word-selective and letter-selective units in vocab 30 than vocab 300.

There were several other notable findings from experiments 1 to 3, with trained models. I found that the average number of selective units per timestep is similar to the number of selective units at the first timestep, suggesting that the number of local units identified in Bowers et al. (2014) at the first timestep would be good estimate of the selectivity of the model in general. I found a number of invariant units, which retain their selectivity for a particular feature across all timesteps. However, only a small proportion of selective units are invariant. This means that the role a unit plays can not be accurately described from its selectivity in the first timestep alone. Put differently, the selective units in Bowers may have appeared to be 'detectors' for a given feature, but this can not be established just from one timestep.

The findings from experiments 1 to 3 are a challenge for the claim that models learn highly selectivity units to avoid the superposition catastrophe. The selective units I find can not be attributed to superposition-constraints, as there was similar number of selective units for lists of one item, as for longer lists. Bowers et al. (2014) defined units as localist if their B-sel was > .5. They claim that there finding show that distributed representations are only effective in a limited conditions, and suggest that their localist units are an effective alternative to distributed representations when there are strong superposition-constraints. No unit in my trained models had B-sel > .5 that they could describe as localist. As such, there is no evidence to support the claim that models learn localist units to avid the superposition-catastrophe.

In experiment 4, I analysed untrained models, comparing the representations of models that used one of three different weight initialization methods. As with trained models, I find no effect of list length and an effect of vocabulary size, such that both word-selective and letter-selective units occur at higher rates for vocab 30. I found strikingly similar levels of selectivity between trained and untrained models when grouped by weight initializer. This highlights the important contribution that weight initialization makes to a trained models range of selectivity scores. Initializing weights with larger values means that training starts with bigger differences in how individual items are represented. I see this in the fact that the mean levels of selectivity for models using LENS initialization are at the same level as other the initializers for words, yet the maximum B-sel scores are higher for LENS initialized models, and there are more word-selective units. These initial starting weights constrain the solutions that are available to the model. Whilst training undoubtedly changes the representations used by the model, my comparison of trained and untrained models suggests that through the process of learning, the *types* of representations used by the model do not change much. If the model starts with a large range of selectivity scores in the hidden layer, it will end with a similar range of selectivity score. My results show that the weight initialization method greatly constrains the selectivity of the solutions discovered by each model during training.

I had assumed that invariant units would occur very rarely in untrained models, yet I found them in similar numbers to trained models. My untrained models also contained the unit with the highest selectivity across all four experiments, a word-selective units with a B-sel of .5. Its presence in an untrained model along with multiple letter-selective units and invariant units highlights the fact that these features, which might be considered to be good examples of 'interpretability' in neural networks, can occur as meaningless artifacts. They were generated through the random initialization process, not via the guiding backpropagation signal that related to task performance.

In my experiments, I found a significant effect of vocabulary size, which was in the same direction for both word-selective and letter-selective units; but no effect of list length. Therefore, I must explain how different size datasets result in different levels of selectivity. Since, selectivity was similar in trained and untrained models, my explanations should *not* rely on factors that occur during training. One possible explanation relates to the probability of selective units occurring by chance due to the random process of weight initialization. For example, when randomly sampling values between zero and one, the distance between the highest value and second highest value is likely to be larger if you sample 30 items than 300 items. By this logic, word-selective units are likely to occur more often in vocab 30 than in vocab 300. For letter-selective units, one can use the analogy of drawing balls from a bag when 10% of the balls are red. The probability that you will draw *all* the red balls before any white balls are drawn is higher when there are 30 balls than when there are 300 balls. In this analogy the order in which the balls are drawn relates to their activation, with the first items having the highest activation. This predicts that there will be more letter-selective units in vocab 30 than the large one. The probabilities that I calculated for this were way off, selective units were found in untrained models at much higher rates than I expected. However, for both word-selectivity and letter-selectivity, these explanations are consistent with my findings: selectivity was higher for vocab 30 than vocab 300. As such, this explanation appears sound in the context of these four experiments. Unfortunately, this explanation offers nothing relating to the results from Bowers et al. (2014). They found that vocab 300 was associated with *reduced* numbers of word-selective units, but *increased* number of letter-selective units. They also found an effect effect of list-length, and I did not. I do not have an explanation that can account for this difference between my study and theirs.

I was not able to train models when they had settings close to those used in Bowers et al. (2014). I was unable to train models that used their optimizer (Doug's momentum) or that used LENS states (hidden states reset to .5 at the start of each list). There may well be other training processes in LENS which contributed to the selectivity which I have not identified. In this study, I have demonstrated the impact that the choice of weight initializer can have on selectivity. My results suggest that their LENS initializer may well have increased the likelihood of highly selective units emerging. However, I used the LENS initializer in experiments 2 and 4, and did not find their pattern of results. As such, whilst the choice of initializer may contribute to the levels of selectivity, it does not explain the differences between my study and their.

In summary, my experiments shown that models can avoid the superposition-catastrophe without increasing their use of highly selective units. This challenges their claim that distributed representations are unsuitable for tasks with high superposition-constraints. I also have shown that the weight initialization process can greatly impact the selectivity of trained models.

Chapter 6

# Selectivity and lesioning

## 6.1 Introduction

Research approaches for understanding how neural networks solve complex tasks can involve analysing hidden units individually. One approach is selectivity analysis: recording the level of activation for items in a dataset to see if activation is associated with the presence of some labelled feature in the data. Researchers have used various selectivity measures including Bowers' selectivity (B-sel, Bowers et al., 2014; Vankov and Bowers, 2017), class conditional mean activation (CCMA, Morcos et al., 2018), precision (Zhou et al., 2015) and maximum-informedness (max-info, Gale et al., 2020).

A different approach to understanding the role of individual units is through lesioning (also known as ablation). This involves 'removing' a unit (or units) from a model (by setting a unit's weights and biases to zero) and comparing the lesioned model's performance to the original model. Where there are items that were correctly classified by the full model but are incorrect in the lesioned model, it suggests that the lesioned unit plays an important role in the recognition of those items.

Lesioning and selectivity analyses can be used in tandem to relate the pattern of activation in a unit (selectivity) to the functional role played by a unit (lesioning). Morcos et al. (2018) tested three models to investigate the relationship between class-selectivity and unit importance. Class selectivity was measured using CCMA; and unit importance was defined as the drop in total accuracy when the unit was lesioned. The logic being that a unit where lesioning leads to a larger drop in accuracy is more important to the model than a unit that where lesioning leads to no drop in accuracy. The models were a 2 layer multi-layer perceptron (MLP) trained on MNIST; an 11 layer convolution neural network (CNN) trained on CIFAR-10 and ResNet50 (a 50 layer CNN) trained on imageNet. In the MLP and ResNet, there was no significant association between selectivity and unit importance. For the CIFAR-10 model they found a moderate negative correlation (Spearman's $r$ = -.43, $p$ < .001). They then tested the correlations separately for each layer. In the final layers of the model, where selectivity was highest, they found no relationship between selectivity and importance. However, in early layers, where selectivity was lowest, they found strong negative correlations meaning that increases in the selectivity of units were associated with a decrease in the drop in accuracy. Put differently, in early layers, the least selective units were most important. Morcos et al. (2018) suggested that high levels of selectivity are either unnecessary for good performance or even detrimental to performance. Therefore, studies that investigate selectivity may be focusing on the wrong (i.e., least important) units for understanding how models succeed on tasks. They concluded that: "...methods for understanding neural networks based on analyzing highly selective single units, or finding optimal inputs for single units, such as activation maximization may be misleading".

It is worth considering why it was in the early layer where Morcos et al. (2018) found a negative association between selectivity and unit importance. Firstly, selectivity was lowest in the early layers because the early layers learn simple features that are relevant to many classes. For example, gabor-like kernels (often found in early layers, e.g., Zeiler and Fergus, 2014) that are sensitive to simple oriented lines will be useful for many classes and therefore will have low selectivity and high importance. Kernels in later layers may be sensitive to something more complex and class-specific (e.g., keyboards or dog-faces, Zeiler and Fergus, 2014), and as such lesioning these units will only impact a subset of classes. The second reason why units in early layers make a greater contribution to overall accuracy is that there are fewer units per layer at early layers than later layers. The number of kernels per layer for the 11-layer CNN used in Morcos et al. (2018) were 64, 64, 128, 128, 128, 256, 256, 256, 512, 512 and 512. Therefore, lesioning a unit in early layers will reduce the representational capacity of the layer to a greater extent than lesioning a unit in later layers.

However, Zhou et al. (2018b) suggested that a unit's role could be better understood by considering the drop in accuracy for each class (e.g., the *class-accuracy-drop*), rather than the drop in in overall accuracy. They found that following lesioning, for many units the accuracy for most classes was unchanged, but there was a significant drop in accuracy for a small subset of classes. As such, seemingly unimportant units (i.e., lesioning led to a small drop in overall accuracy) were in fact specialised units of high importance to a subset of classes or a single class. Zhou et al. (2018b) trained Alexnet and ResNet18 CNNs on the ImageNet and Places datasets, and recorded the selectivity per unit with three selectivity measures: CCMA, class-correlation (Pearson correlation between a unit and an output unit) and concept-alignment (also known as Network Dissection). They then lesioned each unit in turn, and recorded the total accuracy drop and the class-accuracy-drop. A drop in accuracy was recorded with negative values, so if accuracy dropped by 10% this was recorded as -.1. Zhou et al. (2018b) found that unit selectivity had a weak positive correlation with drop in overall accuracy following lesioning. This means that highly selective units were typically associated with a smaller total-accuracy-drop then units with low selectivity. This is consistent with Morcos et al. (2018)'s claim that highly selective units are of low importance for overall accuracy, and with Morcos' results in the 11 layer CNN; although Morcos' found no effect in the MLP and ResNet. Zhou et al. (2018b) then measured the class-accuracy-drop of each unit and identified the class with the largest drop in accuracy, which they call the max-class-drop. They found significant weak negative correlation between selectivity and the max-class-drop. This means that the drop in accuracy for class $x$ is typically larger for a highly selective unit than in a unit with low selectivity. This relationship between selectivity and class-accuracy-drop was further demonstrated with logistic regression classifiers, which were trained to predict which class would have the max-class-drop based on the selectivity scores for all classes. The regression models were trained on units from the conv 5 layer of three models and tested on the conv 5 layer of two different models. Classifiers trained on the three selectivity measures (CCMA, class correlation and network-dissection) all had an accuracy of 38%. The examples of specialization seen in figures 1 and 2 in Zhou et al. (2018b) show that only a few classes are impacted by lesioning. However, they do not give any class-selectivity details for these units such as the the level of selectivity and identity of the most selective class. They do discuss the *feature* with highest selectivity from the concept-alignment measure (e.g., network-dissection), although these features (e.g., 'beds', 'wheels') are broader than the class labels. For example, they show that a unit with selectivity for the feature 'beds' is important for the classes 'youth-hostel', 'bedroom' and 'bed-chamber'. Based on their findings, Zhou et al. (2018b) claimed that units specialise for class (or subset of classes) and that selectivity is a 'good predictor' of a unit's importance for particular classes. This means that selectivity is predictive of the functional role played by a unit.

Other studies had previously considered the effects of lesioning for particular items or features. (Bowers et al., 2016) trained RNNs to recall lists of words and identified 33 highly selective units with selectivity > .1 for a particular letter that formed the words (selectivity was measured with B-sel, see 3 for details). Lesioning these units led to ≤ 2% drop in accuracy for words that did *not* contain the given letter, suggesting that units were not important for words that did not contain the letter that the units were selective for. However, units differed greatly in terms of their importance for words that *did* contain the selective letter. For two units, lesioning led to failure on *all* words containing the selective letter, meaning that the unit was vitally important for those words. However, for eight units there was no effect of lesioning at all, meaning these units were not important for any letter. The average effect of lesioning these units was a 25% drop in accuracy for words containing the selective letter. In summary, high selectivity in these units predicted which words these units were important for; but not *how* important these units were for those words. This study shows that the relationship between selectivity and the functional role played by a unit is weak, even for very highly selective (e.g., linearly-separable) units.

In summary, Zhou et al. (2018b) suggested that the effects of lesioning should be based on the change in accuracy for each class (class-accuracy-drop) rather than the change in total-accuracy, as proposed by Morcos et al. (2018). Zhou et al. (2018b)'s analysis of models using class-accuracy-drop led them to make two claims which have informed this study. Firstly, they claimed that class-accuracy-drop showed that units are specialised for a subset of classes. A lesioned unit is considered to be specialised for class $x$ if: a) many items from class $x$ fail leading to a significant drop in accuracy for class $x$; b) most of the items that fail are from class $x$ such that accuracy for other classes is unaffected and the drop in overall accuracy is modest. My concern here is not whether the models used in Zhou et al. (2018b) were especially interpretable. Rather, I am interested in whether class-accuracy-drop provides a new tool for interpreting models and understanding the functional role played by individual units. The second key claim from Zhou et al. (2018b) is that selectivity measures were good predictors of the effects of lesioning. This was based on the significant correlations between levels of selectivity and max-class-drop, along with the performance of regression models which predicted which class would have have the max-class-drop based on the selectivity scores for that unit. However, two of the selectivity measures they used (CCMA and network-dissection which they refer to as concept-alignment) have been shown to produce inflated selectivity scores Gale et al. (2020). Units can have high selectivity scores in spite of a high false alarm rate or a low hit rate. This raises the question of whether there are different selectivity measures that would be better predictors of the effects of lesioning than those used by Zhou et al. (2018b) and Morcos et al. (2018).

In the first part of this chapter, I will measure the selectivity of a small CNN with CCMA and also Maximum-informedness and Precision, two selectivity measures not used by Zhou et al. (2018b), and measure the effects of lesioning with class-accuracy-drop. I find that in my model, max-class-drop is strongly correlated with the total-accuracy-drop drop. This means that the units with the highest max-class-drop are not specialised. This motivates the search for an alternative to class-accuracy-drop as a measure of the effects of lesioning. The second part of this chapter I propose a new measures of the class-specific effects of lesioning, based on the proportion of the drop in overall accuracy that each class accounts for termed 'drop-prop'. I find that the units with highest drop-prop are not specialised units, however, specialised units are identified when drop-prop and class-accuracy-drop are used in conjunction. There are very few units in my model that are specialised for a particular class. I find that selectivity measures have some predictive power relating to both the class which will have the largest drop in accuracy and the size of the drop in accuracy. However, I also show that the most selective class is rarely the class with the largest drop in accuracy. This suggests that caution is advised when using selectivity measures to predict the functional role of a unit.

## 6.2 Analysis 1

### 6.2.1 Introduction

This study will investigate Zhou et al. (2018b)'s claim that measuring the effects of lesioning with class-accuracy-drop can identify units that are specialised for a particular class. These findings are based on a lesioning study of large, deep CNNs trained on large datasets. I will apply Zhou's methods on a five layer CNN trained on CIFAR-10. My concern is not whether the degree of 'specialization' reported by Zhou extend to small networks per se, but to test whether class-accuracy-drop is a useful tool for understanding the functional role played by individual units.

Selectivity was measured in Zhou et al. (2018b) with four measures: class-selectivity (CCMA), class correlation, concept alignment (network-dissection) and L1 norm. I will use the CCMA selectivity measure used by Zhou, but will also include precision and maximum-informedness to test whether they are better predictors of unit specialisation than CCMA. I have described network-dissection as a precision-based measure, although it is a measure of feature-selectivity based on a set of 1200 visual features. Precision as used here is a measure of class-selectivity that is not specific to any particular task.

In Zhou et al. (2018b) they investigated the relationship between the size of the max-class-drop and the level of selectivity with Spearman rank-order correlations. For this, they state that they use the level of selectivity of most selective class for each unit. They are therefore asking if there is a relationship between the accuracy drop for the class with the largest accuracy drop and the selectivity of the most selective class. This leaves open the possibility that these refer to two different classes: a unit might be most selective for class-$x$ and yet the largest drop in accuracy occurs for class-$y$. In this study I will therefore test for a relationship between the accuracy drop for the class with the largest accuracy drop and the selectivity of that same class.

### 6.2.2 Methods and materials

#### 6.2.2.1 Data and model

**Model:** I conducted a parameter sweep of various models trained on CIFAR-10 including models with 2, 4, and 6 convolutional layers of varying widths. This was done to ensure that my small model performed well for its size. The best performing model consisted of four convolution layers with 3x3 kernels, a stride of 1 and padding to maintain the convolution size. There were pooling layers after the second and fourth convolutional layers, with with 2x2 kernels and stride 2. The number of kernels/units per layer was 32, 32, 64, 64, 512. There was batch normalization before each pooling layer. The model was trained with Adam optimizer and a learning rate of .001.

**Dataset and training:** The model was trained on CIFAR-10 (Krizhevsky et al., 2009). CIFAR-10 is similar to MNIST and fashion (e.g., used in chapter 3) in terms of the number of items and classes, although it is a more difficult task as it used three colour input channels, rather than a single grey-scale channel. CIFAR-10 contains 32x32x3 colour images from 10 real world categories (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). There were 38000 images in the training set and 9600 images used for validation during training. Training was in batches of 32 images for 10 epochs. The validation loss was lowest after 8 epochs so the model was used as saved after 8 epochs. Following training the model was presented with a test set of 12000 images (not in the training or validation set). Test accuracy was .76 (9147 images). Although Zhou et al. (2018b) does not report the overall accuracy for their models, figure 4a in their paper shows class accuracy were in the range .4 to .75, which are similar to my model (for accuracy per class see table 6.1). The activation of all hidden units was recorded for all correct items in the test set. There were 13 dead relu units in the model.

| | Items | Correct | Accuracy |
|---|---|---|---|
| 0. Airplane | 1181 | 787 | .74 |
| 1. Automobile | 1188 | 988 | .83 |
| 2. Bird | 1232 | 848 | .69 |
| 3. Cats | 1274 | 563 | .44 |
| 4. Deer | 1179 | 900 | .76 |
| 5. Dog | 1230 | 744 | .60 |
| 6. Frog | 1123 | 1017 | .91 |
| 7. Horse | 1217 | 1006 | .83 |
| 8. Ship | 1189 | 1115 | .94 |
| 9. Truck | 1187 | 1088 | .92 |
| **Total** | 12000 | 9147 | .76 |

Table 6.1: **Accuracy per class for the CIFAR 10 test set.**

#### 6.2.2.2 Selectivity measures

Selectivity scores are calculated for each class (on correct items only), for each unit in the unlesioned model. A unit's selectivity is given for the class with the highest selectivity score.

**Class Conditional Mean Activation (CCMA)** is a selectivity measure that was used in both Zhou et al. (2018b) (referred to as class-selectivity) and Morcos et al. (2018) to test single-unit selectivity in CNNs. The CCMA for class $A$ compares the mean activation of all images in class $A$, with the mean activation of all images not in class $A$:

$$CCMA = \frac{\text{mean(class } A\text{) - mean(not } A\text{)}}{\text{mean(class } A\text{) + mean(not } A\text{)}} \tag{6.1}$$

Values can be in the range -1 to 1, with positive values indicating that the given class' mean is higher than the mean of other classes, negative values indicating that the given class has a lower mean than the other classes (e.g., the unit is selectively OFF for that class) and a score of zero where the given class has an equal mean to the other classes. The most selective class has the highest (positive) score and as such, will always be ON selective (e.g., positive values).

**Precision** is a measure of the proportion of items above a given threshold from a given class (e.g., Zhou et al., 2015; Rafegas et al., 2017, 2020; Leavitt and Morcos, 2020b).

$$Precision = \frac{\text{number of items from class } A}{\text{100 most active items}} \tag{6.2}$$

Here the threshold is set such that there are 100 items above the threshold unless there are fewer than 100 with an activation > 0.0, in which case the threshold is set at 0.0. A similar measure based on the most active 60 items was used in Zhou et al. (2015). Precision did not test for OFF units.

**Maximum Informedness (Max-info)** identifies the class and threshold where the highest proportion of images above the threshold are from that class (i.e., sensitivity or recall) and the lowest proportion of images above the threshold are not from that class (i.e., specificity, Powers, 2011). The informedness is computed for each class at each threshold, with the highest value selected, given by:

$$MI = max(Sensitivity + Specificity - 1)$$

Max-info did not test for OFF units.

#### 6.2.2.3 Lesioning measures

Note: *n* correct refers to the number of items for a given class that are correct, not the total number of correct items from all classes.

**Class-accuracy-drop** is the measure of class-specific effects of lesioning used in Zhou et al. (2018b). For each class, the difference between accuracy before and after lesioning is given as a proportion of the accuracy prior to lesioning.

$$\text{class-accuracy-drop} = \frac{n \text{ correct after lesioning}}{n \text{ correct before lesioning}} - 1 \tag{6.3}$$

Since a class may improve its accuracy after lesioning, this measure can take values between -1 and 1. For example, if a class accuracy was 60%, which fell to 45% after lesioning, the class-accuracy-drop would be -25%. The max-class-drop is the class with the largest drop in accuracy, which will be the class with the lowest, negative number.

**Total-accuracy-drop** is used as a measure of the effect of lesioning in both Morcos et al. (2018) and Zhou et al. (2018b). This measure is *not* class-specific, and gives only the change in total accuracy, given as:

$$\text{Total-accuracy-drop} = \frac{\frac{\text{n correct after lesioning}}{\text{n items}}}{\text{accuracy before lesion}} \tag{6.4}$$

#### 6.2.2.4 Relationships between unit attributes

**Selectivity and total-accuracy-drop:** The relationship between selectivity and total-accuracy-drop was calculated with Spearman's rank correlation coefficient. Selectivity values were from the class with the highest selectivity for each unit.

**Selectivity and max-class-drop:** Spearman's rank correlation coefficient was calculated to investigate the relationship between selectivity and max-class drop. The max-class-drop is compared with the selectivity of the corresponding class (e.g., the class with the largest drop in accuracy). Note, this is different to Zhou et al. (2018b) where the max-class-drop was compared with the selectivity of the most selective class.

**Total-accuracy-drop and max-class-drop:** Pearson product-moment correlation coefficients were calculated to measure the relationship between the change in total accuracy after lesioning and the the drop in accuracy for the class with the biggest drop (max-class-drop). I expect these to be somewhat related, since the total-accuracy-drop is the sum all class-accuracy-drops. However, any association should be modest if max-class-drop is to identify specialised units.

Note: whilst the values for max-class-drop will differ in analysis 1 and 2, due to different measures of the class-specific effects of lesioning, the values for the total-accuracy-drop are the same for all parts of this chapter.

### 6.2.3 Results

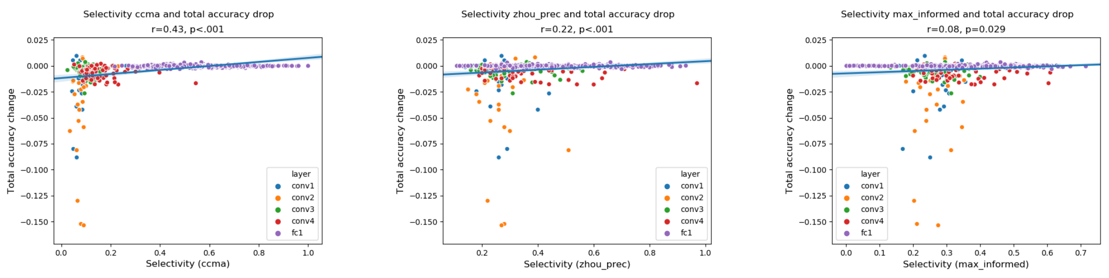
| | Conv 1 mean | Conv 2 mean | Conv 3 mean | Conv 4 mean | Fc 1 mean | Total mean (sd) | Total min. | Total max. |
|---|---|---|---|---|---|---|---|---|
| CCMA | .20 | .09 | .10 | .16 | .55 | .44 (± .23) | .03 | 1.0 |
| Precision | .25 | .27 | .29 | .34 | .49 | .44 (± .17) | .11 | .97 |
| Max-info | .21 | .28 | .27 | .33 | .38 | .36 (± .14) | 0.0 | .72 |
| Total-acc-drop | -.01 | -.03 | 0.0 | -.01 | 0.0 | 0.0 (± .01) | -.15 | .01 |
| max-class-drop | -.07 | -.18 | -.06 | -.07 | -.01 | -.03 (± .06) | -.77 | 0.0 |

Table 6.2: **Descriptive statistics for selectivity and lesioning**. ± indicates the standard deviation.

The levels of selectivity are fairly low (see table 6.2). The average selectivity scores for the whole model are .44, .44 and .36 for CCMA, precision and max-info respectively. These values are low when compared to the levels of selectivity seen in previous chapters (e.g., the iris dataset in chapter 3, section 3.4). The average levels of selectivity are low in early layers and highest in the final fully connected layer, consistent with previous studies (Morcos et al., 2018; Zhou et al., 2017), although the lowest average CCMA is in conv 2. The average total-accuracy-drop from lesioning a single unit is -0.003 (sd = .01), although biggest drop is -.15 and the largest increase is .1. Units in conv 2 had the highest average total-accuracy-drop. Conv 2 also had the largest mean max-class-drop. The average max-class-drop is -.03 (sd = .06) and the largest was a drop of -.77, showing that lesioning a single unit can critical for identifying over three-quarters of a class. This max-class-drop is significantly larger than the biggest max-class-drop reported in Zhou et al. (2018b), which was .5.

#### 6.2.3.1 Correlations between total-accuracy-drop and selectivity measures

To test the relationship between a unit's selectivity and overall importance, I ran Spearman correlations between the total-accuracy-drop and the selectivity of the most selective class for each unit, for each selectivity measure. There were significant positive correlations for all three selectivity measures. CCMA had a moderate correlation with total-accuracy-drop ($r(702) = .43$, $p < .001$), and there weak correlations for Precision ($r(702) = .22$, $p < .001$) and Max-info ($r(702) = .09$, $p = .02$). Positive correlation mean that increases in selectivity were associated with a decrease in the total-accuracy-drop. This is consistent with previous research showing that highly selective units are less important for overall accuracy than units with lower selectivity (Morcos et al., 2018; Zhou et al., 2018b).



a) CCMA b) Precision c) Maximum Informedness

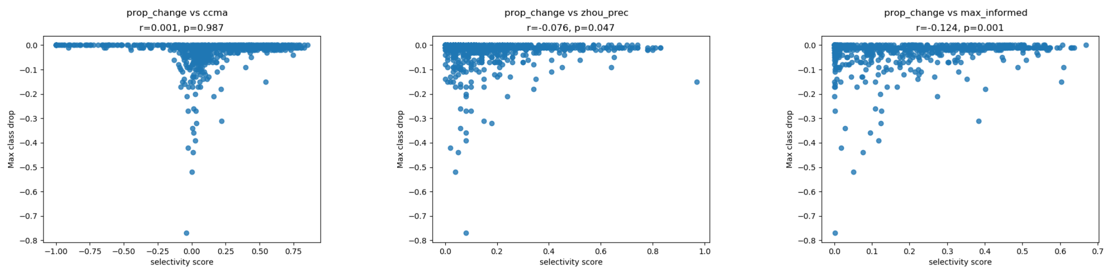
Figure 6.1: **Spearman correlations between the most selective class per unit and total-accuracy-drop, for each selectivity measure**. Selectivity scores are shown on the x-axis and total-accuracy-drop is sown on the y-axis. Colours denote units on different layers: conv 1 = blue; conv2 = orange; conv 3 = green; conv 4 = red; fc 1 = purple.

#### 6.2.3.2 Agreement between class-accuracy-drop and selectivity

In my model, the class with the largest class-accuracy-change (e.g., the max-class-drop) is typically *not* the class with the highest selectivity score. For CCMA, the class with the highest score is also the max-class-drop class for 137 of 704 units (19.4%). There are also 137 units where the class with the highest Max-info is also the max-class-drop (although these are not always the same units where CCMA agreed with max-class-drop). For Precision there were 116 units (16.5%) where the most selective class was also the max-class-drop. In Zhou et al. (2018b) they do not discuss this situation, and it is possible that in their model they found greater agreement between the max-class-drop and most selective class. With this in mind, I analysed the relationships between the max-class-drop and the selectivity score for the corresponding class (i.e., the selectivity of the class with the max-class-drop).

#### 6.2.3.3 Correlations between selectivity and max-class-drop

Figure 6.2 shows that max-class-drop had a weak negative correlations with Max-info ($r(702)$ = -.12, $p$ < .001) and Precision ($r(702)$ = -.08, $p$ < .047). The correlation between max-class-drop and CCMA was not significant ($r(702)$ = .001, $p$ = .99). The negative correlation means that increases in the selectivity score were associated with a decrease in accuracy (i.e., a larger max-class-drop). This means that highly selective units typically have a larger max-class-drop than units with low selectivity). This association is in the same direction as reported by Zhou et al. (2018b). Zhou et al. (2018b) reported a significant correlation with CCMA, which I do not find here. This could be because I used the selectivity scores for the max-class-drop class, rather than the most selective class.

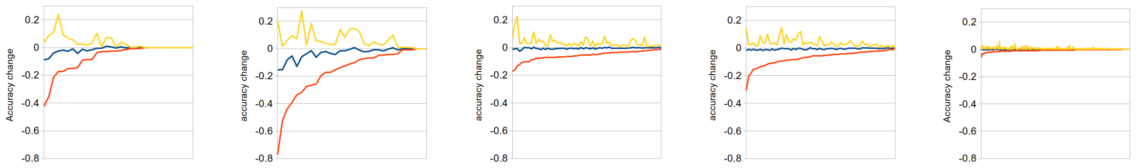


a) CCMA b) Precision c) Maximum Informedness

Figure 6.2: **Correlation between Max-class-drop (y-axis) and the corresponding class' selectivity (x-axis)**. $r$ is the Spearman's rank-order correlation and p is the p-value.

#### 6.2.3.4 Lesion accuracy curves

Figure 6.3 shows the total-accuracy-drop and max-class-drop for each unit. Units are sorted along the $x$-axis, from largest max-class-drop to smallest (red line). The total-accuracy-drop is shown in blue, and the class with the smallest max-class-drop (or largest class increase) is shown in yellow.



a) conv 1 b) conv 2 c) conv 3 d) conv 4 e) fc 1

Figure 6.3: **Accuracy drop curves for each layer in the model**. For each layer, the units are ordered on the $x$-axis from the unit with the largest max-class-drop to the smallest. Change in accuracy is shown on the $y$-axis. Red lines denote max-class-drop, blue lines denote total-accuracy-drop and yellow lines denote min-class-drop (or max-class-increase). There are 512 units in fc1; 64 kernels in conv 3 and conv 4 and 32 convolutional kernels in conv 1 and conv 2.

The first thing to note in figure 6.3, is that the most important units in terms of overall accuracy (i.e., where the blue line drop below zero in layers conv 1 and conv 2) are also the units with the largest max-class-drop (red line). The results in table 6.2 showed that conv 2 had the highest mean total-accuracy-drop and max-class-drop, but here I can see that it is the same units which are driving this. This is problematic, as it suggests that units with the highest max-class-drop are not necessarily specialised for the max-class-drop class, but rather, they are general-purpose and contribute to many classes. However, there are better candidates for specialised units are seen on the left side of the plots for conv3 and conv4. These have a significant max-class-drop (around -.15) and negligible effect on total accuracy. There are also many units where lesioning does not lead to a significant drop in accuracy for any class, and so the red and blue lines are both close to zero. These are seen on the right side of each layer's plot and across the whole of fc 1. This is somewhat surprising as fc 1 is the layer that might be expected to have the most 'specialised' units as selectivity is highest in this layer.

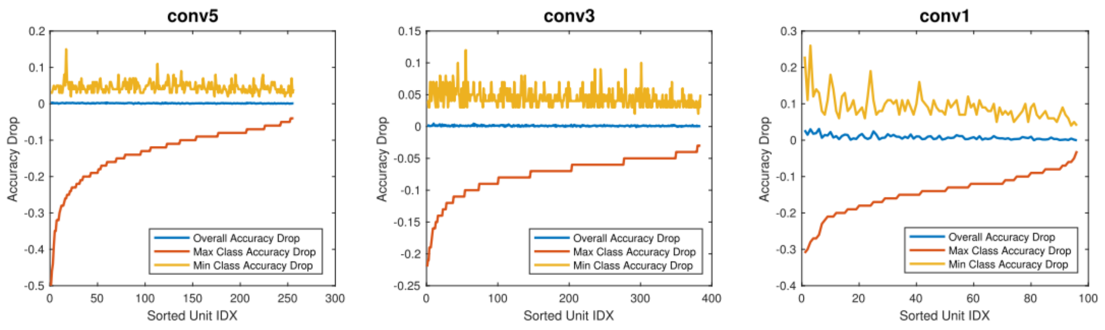


Figure 6.4: **Figure reproduced from Zhou et al. (2018b).** It shows accuracy curves fro three layers in their AlexNet-Places model. Units are sorted from largest max-class-drop to smallest (red line). The blue line shows the total-accuracy-drop and the yellow line shows the change in accuracy for the class with the largest increase. Note: layers are in reverse order.

Figure 6.4 is taken from Zhou et al. (2018b) and it shows the change in accuracy for units in selected layers for one of their models. In comparison to my model, I see that total-accuracy-drop in their model is very stable (blue line). There appear to be no units which lead to a large total-accuracy-drop. I also see that max-class-drop (red line) stays below zero for all units, even on the right side of the plot. This means that there was a significant loss in performance to at least one class for every unit. Together, this shows that most of the units in these layers are specialised for a particular class or subset of classes: a lesioned unit has a significant impact on the accuracy of at least one class, but has minimal impact on overall accuracy.

#### 6.2.3.5 Correlations between total-accuracy-drop and max-class-drop

The lesion accuracy curves (figure 6.3) show that in my model, the units largest max-class-drop are also the units with the largest total-accuracy-drop. To investigate this relationship further, Pearson correlations were calculated between the max-class-drop and total-accuracy-drop for all layers. As shown in fig 6.5, there was a very strong positive correlation ($r(702) = .89$, $p < .001$). This means that as the class-specific importance of a unit increases (i.e., larger max-class-drop), so does the overall importance of the unit (i.e., larger total-accuracy-drop). Put differently, units with a high max-class-drop are likely to also have many items from other classes that failed. For example, the unit with the largest max-class-drop was also the unit with the largest total-accuracy-drop (see the bottom left of figure 6.5). This means that max-class-drop is not identifying units that are specialised for a particular class or subset of classes as suggested by Zhou et al. (2018b), but rather units with the highest max-class-drop are non-specialist, general-purpose units that contribute to many classes.

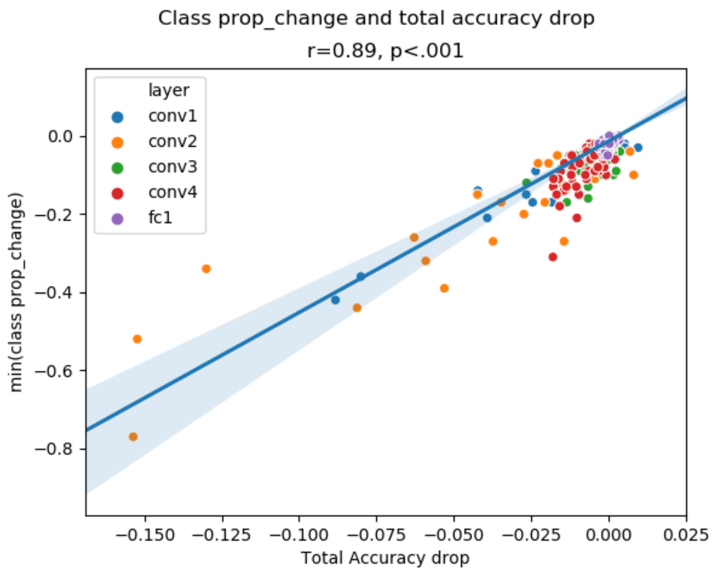


Figure 6.5: **Pearson correlations between max-class-drop and total accuracy after lesioning.** Total-accuracy-drop is sown on the x-axis, and is given as a proportion of the accuracy in the unlesioned model. Measures of lesioning are on the y-axis and show the drop in accuracy for the class with the largest drop in accuracy according to the lesioning measure.

#### 6.2.3.6 Units of interest

Figure 6.6 shows the units with the largest max-class-drop, which are also the units with the largest total-accuracy-drop. For both units, the most selective class is not same as the max-class-drop. For unit 2.10, the max-class-drop is class 4, yet all selectivity measures score class 7 highest. For unit 2.2, the max-class-drop us class 5, but all selectivity measures score class 1 highest. This figure demonstrates that the units with the highest max-class-drop are not specialised for a particular class.

Figure 6.7 shows the unit with the joint highest CCMA score. There is only a single active item for this unit, and as such it has a CCMA of 1.0 for class 3. However, although this unit scores

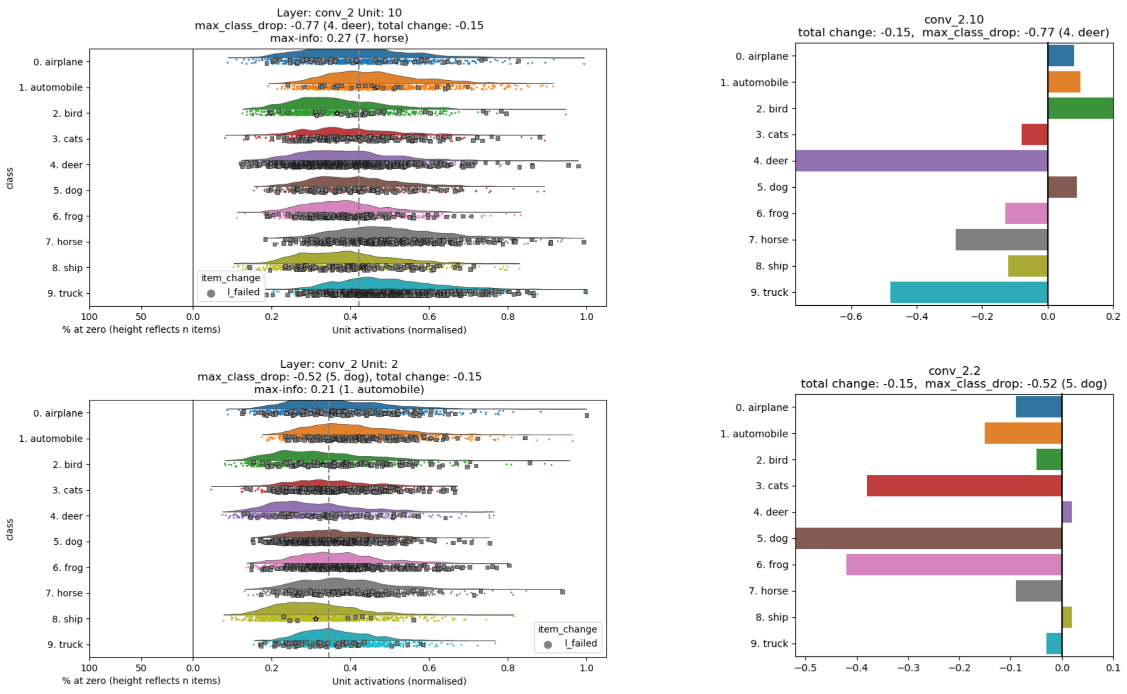


Figure 6.6: **Plots showing the units with the highest max-class-drop.** These units also have the highest total-accuracy-drop. *Left panel*: Shows the normalised activations on the $x$-axis, the $y$-axis is ordered by class. Each point represents a single item, with a probability density function fitted above. The proportion of items per class with zero activation is shows by bars on the far-left. Items that failed when the unit was lesioned are shown as large grey squares (note: items that were incorrect in the full model, but correct after lesioning are not shown). Dashed grey lines show the threshold for the Max-info for the class with the highest max-info score. The selectivity (Max-info) class and score are shown above the plot. *Right panels*: The class-accuracy-drop values for each class. *top row*: Conv 2.10 has a total-accuracy-drop of -.15 and a max-class-drop of -.77 for class 4. The most selective class is class 7. *Bottom row*: Conv 2.2 also has a total-accuracy-drop of -.15 and a max-class-drop of -.52 for class 5. Class 1 has highest selectivity.

high for selectivity, lesioning shows that it is not specialised or important for any class. A similar story is seen for the unit with the highest Max-info score, shown in figure 6.8. The unit has a Max-info score of .72 for class 1, with CCMA of .55 and precision of .91 for the same class. Again, lesioning has no significant impact on this unit. These units demonstrate that high selectivity does not equate with loss in accuracy for the selective class. However, the unit with the highest precision (figure 6.9 does appear to show some specialization. The most selective class is class 1, with precision of .97, Max-info of .60 and CCMA of .55. The max-class-drop is also class 1, with a substantial drop of 15% after lesioning. There is also a small drop in accuracy of 5% for class 3 and a drop for class 1 of around 1%. The total-accuracy-drop is 2%.

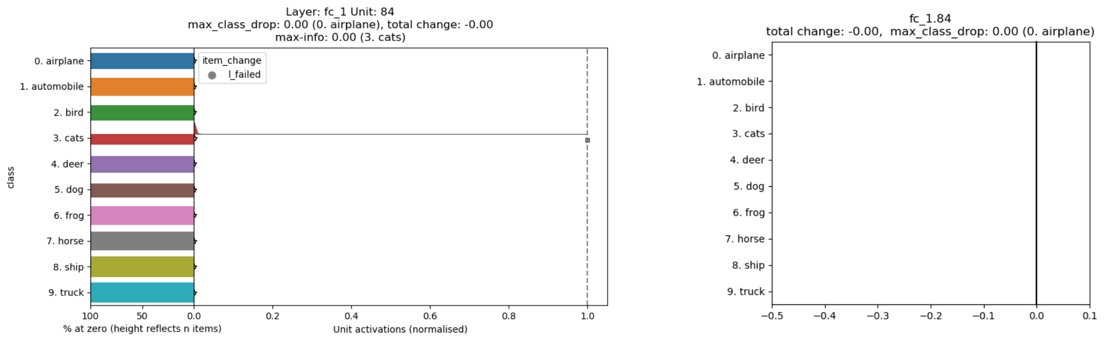


Figure 6.7: **Fc1.84: This unit has the joint highest CCMA score** (class 3: 1.0).

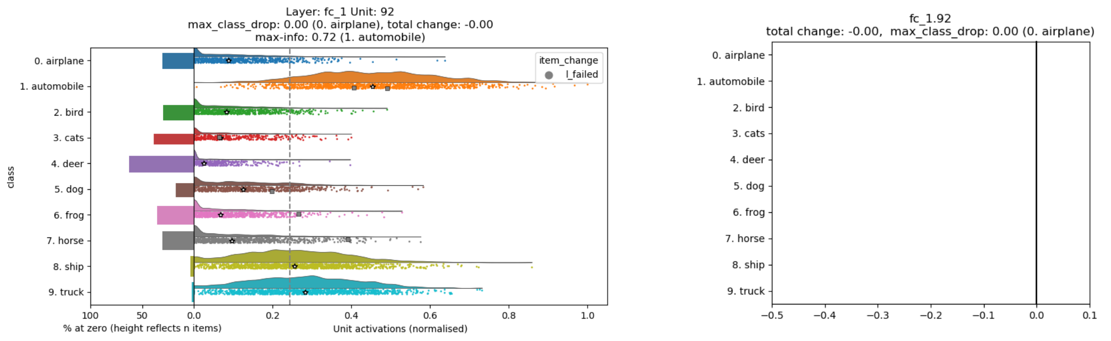


Figure 6.8: **Fc1.92: This unit has the highest Max-info score** (class 1: .72).

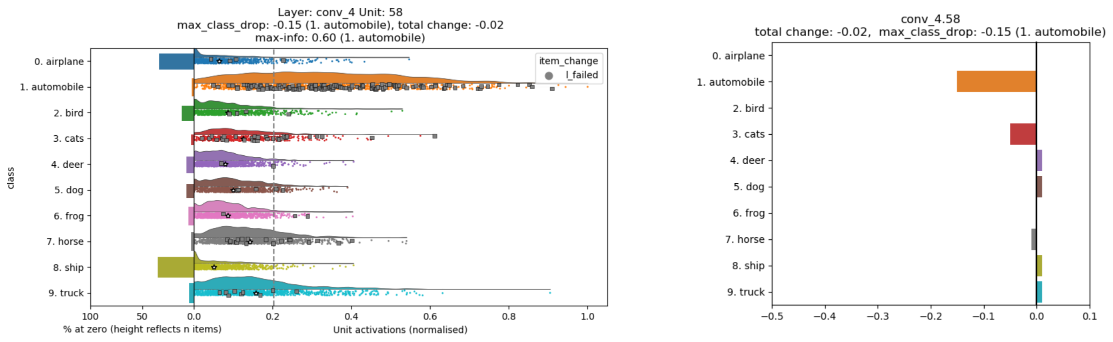


Figure 6.9: **Conv4.58: This unit has the highest Precision score** (class 1: .97).

### 6.2.4 Discussion

This study investigated class-accuracy-drop, a measure of the class-specific effects of lesioning proposed by Zhou et al. (2018b). I applied class-accuracy-drop to a small CNN trained on CIFAR-10, to see whether it would identify units that are specialised for a particular class. I also aimed to test whether max-info or precision were better predictors of class-accuracy-drop than CCMA, a selectivity measure used by Zhou et al. (2018b). I measured the selectivity all units and then lesioned each unit in turn, recording the change in total accuracy as well as the change in accuracy for each class. I found that all three selectivity measures were positively correlated with the total-accuracy-drop. This was consistent with the view that the most important units are not the most selective (Morcos et al., 2018; Zhou et al., 2018b). Also consistent with Zhou et al. (2018b) were the significant weak negative correlations between max-class-drop and the same class' selectivity scores for Max-info and precision (the association with CCMA was not significant). The significant correlations mean that selectivity is predictive of the importance of a unit for a particular class. However, I also find that the class with the largest accuracy drop was also the most selective class in fewer than 20% of all units. Whilst this is above chance levels, it is far from good predictive power. The fact that the size of the drop in accuracy for the class with the largest drop can be predicted from class selectivity scores is only useful if one knows which class's selectivity score to use.

An unexpected finding was a very strong positive correlation between max-class-drop and total-accuracy-drop. This is problematic as it means that in my model, a high max-class-drop was not indicative of importance to one class, but to many. This motivates a re-think of the approach to measuring the class-specific effects of lesioning.

Applying class-accuracy-drop to my model reveals interesting differences in the representations learned by units in my model, compared to those in Zhou et al. (2018b). Their lesion accuracy curves show that no unit led to a large drop in total accuracy, while I find a small subset of units in conv 1 and conv 2, which are of high importance to many classes. In Zhou et al. (2018b), their lesion accuracy curves suggest that there was a drop in accuracy for at least one class in *all* units for they layers that they have reported. In contrast, there are many units in conv 1 and fc 1 in my model where no class is significantly impacted. This suggests that the effects of lesioning are more varied in my model than in Zhou et al. (2018b).

One possible reason for the different representations in my model is that it is considerably smaller than the model used by Zhou et al. (2018b). On average, units in my model which has 704 kernels and units, make a larger contribution to overall accuracy in AlexNet, which has 9568. Similarly, differences in datasets also explain differences in my results. A change in accuracy for one class makes a larger contribution to a change in total accuracy for CIFAR-10, with ten classes, than for Places, with 365 classes, or imageNet with 1000! Zhou's models have around 14 times more units that mine, but their datasets have 30 to 100 times as many classes. Therefore, in Zhou's model there are fewer units per-class than in my model. This perhaps explains why for their model, a sizable max-class-drop was the norm: there are fewer units available for each class, so units have to become specialised. In comparison, my model is more robust to the the effects of lesioning: there are a greater number of units available to represent each class, so many units do not need to specialise. Nevertheless, even in the absence of any strong specialization, some units may be more specialised than others.

Zhou et al. (2018b) did not report a correlation between max-class-drop and total-accuracy-drop, but figures from their study suggest that total-accuracy-drop was stable in their model, with no unit leading to a significant drop in overall accuracy. Similarly, most units had a large max-class-drop, justifying their claim that most units were specialised. The accuracy drop curves (figure 6.3) suggest that units in my model can be grouped into three types: some are important to many classes, shown by the large total-accuracy-drop; others are not uniquely important to any class, indicated by a negligible max-class-drop; finally a set of units which are specialised, having a sizable max-class-drop and small total-accuracy-drop. My results highlight the fact that max-class-drop is unsuitable for differentiating between units which are specialised for a particular class and more broadly tuned units of importance to a many classes. In both cases the unit will have a high max-class-drop. This limitation of class-accuracy-drop was not apparent in Zhou et al. (2018b), but having identified it, it appears that class-accuracy-change is not a reliable measure of the effects of lesioning. However, a measure that can reliably quantify the degree of specialization of units and help identify the most specialised would be a useful tool. To identify the most specialised units in my model requires identifying units where the max-class-drop is large in relation to the total-accuracy-drop. In the second part of this chapter, I propose drop-prop - a measure of the class-specific effects of lesioning that identifies the units where the max-class-drop accounts for the highest proportion of the total-accuracy-drop.

## 6.3 Analysis 2

### 6.3.1 Introduction

In analysis 2 I re-analyse the lesioning data from the same model used in analysis 1. I use drop-prop, a different measure of the effects of lesioning and compare the results with class-accuracy-drop, the measure used in analysis 1. I find that these two measures highlight different aspects of a unit's importance for a given class and can be used in conjunction to identify the units that are most specialised. I also investigate whether Max-info or precision are better predictors of class-importance than CCMA. I find that CCMA has stronger correlations with drop-prop than Max-info or precision. However, logistic regression models trained to predict which class will have the max-class-drop have a better performance if using Max-info or precision scores than from using CCMA.

### 6.3.2 Methods

#### 6.3.2.1 Lesioning measures

**Drop-proportion (drop-prop)** is a new measure of the class-specific effects of lesioning, proposed in response to limitations of class-accuracy-drop. Drop-prop measures how much each class contributes to any drop in overall accuracy. For classes with no decrease in accuracy (or an increase), drop-prop will return a value of zero. For classes with a decrease in accuracy, this measures gives the decreases as a proportion of the sum of all classes where accuracy dropped (i.e., "what proportion of classes where accuracy dropped does this class account for"?). Values for drop-prop are in the range 0-1 and the max-class-drop is the class with the highest value (e.g., the class that accounts for the largest proportion of the drop in accuracy). Note that this is scored in the opposite direction to class-accuracy-drop where a score of -1 indicated the largest drop in accuracy.

**Algorithm 3** Calculate drop-prop

```
if $n$ correct after lesioning $\geq n$ correct before lesioning then
    Drop-prop = 0
else
    Drop-prop = $\frac{\text{difference in items correct}}{sum(\text{classes where items correct dropped after lesioning})}$
end if
```

#### 6.3.2.2 Relationships between unit attributes

**Logistic regression:** An ordinary least squares logistic regression classifier was trained to predict which class would have the largest drop in accuracy (the max-class-drop class) from the ten selectivity scores per unit. The method is different to that used in Zhou et al. (2018b). They trained regression models on the conv 5 layer from three AlexNet models and tested it on the conv 5 layer of two different AlexNet models. Here the activations from all layers are used. The order of the unit was shuffled and then split: the models were trained on 80% of the units and tested on the remaining 20%. The regression results were compared with a dummy classifier that always predicted the most frequent max-class-drop label in the training set.

### 6.3.3 Results

| | Conv 1 mean | Conv 2 mean | Conv 3 mean | Conv 4 mean | fc 1 mean | total mean (sd) | total min. | total max. |
|---|---|---|---|---|---|---|---|---|
| Drop-prop | .31 | .39 | .44 | .45 | .56 | .52 (± .22) | 0.0 | 1.0 |
| Class-acc-drop | -.07 | -.18 | -.06 | -.07 | -.01 | -.03 (± .06) | -.77 | 0.0 |

Table 6.3: **Descriptive statistics for selectivity and lesioning**. Class-acc-drop (class-accuracy-drop, from analysis 1) is shown for comparison. All selectivity scores and total-accuracy-drop scores are the same and in analysis 1.

For drop-prop, the average max-class-drop was .52 (sd = .22) which means that one class typically accounted for about half of the drop in accuracy. There is an increase in mean drop-prop scores through the layers. In conv 1, a single class typically accounted for a third of items that failed, but in fc 1, a single class typically accounted for over half of all items that failed. This suggests that units were more specialised for one class in later layers than early layers.

#### 6.3.3.1 Correlations between total-accuracy-drop and max-class-drop

In the previous section I found a very strong relationship for class-accuracy-drop and total-accuracy-drop that was problematic. Therefore the first correlations I tested was between total-accuracy-drop and max-class-drop (the highest scoring class from drop-prop, see figure 6.10). There is a weak positive correlation ($r(702) = .14$, $p < .001$). Since drop-prop is scored in the opposite direction to class-accuracy-drop, this positive correlation therefore means that increases in total-accuracy-drop (i.e., fewer items fail) are associated with increases in the drop-prop (i.e., the class accounts for a higher proportion of the failures).

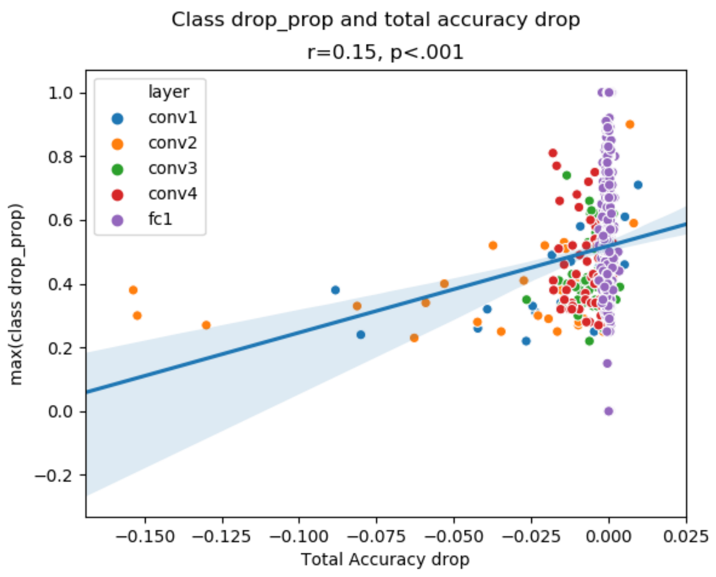
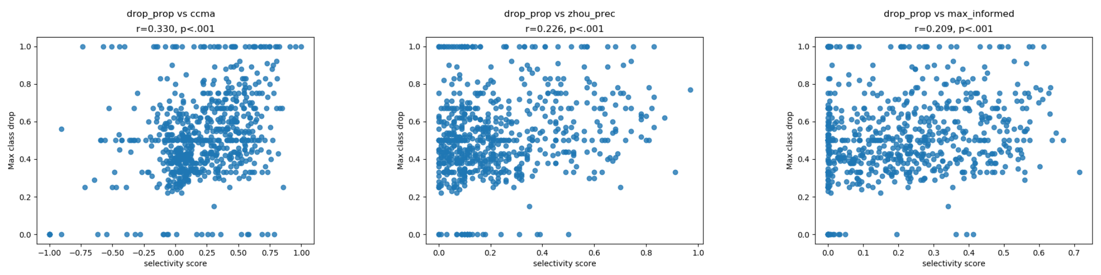


Figure 6.10: **Pearson correlations between max-class-drop and total accuracy after lesioning for drop-prop**. Total-accuracy-drop is shown on the x-axis, and is given as a proportion of the accuracy in the unlesioned model. Max-class-drop is on the y-axis. For drop-prop, a low value close to zero means that the max-class-drop accounted for a small proportion of all items that failed, and a high value of 1.0 means that the max-prop-class was the only class where performance dropped.

#### 6.3.3.2 Correlations between selectivity and max-class-drop

I ran Spearman correlations to investigate whether the effects of lesioning as measured with drop-prop are associated with the three selectivity measures. As with analysis 1, correlations here are between the max-class-drop and the selectivity of the corresponding class, not necessarily the most selective class (see table 6.4 for details). All selectivity measures were significantly correlated with drop-prop. There were weak positive correlations between drop-prop and CCMA ($r(702) = .33$, $p < .001$); Precision ($r(702) = .23$, $p < .001$) and Max-info ($r(702) = .21$, $p < .001$). These positive correlations mean that increases in selectivity are associated with increases in the proportion of the drop in accuracy that the max-prop-class accounts for. Therefore, the drop in accuracy for class-$x$ is likely to account for a higher proportion of the total-accuracy-drop if the units has high selectivity for class-$X$ than in a unit with low selectivity for that class. This is consistent with Zhou et al. (2018b) claim that selectivity measures predict the effects of lesioning.



a) CCMA b) Precision c) Maximum Informedness

Figure 6.11: **Correlation between drop-prop max-class-drop (y-axis) and the corresponding class' selectivity (x-axis)**. r is the Spearman's rank-order correlation and p is the p-value.

#### 6.3.3.3 Agreement between lesioning measures and selectivity

Having established a relationship between the magnitude of selectivity scores and drop-prop scores, it is worth noting that the class which accounts for most of the accuracy drop is often different to the most selective class. Table 6.4 reports how often the max-prop-class was also the most selective class. For all three selectivity measures, the most selective class is in agreement with drop-prop more often than with class-accuracy-drop, but agreement is still below 32% for all measures. If the classes were all of equal size, then the class with the highest score for class-accuracy-drop would also have the highest drop-prop. However, since they are not of equal size, there are units where these measures disagree too. For example, unit conv 2.2 (shown in figure 6.6, bottom row) has class 6 as the highest scoring class for drop-prop, but class 5 for class-accuracy-drop. There are more items that fail from class 6 (427 items) than for class 5 (390 items), so class 6 account for a greater proportion of items that failed. But there are fewer correct items in class 5 (744) than class 6 (1017), so there is a greater change in accuracy for class 5 than class 6.

| | Drop prop | class-accuracy drop |
|---|---|---|
| CCMA | 211 (31%) | 137 (20%) |
| Prec | 194 (28%) | 116 (17%) |
| Max info | 203 (29%) | 137 (20%) |

Table 6.4: **A count of the number of units where the class with the highest selectivity was also the max-class-drop**. class-accuracy-drop is shown for comparison. Agreement as a percentage of the 691 active units is given in parenthesis.

#### 6.3.3.4 Logistic regression

A logistic regression classification model was trained to predict the identity of the max-class-drop class for each unit from the vector of selectivity scores for each unit. It was trained on 80% of the units and tested on the remaining 20%. For comparison, a dummy-model predicted the class that was most frequently max-class-drop for all units. The dummy model results differ as the most frequent max-class-drop class differed for class-accuracy-drop and drop-prop. For drop-prop, all selectivity measures outperformed the dummy model. For class-accuracy-drop, none of the measures outperformed the dummy model.

| | Drop prop | class-accuracy drop |
|---|---|---|
| CCMA | .33 | .29 |
| Precision | .42 | .23 |
| Max-info | .42 | .32 |
| Dummy model | .26 | .48 |

Table 6.5: **Logistic regression classifier accuracy predicting the identity of the max-class-drop class for each unit from the selectivity scores of each unit.** For comparison a dummy model predicted the most frequent class label.

#### 6.3.3.5 Combining drop-prop and class-accuracy-drop

Figure 6.12 shows the units with the highest drop-prop scores. As was the case for class-accuracy-drop in analysis 1, drop-prop does not identify units that are specialised for a particular class. The highest scoring units for drop-prop score 1.0, because accuracy only drops for a single class. However, in each of these cases the number of items that failed from that class was very small. A unit that is specialised for class-$x$ should meet two criteria: 1) many items from class-$x$ fail when the unit is lesioned, 2) most items that fail are from class-$x$. Class-accuracy-drop (as used in the first analysis), was good at identifying units where many items from class-$x$ failed, but was unreliable because it didn't take into account failure from other classes. Drop-prop has proved to be good at the second part: identifying units when failures were mostly from the same class.

However, it is limited because it doesn't take into account the proportion of a particular class that failed.

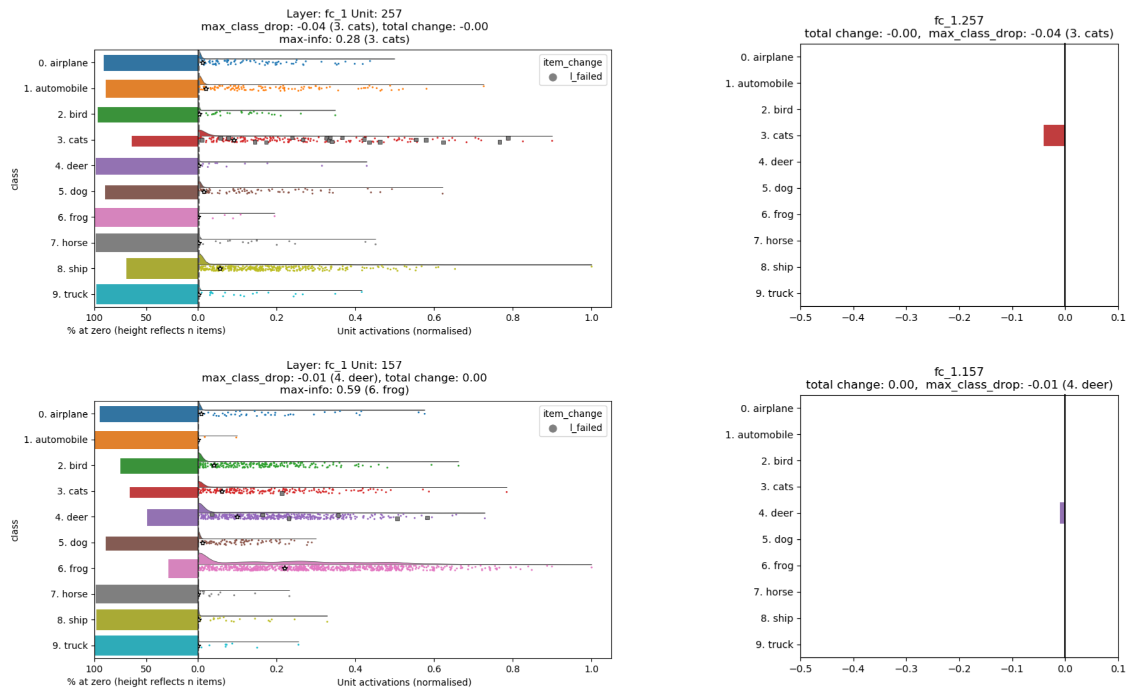


Figure 6.12: **Examples of units with the highest max-class-drop for drop-prop** (1.0).

Figure 6.13 shows the class-accuracy-change plotted against drop-prop. There is a significant weak positive correlation between them, meaning that increases in proportion of the drop in accuracy the the max-class accounts for are associated with fewer items from that class failing. In order to identify units that are specialised, I can use data from both class-accuracy-drop and drop-prop. This can be done by selecting the units where class-accuracy-drop is > .05 to make sure that at least a twentieth of one of the classes fails. Secondly I can select only units where drop-prop is > .5 which ensures that more than half of the failures are from the same class. Applying these two criteria leave 36 units, or .05% of all units. These 36 units are shown in the white area of figure 6.13. The thresholds for class-accuracy-change and drop-prop were arbitrary, and the plot shows that these 'specialised' units are not separate from the other units in any way. Of these units, the highest drop-prop was .89. There were 46 units with a drop-prop of 1.0, but they all had less than .05 class-accuracy-drop. The highest class-accuracy-drop was -.31. There were 8 units with higher class-accuracy-drop, up to -.71, but they all had a drop-prop score below .5. For examples of these 'specialised' units, see figure 6.14).

The right-hand panels in figure 6.14 shows the change in accuracy for each class, and confirms that units are specialised. However, the left-hand panel, showing the levels of activation for each class shows that these units are not very selective, all having Max-info scores below .5. The mean selectivity for these 36 units is below the average for the whole model. For CCMA the mean

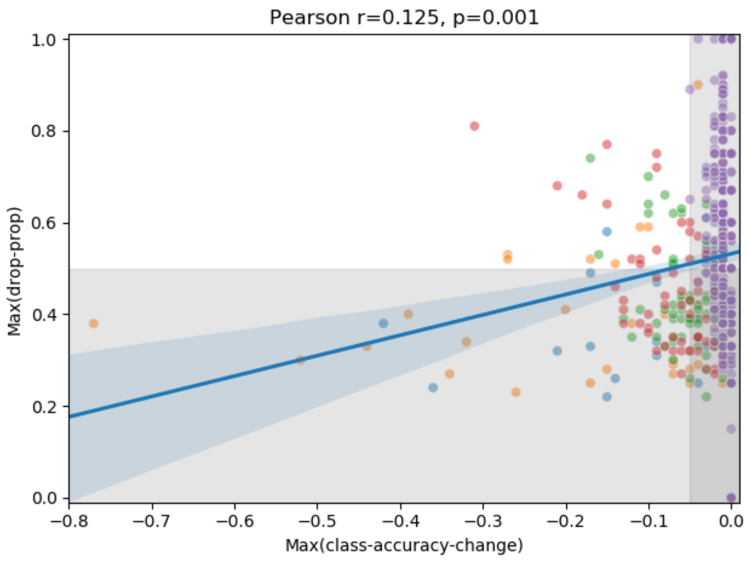
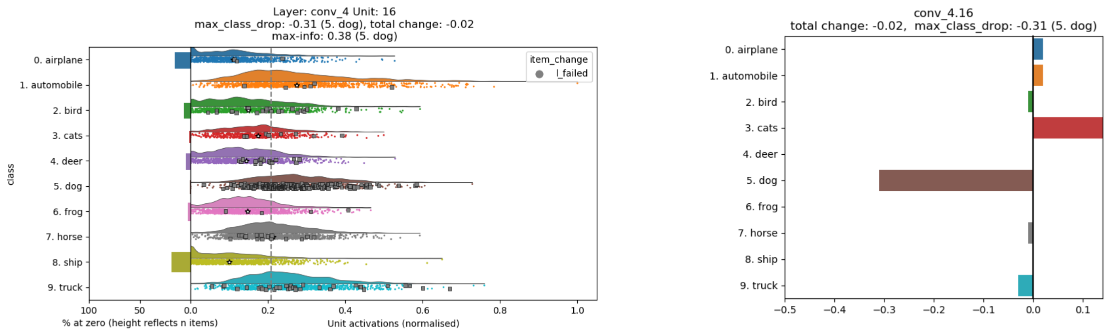
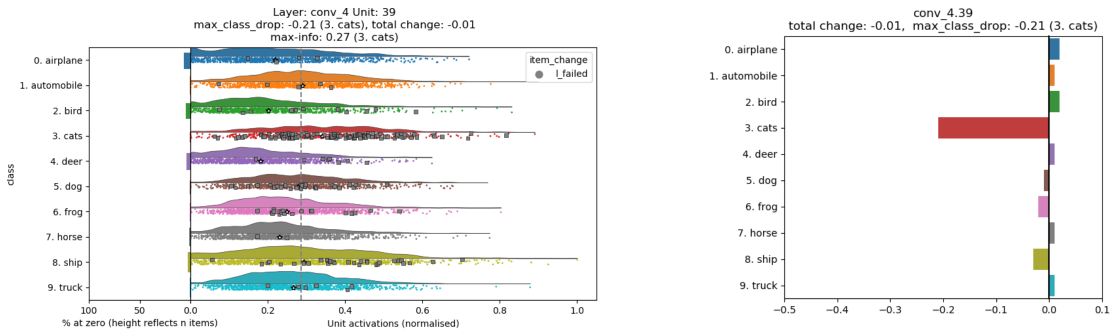


Figure 6.13: **Pearson correlations between the highest scoring class for class-accuracy-change and drop-prop** (or max-class-drop vs max-class-drop). Class-accuracy-change is shown on the $x$-axis, and drop-prop is shown on the $y$-axis. The grey areas indicate units with a class-accuracy-drop <.05 or drop-prop < .5. 668 of the 704 units are in this grey area. The remaining 36 units (in the white area) are considered to be specialised for a particular class.

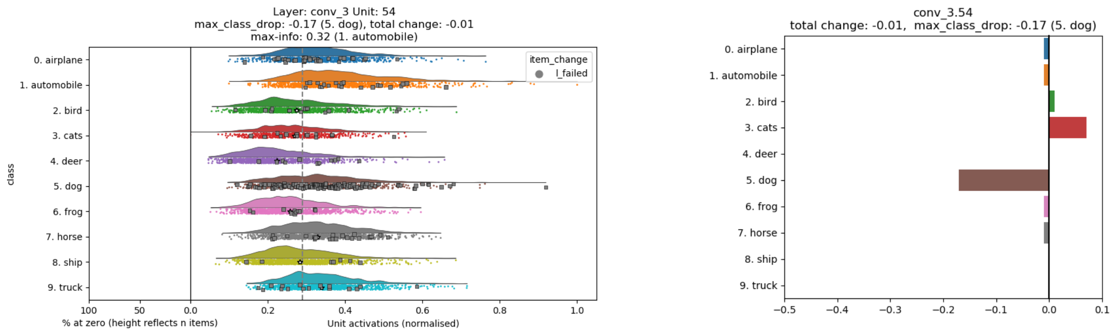
was .15 (sd = .12); the mean Max-info was .32 (sd = .11); and the mean precision was .36 (sd = .17). Of the 100 units with the highest Max-info scores, only two are specialised units. Eight of the top 100 units for precision are are specialised and none of the 100 units with the highest CCMA scores. These units are all dense, with very few items having an activation of zero. There is nothing about the pattern of activations that wold suggest that these units are specialised for a particular class. The agreement between the max-drop-class and the most selective class was still low for these 36 units. The class with the highest Max-info was the same as the max-prop-class for only 15 of these units, CCMA agreed on 13 units and Precision agreed on 9 units. Having identified units where the effects of lesioning show them to be specialised for a particular class, I find that these units have low selectivity, and they are typically *not* most selective for the class that they are specialised for. In this model, selectivity is not a good predictor of unit function.



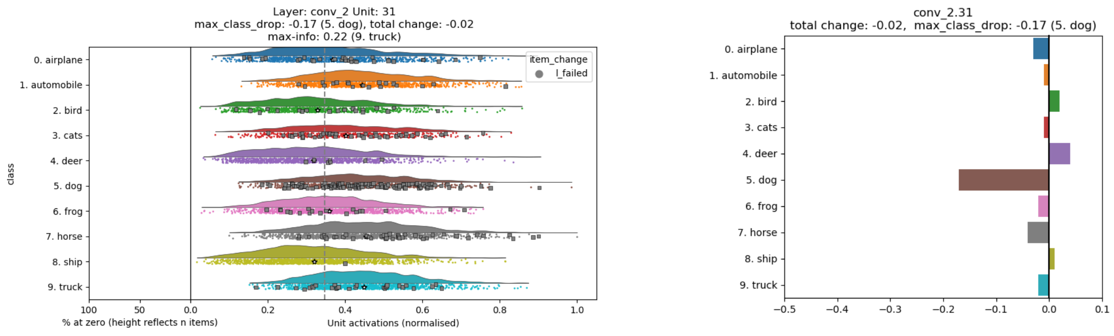
Conv 4.16. Class-accuracy-drop = -.31, Drop-prop = .81



Conv4.39. Class-accuracy-drop = -.21, Drop-prop = .68



Conv3.54. Class-accuracy-drop = -.17, Drop-prop = .74



Conv2.31. Class-accuracy-drop = -.17, Drop-prop = .52

Figure 6.14: **Plots showing 'specialised' units.** These units have many items that fail (class-accuracy-drop > .05); but items that fail are mostly from the same class (drop-prop > .5). The max-class-drop scores in the figures are for class-accuracy-drop.

### 6.3.4 Discussion

I re-analysed the model from analysis 1 using drop-prop, a new measure of the class-specific effects of lesioning. Drop-prop gives the drop in accuracy for each class as a proportion of the sum of all classes where accuracy dropped. This is a different approach to class-accuracy-drop, which gives the change in accuracy for each class relative to the accuracy in the unlesioned model.

In order to asses the suitability of drop-prop I calculated correlations between the change in total accuracy and the max-class-drop. I found a weak positive correlation between total-accuracy-drop and max-class-drop, meaning that decreases in total-accuracy-drop (e.g., fewer items failed) are associated with increases in drop-prop (e.g., max-prop-class accounts for a larger proportion of all items that failed). This means that drop-prop is only weakly associated with total-accuracy-drop, rather than the problematic strong association found for class-accuracy-drop in analysis 1. I also found weak positive correlation between the max-class-drop and all three selectivity measures. This means that increases in selectivity are associated with increases in the proportion of the drop in accuracy that the max-prop-class accounts for. This association gives further support to the use of drop-prop as a useful alternative to class-accuracy-drop.

There was greater agreement between the drop-prop and the selectivity measures about the highest scoring class than there was for class-accuracy-drop and selectivity measures. However, agreement was less than 32% for all measures. This finding pours some cold water on the significant correlations between selectivity and the effects of lesioning, at least insofar as describing a units as ‘selective for’ a given class is predictive of its role. Yes, the level of selectivity is predictive of the effects of lesioning on that class, as shown by the significant correlations; but the agreement scores show that the class the unit is ‘selective’ for is not the class most affected by lesioning. Logistic regression models were trained to predict which class would be most affected by lesioning, from the selectivity scores for all classes. Separate models were trained for each combination of lesioning and selectivity measures. For class-accuracy-drop, dummy models outperformed all regression models, meaning that the regression models had poor predictive power. The regression models for drop-prop all out performed the dummy model, although notably the dummy model performance was lower for these three lesioning measures. Models trained on precision and max-info has 42% accuracy, which is higher than the scores from Zhou et al. (2018b), but still shows that the relationship between selectivity and the class specific effects of lesioning are weak.

A second aim was to examine the relationship between the effects of lesioning and various selectivity measures. I had hoped to identify units with superior predictive power to those used in Zhou et al. (2018b) and Morcos et al. (2018). I found a significant weak negative correlations between class-accuracy-drop and Max-info, and also a slightly weaker one with precision, whilst the correlation with CCMA was not significant. For drop-prop, all three measures had significant correlations, yet CCMA was the strongest, and Max-info was the weakest. It is hard to separate the three in terms of the association between the magnitude of the selectivity and lesioning scores. In terms of the relationship between the identify of the most selectivity class and the class most affected by lesioning: none of the regression models trained on selectivity scores out-performed the dummy regression model for class-accuracy-drop. For drop-prop, all selectivity measures did, although CCMA performed worse than precision and CCMA. Yet, agreement between the class with the highest selectivity and highest drop-prop score was CCMA. As such I suggest that these three selectivity measures are similar in ability to predict the class-specific effects of lesioning. Rather than identifying a measure with a closer relationship, this study has highlighted how weak the correspondence between selectivity and the effects of lesioning are. The correlations are weak and the regression models are wrong more often than they are correct. This disparity between the effects of lesioning and selectivity is clearly apparent in the visualizations of the specialised units shown in figure 6.14. There is nothing in the activation of the units that indicates which classes will have a significant drop in accuracy and which will not. Whilst this is not uniformly true, conv4.58 (shown in figure 6.9 has the highest precision score and CCMA score of the specialised units and the joint highest Max-info. In this unit, max-class-drop has higher activation than other classes. But this was an exception, and in general the specialist units did not have high selectivity.

In summary, analysis 2 found that selectivity measures had a stronger relationship with drop-prop than class-accuracy-drop. This is seen in the stronger correlation strengths, higher regression scores and more frequent agreement about the max class for drop-prop than class-accuracy-drop. This makes selectivity a better predictor of the class-specific-effects of lesioning as measured by drop-prop, than class-accuracy-drop. This does not mean that it is a better measure, but only that what it measures (the proportion of the drop in accuracy that a class accounts for) is more closely related than class-accuracy-drop to the unit's activations as summarised by three selectivity measures. I did not find that Max-info or precision were superior to CCMA as predictors of the effects of lesioning, and I note that although all three measures has some predictive power, the predictive abilities and correlation strengths were weak.

## 6.4 General discussion and conclusions

This study set out to test two claims by Zhou et al. (2018b): that models learn units that are specialised for a particular class; and that selectivity measures are a good predictor of this specialization. I trained a CNN on an image classification task, as was the case for Zhou et al. (2018b), although both the model and the dataset were considerably smaller.

Regarding the claim that models use specialised units, I used class-accuracy-change and drop-prop, two measures of the effects of lesioning, to try identify units that were specialised. I found that neither measure was ideal for the task: class-accuracy-drop identified units where a large proportion of one class failed, but typically, many items from other other classes failed too. Drop-prop identified units where failures were all from one class, but typically only a few items from that class failed. However, using these two measures in combination did allow us to identify a subset of units which could be described as specialised: many items from one class failed, and they accounted for most of the incorrect items. However, whilst most units in Zhou et al. (2018b) appeared to be specialised (e.g., large max-class-drop and small total-accuracy-drop), only .05% of units in my model showed apparent specialization. Although I do not dispute Zhou et al. (2018b)'s claim insofar as they apply to their models, this study shows that such strong specialization is not a general property of CNNs. The degree of specialization may vary between models (and tasks), and further study might investigate factors which relate to the degree of specialization.

Regarding the claim that selectivity measures are good predictors of the effects of lesioning, I find that although all three measures were associated with the effects of lesioning, there was low agreement between the most selective class and the class most impacted by lesioning (e.g., $< 32\%$). It could be the case that my model is different to those used in Zhou et al. (2018b), in terms of the low agreement. They did not address this question, so I do not know whether the most selective class was also the class with the largest drop in accuracy in their models. The low agreement in my model has strong implications for using selectivity to predict unit function. A unit that is most selective for $x$ might actually be *more* important for the representation of $y$. This further highlights the risks of claiming that a unit is a 'detector' for $x$ based on the high selectivity. Further, the units in this model that are most specialised typically had weak selectivity. I therefore suggest that caution is advised when inferring the functional role of a unit based on its selectivity alone. My findings are generally consistent with the claim from Morcos et al. (2018) that efforts to understand neural networks by focusing on the most selective units may be misleading.

Along with the finding that selectivity is only weakly associated with importance for a given class, this chapter also finds that a unit's level of activation for a given item does not perfectly predict the importance of the unit for classification of that item. Items that fail when a unit is lesioned are often not the most active items. In figures 6.6 and 6.14, there are items that fail across all levels of activation (e.g., low, medium or high) whilst other items with similar activation do not fail. This finding is not only relevant for neural network research, but also for psychology and neuroscience; where high activation of a neuron (or group of neurons) is equated with importance for a particular stimulus. Whilst this assumption is reasonable, the results here highlight a more complex relationship between level of activation and importance. In distributed representations the importance of a unit for a given item may depend on a number of other units that are also active for that item. As discussed in chapter 4, there may be classes (or items) for which no units have high activation (e.g., class 2 in figure 4.8). Yet the model succeeds on these classes, and therefore, units with intermediate or low activation for these classes must be somewhat important for their classification. In other words, high activation for an item does not guarantee that the unit makes a significant contribution to the recognition of that item. Similarly, some items with intermediate or low activation fail when a unit is lesioned, demonstrating that units can play an important role for items with low activation.

This study found that in my model, most units were not specialised. However, the lesioning measures did allow us to identify the few units that were most specialised. I find that although selectivity has some predictive power regarding class-importance, the correspondence between selectivity and the effects of lesioning is low. As such, this study highlights the fact that high selectivity is not necessarily indicative of high importance for a particular class.

# Chapter 7

# Conclusions

This thesis has examined selectivity in neural networks. The aims were to survey ways that selectivity is measured; to identify aspects of the dataset, model, task and training affect levels of selectivity; and to explore the relationship between selectivity and unit function.

## 7.1 Measuring Selectivity

In chapter 3 I compared four selectivity measures on units from three model types, with four model sizes, trained on three datasets. This resulted in a diverse set of representations: normally distributed, exponentially distributed, binary-like, dense, and sparse. This gave opportunity to demonstrate situations where the measures did not perform as desired.

I found that none of the measures were adequately able to capture the whole range of possible selectivity scores, with all measures showing either floor or ceiling effects. A B-sel of zero does not mean that the activation for all classes is equal (e.g., floor effect). The other three measures showed ceiling effects. Precision and max-info score any linearly-separable class at 1.0, although this is far from the highest possible selectivity. CCMA would give a score of 1.0 whenever there was only a single active class, regardless of the proportion of items from that class that were active. This limitation with CCMA has previously been discussed with reference to hypothetical data (Gale et al., 2019, 2020), although here I provide an example of a maximally sparse unit from real data. The floor and ceiling effects have consequences for understanding the levels of selectivity in the literature. A unit reported as having the maximum or minimum possible selectivity score might be far from maximally (or minimally) selective. In this chapter I propose the use of maximum-informedness as a measure of selectivity in neural networks. I found that although it shows ceiling effects, it is a more reliable indicator of class-selectivity then B-sel, CCMA and precision. The ceiling effects associated with maximum-informedness could be avoided if selectivity was reported as the sum of maximum-informedness and B-sel, which would give a measure in the range 0-2. Although the different measures provided divergent estimates of the selectivity of some units, there were units that were judged to be highly selective by *all* measures. This provides further evidence of highly interpretable units, that is, units where the activation is interpretable with respect to the input.

## 7.2 Factors that Impact on Selectivity

In chapter 3, I also looked at the effect of dataset, model type, model size on selectivity. The four measures were rarely unanimous regarding the effects of these different factors. However, the four measures did agree on an effect of activation function, with significantly higher selectivity for single layer models using relus than sigmoids. The measures also all had higher selectivity in the final layer of Relu4 models than in the first layer. The four measures did not agree on the effect of model size or model depth.

Across four experiments in chapter 4 I found that increasing the arbitrariness of input out mappings led to reduced selectivity for that dataset or class. The effect of arbitrariness on class-selectivity had not previously been demonstrated in the literature. Increasing the systematicity led to higher selectivity. This finding is surprising in the context of predictions that selectivity will be higher for arbitrary one-to-one tasks (e.g., Hinton et al., 1986; McClelland et al., 1995; Kumaran et al., 2016), and previous research reporting no effect of arbitrariness (Vankov and Bowers, 2017). The relationship between arbitrariness and selectivity was well captured with the similarity difference measure (mean within-class cosine similarity minus mean between-class cosine similarity) for pseudo-randomly generated data; and it predicted the levels of selectivity for classes in the unmodified MNIST dataset. That is, differences in within and between class similarity can explain why selectivity is higher for classes 0 and 1 in MNIST than for other classes. However, the similarity difference was less predictive of selectivity for the experiments which manipulated the MNIST dataset. This could be because the similarity differences between classes was already low.

In chapter 4, systematic datasets were easier to learn than more arbitrary datasets (e.g., inferred through the training time or accuracy). This suggests a more general role of task difficulty on selectivity. For example, in chapter 3, mean accuracy suggest that iris was the easiest dataset and fashion was the most difficult. Three of the four measures had highest selectivity for iris and lowest for fashion. Similarly, in chapter 5 I consistently found more highly selective units for the small vocabulary than the large vocabulary. However, this effect is not apparent everywhere: LSTMs in experiment 3 of chapter 5 clearly found the task much easier than the simple RNNs in experiments 1 and 2 (e.g., LSTMS had higher accuracy), yet there were fewer highly selective units in experiment 3 than in experiments 1 and 2.

In experiment 2 of chapter 4 I demonstrated that not every class needs to have a unit in the model where that class is most selective. There were no units that were most selective for class 2, yet class 2 had 100% accuracy. This highlights the fact that being most selective for a class does not imply that the unit exclusively represents information for that class. A unit can represent items from many classes, yet it is most selective for just one class. A unit can be important for many classes, yet it is most important for just one. It is useful to have short hand when describing complicated thing concisely. However, phrases like 'unit-$y$ is selective for class-$x$' should *not* be taken as suggesting that unit-$y$ is *only* important for class-$x$.

In chapter 5 I investigated the claim that neural networks learn localist units in order to avoid the superposition-catastrophe (Bowers et al., 2014). Across three experiment using trained recurrent models I found no 'localist units' (e.g., with B-sel > .5) and I found no effect of list length (e.g., the main variable relating to the superposition-catastrophe) on the number of highly selective units (e.g., with B-sel > .1 or 0, for words and letters respectively). That is, there was no significant difference between the number of highly selective units in models trained on single items (e.g., with no superposition-constraints) and models trained on long lists (e.g., with superposition constraints). I note the impact of weight-initialization on levels of selectivity in my models; but this does not explain the differences between my models and Bowers'. This study does not rule out an effect of superposition-constraints on selectivity; nor does it falsify Bowers' claims relating to their model. It does suggests that the effects in Bowers et al. (2014) do not apply to recurrent models in general and may have been specific to their model or training regime. This study demonstrates that distributed representations can support model performance, even in the face of the superposition-catastrophe.

The results if the studies presented here highlight that the representations learned by NN are highly sensitive to changes in dataset, model and training parameters. I find effects of model depth and size; activation function and weight initialization on selectivity. Similarly, I report significant differences in selectivity between models trained on similar datasets (e.g., MNIST vs MNIST with a predictive cue added to two classes in chapter 4). As such, the results here suggest that the representations learned by NN are highly variable, small changes to these parameters can lead to significant differences in learned representations making comparison between the internal states of a model and human potentially challenging.

## 7.3 Selectivity and Unit Function

Chapter 6 explored ways of measuring the class-specific effects of lesioning, and the relationship between the effects of lesioning and selectivity. I used the class-accuracy-change method from Zhou et al. (2018b), with which they had found that units showed a degree of specialisation. In my model, the max-class-drop from class-accuracy-change was confounded with total-accuracy-change. That is, the units with the largest drop in accuracy for one class, also had a large drop in accuracy for other classes. This makes this measure unsuitable for identifying specialised units. I proposed a variation called drop-prop; but this also proved unsuitable for identifying specialised units. However, in conjunction, these two measures were able to find a small subset of units which displayed some degree of specialisation (e., lesioning led to a significant drop in accuracy for one class only).

Although the measures were not suitable for identifying specialised units, both class-accuracy-drop and drop-prop highlighted relevant aspects of the effects of lesioning on a per-class basis. There were significant correlations between the max-class-drop (e.g., the effect of lesioning for the class with the largest effect) and selectivity for that class. This is consistent with Zhou et al. (2018b). However, importantly, I show that the most selective class was only the max-class-drop for between 17-31% of units (depending on the measure of selectivity and lesioning). Logistic regression models performed above chance at predicting the identity of the max-class-drop from the selectivity scores for drop-prop, but not class-accuracy-drop. However, even for drop-prop, accuracy was less than 50% meaning that predictions were incorrect more often than they were correct. In other words, although there is an association between the magnitude of the effects of lesioning and selectivity for that class; selectivity is a poor predictor of which class will be most affected by lesioning.

Chapter 6 has relevance regarding the interpretability of units. Selectivity allows a model to be interpreted with respect to some aspect of the input; e.g., is the unit tuned to one class. Lesioning allows a unit to be interpreted with respect to the output; e.g., how does this unit contribute to a particular response. In this sense, lesioning can show the functional role played by a unit. This chapter highlight the fact that the relationship between these two factors is weak. The most selective units were not the most specialised; the most specialised units had low selectivity, and were typically most selective for classes other than the one they were specialised for. I do not deny an association between selectivity and unit function, neither do I deny that this association can be stronger than in my model (e.g., as reported by Zhou et al. (2018b). The main point is that specialised neurons are not a general property of CNNs. CNNs may vary in their degree of specialisation, and further study will be required to identify factors that relate to the degree of specialisation.

The studies here show that the relationship between a unit's level of activation for an item and the unit's importance for the item is far from perfect. In chapter 6 we find that the items that fail when a unit is lesioning are not necessarily items with high activation, which many items with high activation do not fail. Similarly, in chapter 4 there are classes for which no unit has high activation, yet the model has perfect accuracy. These findings are important for psychology and neuroscience, where high activation is typically equated with importance.

## 7.4 Relevance/Importance

Chapter 3 highlights the need for a clearer idea of what constitutes high selectivity, especially regarding sensitivity. That is, should a unit have a high selectivity score for a class where only a subset of items are active? Where research shows units with high selectivity scores (e.g., approaching 1.0, Morcos et al., 2018; Zhou et al., 2015) it is important to know what this means. This chapter also introduces maximum-informedness as a measure of selectivity in neural networks.

Three of the chapters in this thesis were based on attempts to replicate previous research. In each case, I did not find the same effects as in the original studies. This highlights that diversity of neural networks, what is true of one model might not be true of others.

In the case of chapter 4, I found a significant effect of arbitrariness on selectivity, where Vankov and Bowers (2017) found none. This can likely be attributed to their use of B-sel, which is not sensitive to changes in selectivity for units with low selectivity (e.g., not linearly-separable). The finding of an effect of systematicity on selectivity has implications for understanding why selectivity is higher for one dataset than another; and perhaps more importantly, why one class has higher selectivity than other classes. This finding also suggests that selectivity may be higher for easier tasks than more difficult tasks. Task difficult may include systematicity as a component, however, there may be other factors such as the number of items or classes. Future research could explore this direction.

For chapters 5 and 6 the same selectivity measures were used as in the original studies; so differences in my findings must come from elsewhere. For chapter 5 I used the same size model and data and were used in Bowers et al. (2014), but there details of their implementation that I was unable to successful apply in my model (e.g., they used a custom optimizer and treated hidden states differently at the start of the list). These factors may explain why I did not replicate their effects. Nonetheless, I show that pressure to avoid the superposition-catastrophe on selectivity is not as strong as suggested in Bowers et al. (2014).

In chapter 6 my model and task were smaller those used by Zhou et al. (2018b), so the difference in effects may be a product of the differences in model or dataset size. My findings demonstrate that the degree of specialisation may vary between models, and that claims regarding unit function based solely on selectivity values alone should be treated with caution.

## 7.5 Limitations and Reflections

For chapter 6 I had hoped to be able to train and analyse alexNet or a similar sized model (e.g., VGG16, resNet20) on imageNet. This would have allowed a much closer replication of Zhou et al. (2018b). My limited ability as a programmer meant that this was unsuccessful. However, in using a smaller model, this chapter highlights that findings from these large benchmark datasets and models may not generalize to smaller models.

Similarly, I had intended to directly replicate Bowers et al. (2014) in chapter 5, and then investigate the effects of dataset on the superposition-catastrophe. Their models appeared to be fairly straight-forward simple-recurrent models, however, details of their exact methodology revealed several arcane parameters (e.g., their optimizer, weight initializer and hidden states). Ultimately I failed to understand the impact of these parameters on training or selectivity, and as such, I can not explain the difference in the results between their study and my replications. The failure to capture these effects is important, in that it shows that they are not robust and generalizable. But it also meant I was not able to proceed in the hoped direction.

This lack of generalizability of findings from selectivity research in neural networks naturally casts doubts on my findings here. I have generally used a variety of models (e.g., MLPs, CNNs, RNNs), activation functions and sizes, which should make the findings more robust than had I just focused on a single model. However, it is not clear how well these findings will generalize to the large-scale models often used in published neural-network research.

Similarity difference was a good predictor of the effects of systematicity on selectivity for small semi-randomly generated datasets, but less so for the larger MNIST dataset. Better measures of the similarity or systematicity would be useful for exploring the role of systematicity in more detail.

Based on these conclusions, future studies could address the need for a selectivity measure which can capture the full range of possible selectivity scores; i.e., where the the maximum value is equivalent to a B-sel of 1.0. I have shown that such high selectivity can occur (e.g., a unit with OFF B-sel of 1.0 in chapter 3), yet the other three measures I included reach ceiling for units with demonstrably lower class-selectivity. A measure which covers the full range of possible selectivity would make comparison between units in different studies easier. This is not trivial, unit activations are diverse. The suitability of a measure is sometimes decided based on the distribution from which the data is drawn. However, activations may come from very different distributions (e.g., normal, exponential); the proportion of zeros may differ greatly, the number of classes can vary and classes classes can be of different sizes. Failing this, future studies might explicitly discuss how they will deal with sparse units, OFF units and unequal class sizes.

## 7.6 Summary

In this thesis I have investigated how selectivity is measures and have shown that commonly used methods can not capture the full range of possible selectivity values. This limits the reliability of the data and conclusions drawn from this data.

I have investigated factors that impact on selectivity and I have demonstrated a causal role for the systematicity of a class or dataset on class-selectivity. This is significant as it is in the opposite direction as claims relating to selectivity in one-to-one tasks. I also consistently found higher selectivity scores for tasks with shorter training times or higher accuracy, than for tasks with lower accuracy or requiring more training. This suggests that high selectivity may relate to task difficulty. I failed to find any evidence to support the claim that pressure to avoid the superposition-catastrophe leads recurrent models to learn localist representations.

Finally, I have investigated the relationship between selectivity and unit function and I have demonstrated in my model the class with the highest selectivity is rarely the class with the biggest impact of lesioning. In other words, class-selectivity is a poor predictor of class-importance. This suggests that caution is advised when using selectivity values to predict the functional role of a unit.

# BIBLIOGRAPHY

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I. J., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2016).
TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.

Adadi, A. and Berrada, M. (2018).
Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI).
*IEEE Access*, 6:52138–52160.

Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., Belopolsky, A., Bengio, Y., Bergeron, A., Bergstra, J., Bisson, V., Snyder, J. B., Bouchard, N., Boulanger-Lewandowski, N., Bouthillier, X., de Brébisson, A., Breuleux, O., Carrier, P.-L., Cho, K., Chorowski, J., Christiano, P., Cooijmans, T., Côté, M. M.-A., Courville, A., Dauphin, Y. N., Delalleau, O., Demouth, J., Desjardins, G., Dieleman, S., Dinh, L., Ducoffe, M., Dumoulin, V., Kahou, S. E., Erhan, D., Fan, Z., Firat, O., Germain, M., Glorot, X., Goodfellow, I. J., Graham, M., Gulcehre, C., Hamel, P., Harlouchet, I., Heng, J.-P., Hidasi, B., Honari, S., Jain, A., Jean, S., Jia, K., Korobov, M., Kulkarni, V., Lamb, A., Lamblin, P., Larsen, E., Laurent, C., Lee, S., Lefrancois, S., Lemieux, S., Léonard, N., Lin, Z., Livezey, J. A., Lorenz, C., Lowin, J., Ma, Q., Manzagol, P.-A., Mastropietro, O., McGibbon, R. T., Memisevic, R., van Merriënboer, B., Michalski, V., Mirza, M., Orlandi, A., Pal, C., Pascanu, R., Pezeshki, M., Raffel, C., Renshaw, D., Rocklin, M., Romero, A., Roth, M., Sadowski, P., Salvatier, J., Savard, F., Schlüter, J., Schulman, J., Schwartz, G., Serban, I. V., Serdyuk, D., Shabanian, S., Simon, É., Spieckermann, S., Subramanyam, S. R., Sygnowski, J., Tanguay, J., van Tulder, G., Turian, J., Urban, S., Vincent, P., Visin, F., de Vries, H., Warde-Farley, D., Webb, D. J., Willson, M., Xu, K., Xue, L., Yao, L., Zhang, S., and Zhang, Y. (2016).
Theano: A Python framework for fast computation of mathematical expressions.
*arXiv preprint arXiv:1605.02688*.

Barlow, H. B. (1953).

Summation and inhibition in the frog's retina.
*Journal of Physiology*, 119(1):69.

Barlow, H. B. (1972).
Single units and sensation: A neuron doctrine for perceptual psychology?
*Perception*, 1(4):371–394.

Bau, D., Zhu, J.-Y., Strobelt, H., Zhou, B., Tenenbaum, J. B., Freeman, W. T., and Torralba, A. (2018).
GAN Dissection: Visualizing and Understanding Generative Adversarial Networks.
*arXiv:1811.10597v2*.

Berkeley, I. S., Dawson, M. R., Medler, D. A., Schopflocher, D. P., and Hornsby, L. (1995).
Density plots of hidden value unit activations reveal interpretable bands.
*Connection Science*, 7(2):167–187.

Berkeley, I. S. and Gunay, C. (2004).
Conducting banding analysis with trained networks of sigmoid units.
*Connection Science*, 16(2):119–128.

Botvinick, M. and Plaut, D. C. (2006).
Short-term memory for serial order: a recurrent neural network model.
*Psychological review*, 113(2):201.

Bowers, J. S. (2009).
On the biological plausibility of grandmother cells: implications for neural network theories in psychology and neuroscience.
*Psychological review*, 116(1):220–251.

Bowers, J. S. (2010).
More on grandmother cells and the biological implausibility of PDP models of cognition: A reply to Plaut and McClelland (2010) and Quian Quiroga and Kreiman (2010).
*Psychological Review*, 117(1):300–306.

Bowers, J. S. (2011).
What is a grandmother cell? And how would you know if you found one?
*Connection Science*, 23(2):91–95.

Bowers, J. S., Vankov, I. I., Damian, M. F., and Davis, C. J. (2014).
Neural networks learn highly selective representations in order to overcome the superposition catastrophe.
*Psychological Review*, 121(2):248–261.

Bowers, J. S., Vankov, I. I., Damian, M. F., and Davis, C. J. (2016).
Why do some neurons in cortex respond to information in a selective manner? Insights from artificial neural networks.
*Cognition*, 148:47–63.

Chollet, F. and Others (2015).
Keras.

Coltheart, M. (2017).
Grandmother cells and the distinction between local and distributed representation.
*Language, Cognition and Neuroscience*, 32(3):350–358.

Coltheart, M., Rastle, K., Perry, C., Langdon, R., and Ziegler, J. (2001).
DRC: A Dual Route Cascade Model of Visual Word Recognition and Reading Aloud.
*Psychological Research*, 108(1):204–256.

Dalvi, F., Durrani, N., Sajjad, H., Belinkov, Y., Bau, A., and Glass, J. (2019).
What Is One Grain of Sand in the Desert? Analyzing Individual Neurons in Deep NLP Models.
*Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6309–6317.

Davis, C. J. (2001).
*The self-organising lexical acquisition and recognition (SOLAR) model of visual word recognition.*
PhD thesis, Royal Holloway, University of London.

Davis, C. J. (2010).
The spatial coding model of visual word identification.
*Psychological Review*.

Dell, G. S. (1986).
A Spreading-Activation Theory of Retrieval in Sentence Production.
*Psychological Review*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009).
Imagenet: A large-scale hierarchical image database.
In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

DiCarlo, J. J., Zoccolan, D., and Rust, N. C. (2012).
How does the brain solve visual object recognition?
*Neuron*, 73(3):415–434.

Donnelly, J. and Roegiest, A. (2019).

On interpretability and feature representations: An analysis of the sentiment neuron.
*Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11437 LNCS:795–802.

Dosilovic, F. K., Brcic, M., and Hlupic, N. (2018).
Explainable artificial intelligence: A survey.
*2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings*, pages 210–215.

Doumas, L. A. and Martin, A. E. (2018).
*Learning structured representations from experience*, volume 69.
Elsevier Inc., 1 edition.

Dujmović, M., Malhotra, G., and Bowers, J. S. (2020).
What do adversarial images tell us about human vision?
*eLife*, 9:1–29.

Dunn, O. J. (1964).
Multiple Comparisons Using Rank Sums.
*Technometrics*.

Eichenbaum, H. (2017).
Barlow versus Hebb: When is it time to abandon the notion of feature detectors and adopt the cell assembly as the unit of cognition?
*Neuroscience Letters*.

Elman, J. L. (1990).
Finding Structure in Time.
*Cognitive Science1*, 14(2):179—-211.

Farah, M. J. and Wallace, M. A. (1992).
Semantically-bounded anomia: Implications for the neural implementation of naming.
*Neuropsychologia*, 30(7):609–621.

Fisher, R. A. and Marshall, M. (1936).
Iris data set: accessed 9th August 2017.

Földiák, P. (2009).
Neural Coding: Non-Local but Explicit and Conceptual.
*Current Biology*, 19(19):904–906.

Fong, R. C. and Vedaldi, A. (2017).
Interpretable Explanations of Black Boxes by Meaningful Perturbation.
*Proceedings of the IEEE International Conference on Computer Vision*, 2017-Octob:3449–3457.

Freedman, D. J. and Assad, J. A. (2006).
Experience-dependent representation of visual categories in parietal cortex.
*Nature*, 443(7107):85–88.

Gale, E. M., Blything, R., Martin, N., Bowers, J. S., and Nguyen, A. (2019).
Selectivity metrics provide misleading estimates of the selectivity of single units in neural networks.
In *Cognitive Science*, pages 1808–1814, Montreal.

Gale, E. M., Martin, N., Blything, R., Nguyen, A., and Bowers, J. S. (2020).
Are there any 'object detectors' in the hidden layers of CNNs trained to identify objects or scenes?
*Vision Research*, 176(January):60–71.

Gale, E. M., Martin, N., and Bowers, J. S. (2018).
When and where do feed-forward neural networks learn localist representations?
*CoRR*, abs/1806.0:1–17.

Geirhos, R., Michaelis, C., Wichmann, F. A., Rubisch, P., Bethge, M., and Brendel, W. (2019).
Imagenet-Trained Cnns Are Biased Towards Texture; Increasing Shape Bias Improves Accuracy and Robustness.
In *ICLR*, pages 1–22, New Orleans.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. (2019).
Explaining explanations: An overview of interpretability of machine learning.
*Proceedings - 2018 IEEE 5th International Conference on Data Science and Advanced Analytics, DSAA 2018*, pages 80–89.

Glorot, X. and Bengio, Y. (2010).
Understanding the difficulty of training deep feedforward neural networks.
In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249—-256.

Glorot, X., Bordes, A., and Bengio, Y. (2011).
Deep sparse rectifier neural networks.
In *Journal of Machine Learning Research*.

Goldberg, Y. and Levy, O. (2014).
word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method.
*arXiv preprint arXiv:1402.3722*.

Goodfellow, I. J., Bengio, Y., and Courville, A. (2016).
*Deep Learning*.

MIT press.

Grainger, J. and Jacobs, A. M. (1996).
Orthographic Processing in Visual Word Recognition: A Multiple Read-Out Model.
*Psychological Review*.

Graves, A., Fernandez, S., Gomez, F., and Schmidhuber, J. (2006).
Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks.
*Proceedings of the 23rd international conference on Machine Learning*, pages 369–376.

Graves, A., Mohamed, A. R., and Hinton, G. E. (2013).
Speech recognition with deep recurrent neural networks.
In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*.

Guest, O. and Love, B. C. (2019).
Levels of Representation in a Deep Learning Model of Categorization.
*bioRxiv*, page 626374.

He, K., Zhang, X., Ren, S., and Sun, J. (2015).
Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.
In *Proceedings of the IEEE international conference on computer vision*, pages 1026—-1034.

He, K., Zhang, X., Ren, S., and Sun, J. (2016).
Deep residual learning for image recognition.
*Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:770–778.

Hebb, D. (1949).
*The Organisation of Behaviour: A neuropsychological approach*.
J. Wiley; Chapman & Hall.

Higgins, I., Chang, L., Langston, V., Hassabis, D., Summerfield, C., Tsao, D. Y., and Botvinick, M. (2020).
Unsupervised deep learning identifies semantic disentanglement in single inferotemporal neurons.
*arXiv preprint arXiv:2006.14304*, pages 1–24.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2017).
B-VAE: Learning Basic Visual Concepts with a Constrined Variational Framework.
In *International Conference on Learning Representations*, pages 1–22.

Hinton, G. E. (1986).
Learning distributed representations of concepts.
In *Proceedings of the eighth annual conference of the Cognitive Science Society*.

Hinton, G. E., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., and Kingsbury, B. (2012).
Deep Neural Networks for Acoustic Modeling in Speech Recognition.
*Ieee Signal Processing Magazine*.

Hinton, G. E., McClelland, J. L., and Rumelhart, D. E. (1986).
Distributed representations.
In *Parallel Distributed Processing*, chapter 3, pages 77–109. MIT Press.

Hinton, G. E., Vinyals, O., and Dean, J. (2015).
Distilling the Knowledge in a Neural Network.
*arXiv preprint arXiv:1503.02531*, pages 1–9.

Hochreiter, S. and Schmidhuber, J. (1997).
Long Short-Term Memory.
*Neural Computation*.

Hubel, D. H. and Wiesel, T. N. (1959).
Receptive fields of single neurones in the cat's striate cortex.
*The Journal of physiology*, 148(3):574—-591.

Hubel, D. H. and Wiesel, T. N. (1962).
Receptive fields, binocular interaction and functional architecture in the cat's visual cortex.
*Journal of Physiology*, 160(1):106—-154.

Hummel, J. E. and Holyoak, K. J. (1997).
Distributed representations of structure: A theory of analogical access and mapping.
*Psychological review*, 104(3):427—-466.

Itti, L., Koch, C., and Niebur, E. (1998).
A model of saliency-based visual attention for rapid scene analysis.
*IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Jeffrey Johnston, W., Palmer, S. E., and Freedman, D. J. (2020).
Nonlinear mixed selectivity supports reliable neural computation.
*PLoS Computational Biology*, 16(2):1–37.

Karpathy, A. (2016).
CS231n Convolutional Neural Networks for Visual Recognition.

Karpathy, A., Johnson, J., and Fei-Fei, L. (2016).
Visualizing and Understanding Recurrent Networks.
In *Iclr*, pages 1–13.

Kementchedjhieva, Y. and Lopez, A. (2018).
'Indicatements' that character language models learn English morpho-syntactic units and regularities.
In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 145–153.

Kingma, D. P. and Ba, J. (2014).
Adam: A method for stochastic optimization.
*arXiv preprint arXiv:1412.6980*.

Konorski, J. (1967).
*Integrative activity of the brain: an interdisciplinary approach.*
University of Chicago Press.

Kornblith, S. and Tsao, D. Y. (2017).
How thoughts arise from sights: inferotemporal and prefrontal contributions to vision.
*Current Opinion in Neurobiology*, 46:208–218.

Krizhevsky, A., Hinton, G. E., and Others (2009).
Learning multiple layers of features from tiny images.
Technical report, Citeseer.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012).
Imagenet classification with deep convolutional neural networks.
In *Advances in neural information processing systems*, pages 1097–1105.

Kruskal, W. H. and Wallis, W. A. (1952).
Use of Ranks in One-Criterion Variance Analysis.
*Journal of the American Statistical Association*.

Kumaran, D., Hassabis, D., and McClelland, J. L. (2016).
What learning systems do intelligent agents need? complementary learning systems theory updated.
*Trends in Cognitive Sciences*, 20(7):512–534.

Lakretz, Y., Kruszewski, G., Desbordes, T., Hupkes, D., Dehaene, S., and Baroni, M. (2019).
The emergence of number and syntax units in LSTM language models.
*arXiv preprint arXiv:1903.07435*.

Landau, B., Smith, L. B., and Jones, S. S. (1988).
The importance of shape in early lexical learning.
*Cognitive Development*.

Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. (2013).
Building high-level features using large scale unsupervised learning.
In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 8595–8598. IEEE.

Leavitt, M. L. and Morcos, A. S. (2020a).
On the relationship between class selectivity, dimensionality, and robustness.
*arXiv preprint arXiv:2007.04440*.

Leavitt, M. L. and Morcos, A. S. (2020b).
Selectivity considered harmful: evaluating the causal impact of class selectivity in DNNs.
*Arxiv*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998).
Gradient-based learning applied to document recognition.
*Proceedings of the IEEE*, 86(11):2278–2323.

Lettvin, J. Y., Maturana, H. R., McCulloch, W. S., and Pitts, W. H. (1959).
What the frog's eye tells the frog's brain.
*Proceedings of the IRE*, 47(11):1940—-1951.

Li, Y., Yosinski, J., Clune, J., Lipson, H., and Hopcroft, J. (2016).
Convergent Learning: Do different neural networks learn the same representations?
In *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, pages 196–212, San Juan, Puerto Rico.

Liu, R., Mu, P., and Zhang, J. (2019).
On the convergence of ADMM with task adaption and beyond.
*arXiv*.

Mahendran, A. and Vedaldi, A. (2015).
Understanding deep image representations by inverting them.
*Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June:5188–5196.

Mahendran, A. and Vedaldi, A. (2016).
Visualizing Deep Convolutional Neural Networks Using Natural Pre-images.
*International Journal of Computer Vision*, 120(3):233–255.

Malhotra, G., Evans, B., and Bowers, J. S. (2019).
Adding biological constraints to CNNs makes image classification more human-like and robust.
In *2019 Conference on Cognitive Computational Neuroscience*, pages 256–259, Berlin.

Marr, D., Willshaw, D., and McNaughton, B. (1991).
Simple Memory : A Theory for Archicortex.
In L., V., editor, *From the Retina to the Neocortex*, pages 59—-128. Birkhäuser Boston.

Martin, A. E. and Doumas, L. A. (2019).
Predicate learning in neural systems: using oscillations to discover latent structure.
*Current Opinion in Behavioral Sciences*, 29:77–83.

Martin, K. A. (1994).
A brief history of the "feature detector".
*Cerebral Cortex*, 4(1):1–7.

McClelland, J. L. (1994).
The interaction of nature and nurture in development: A parallel distributed processing perspective.
*International Perspectives on Psychological Science, Volume I: Leading themes*, 1:57–88.

McClelland, J. L., McNaughton, B. L., and O'Reilly, R. C. (1995).
Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory.
*Psychological Review*, 102(3):419–457.

McClelland, J. L. and Rumelhart, D. E. (1981).
An interactive activation model of context effects in letter perception: I. An account of basic findings.
*Psychological review*, 88(5):375.

Millman, K. J. and Aivazis, M. (2011).
Python for scientists and engineers.

Morcos, A. S., Barrett, D. G., Rabinowitz, N. C., and Botvinick, M. (2018).
On the importance of single directions for generalization.
In *International Conference on Learning Representations*, pages 1–15.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. (2019).
Definitions, methods, and applications in interpretable machine learning.
*Proceedings of the National Academy of Sciences of the United States of America*, 116(44):22071–22080.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011).
Reading Digits in Natural Images with Unsupervised Feature Learning.
In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Nguyen, A., Yosinski, J., and Clune, J. (2015).
Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images.
*Computer Vision and Pattern Recognition, 2015 IEEE Conference on*, pages 427–436.

NVIDIA, Vingelmann, P., and Fitzek, F. H. (2020).
CUDA, release: 10.2.89.

Olah, C., Cammarate, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. (2020).
Zoom In: An Introduction to Circuits.
*Distill*.

Open AI (2020).
Microscope (OpenAI).

Page, M. (2000).
Connectionist modelling in psychology: a localist manifesto.
*Behavioral and Brain Sciences*, 23(4):476–477.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011).
Scikit-learn: Machine Learning in {P}ython}.
*Journal of Machine Learning Research*, 12:2825—-2830.

Plaut, D. C. and McClelland, J. L. (2000).
Stipulating versus discovering representations.
*Behavioral and Brain Sciences*, 23(4):489–491.

Plaut, D. C. and McClelland, J. L. (2010).
Locating object knowledge in the brain: Comment on Bowers's (2009) attempt to revive the grandmother cell hypothesis.
*Psychological Review*, 117(1):284–288.

Plaut, D. C., Mcclelland, J. L., Seidenberg, M. S., and Patterson, K. (1996).
Understanding normal and impaired word reading computational principles in quasi-regular domains.
*Psychological Review*, 103(1):56–115.

Powers, D. M. W. (2003).
Recall & Precision versus The Bookmaker.
In *International Conference on Cognitive Science*, pages 529–534.

Powers, D. M. W. (2011).
Evaluation: from Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation.
*Journal of Machine Learning Technologies*, 2(1):37–63.

Qian, P., Qiu, X., and Huang, X. (2016).
Analyzing Linguistic Knowledge in Sequential Model of Sentence.
In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 826–835.

Radford, A., Jozefowicz, R., and Sutskever, I. (2017).
Learning to Generate Reviews and Discovering Sentiment.
*arXiv preprint arXiv:1704.01444*.

Rafegas, I., Vanrell, M., and Alexandre, L. A. (2017).
Understanding trained CNNs by indexing neuron selectivity.
In *International Conference on Learning Representations*, pages 1–13.

Rafegas, I., Vanrell, M., Alexandre, L. A., and Arias, G. (2020).
Understanding trained CNNs by indexing neuron selectivity.
*Pattern Recognition Letters*, 136:318–325.

Ricci, M., Kim, J., and Serre, T. (2018).
Not-So-{CLEVR}: Visual Relations Strain Feedforward Neural Networks.
*arXiv preprint arXiv:1802.03390*, pages 1–11.

Rogers, T. T. and McClelland, J. L. (2008).
Précis of Semantic Cognition: A Parallel Distributed Processing Approach.
*Behavioral and Brain Sciences*, 31(06):689.

Rogers, T. T. and Mcclelland, J. L. (2014).
Parallel distributed processing at 25: Further explorations in the microstructure of cognition.
*Cognitive science*, 38(6):1024–1077.

Rohde, D. L. (1999).
LENS: The light, efficient network simulator.

Rosenblatt, F. (1961).
Principles of neurodynamics. perceptrons and the theory of brain mechanisms.
Technical report, Cornell Aeronautical Lab Inc Buffalo NY.

Rumelhart, D. E. and McClelland, J. L. (1982).
An interactive activation model of context effects in letter perception: II. The contextual enhancement effect and some tests and extensions of the model.
*Psychological review*, 89(1):60.

Rumelhart, D. E., McClelland, J. L., Group, P. R., and Others (1986a).
*Parallel distributed processing: Explorations in the microstructures of cognition. Volume 1: Foundations*.
The MIT Press, Cambridge, MA.

Rumelhart, D. E., McClelland, J. L., PDP Research Group, and Others (1986b).
*Parallel Distributed Processing, Volume 2. Psychological and Biological Models*.
MIT Press.

Rust, N. C. and Dicarlo, J. J. (2012).
Balanced increases in selectivity and tolerance produce constant sparseness along the ventral visual stream.
*Journal of Neuroscience*, 32(30):10170–10182.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Muller, K. R. (2016).
Evaluating the Visualization of What a Deep Neural Network Has Learned.
*IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13.

Schmidhuber, J. (2015).
Deep Learning in neural networks: An overview.
*Neural Networks*, 61:85–117.

Sherrington, C. (1940).
*Man on his Nature*.
Cambridge University Press.

Simonyan, K. and Zisserman, A. (2015).
Very deep convolutional networks for large-scale image recognition.
*3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–14.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., , Vanhoucke, V., and Rabinovich, A. (2015).
Going deeper with Convolutions.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1—-9.

Szegedy, C., Zaremba, W., and Sutskever, I. (2013).
Intriguing properties of neural networks.

*arXiv preprint arXiv:...*, pages 1–10.

Thomas, E. and French, R. (2017).
Grandmother cells: much ado about nothing.
*Language, Cognition and Neuroscience*, 32(3):342–349.

Thorpe, S. J. (1995).
Localized versus distributed representations.
In Arbib, M. A., editor, *The handbook of brain theory and neural networks*, pages 549—-552. MIT Press.

Tsao, D. Y., Freiwald, W. A., Tootell, R. B., and Livingstone, M. S. (2006).
A cortical region consisting entirely of face-selective cells.
*Science*, 311(5761):670—-674.

Tsvetkov, C., Malhotra, G., Evans, B. D., and Bowers, J. S. (2019).
Adding biological constraints to deep neural networks reduces their capacity to learn unstructured data.
In *Poster presented at the 42nd Annual Conference of the Cognitive Science Society (p. 2358)*, pages 2358—-2364.

Van Rijsbergen, C. J. (1979).
Information Retrieval, 2nd edition.
*Butterworths*.

Vankov, I. I. and Bowers, J. S. (2017).
Do arbitrary input–output mappings in parallel distributed processing networks require localist coding?
*Language, Cognition and Neuroscience*, 32(3):392–399.

Von Der Malsburg, C. (1986).
Am I thinking assemblies?
In *Brain theory*, pages 161—-176. Springer.

Xiao, H., Rasul, K., and Vollgraf, R. (2017).
Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms.
*arXiv preprint*, arXiv:1708:1–6.

Yamins, D. L. K., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D., and DiCarlo, J. J. (2014).
Performance-optimized hierarchical models predict neural responses in higher visual cortex.
*Proceedings of the National Academy of Sciences of the United States of America*, 111(23):8619–8624.

Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., and Lipson, H. (2015).
Understanding Neural Networks Through Deep Visualization.
*arXiv preprint arXiv:1506.06579*, page 12.

Zeiler, M. D. and Fergus, R. (2014).
Visualizing and Understanding Convolutional Networks.
*Computer Vision–ECCV 2014*, 8689(PART 1):818–833.

Zeugmann, T., Poupart, P., Kennedy, J., Jin, X., Han, J., Saitta, L., Sebag, M., Peters, J., Bagnell, J. A., Daelemans, W., Webb, G. I., Ting, K. M., Ting, K. M., Webb, G. I., Shirabad, J. S., Fürnkranz, J., Hüllermeier, E., Matwin, S., Sakakibara, Y., Flener, P., Schmid, U., Procopiuc, C. M., Lachiche, N., and Fürnkranz, J. (2011).
Precision and Recall.
In *Encyclopedia of Machine Learning*, pages 781–781. Springer US, Boston, MA.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016).
Understanding deep learning requires rethinking generalization.
*arXiv preprint arXiv:1611.03530*.

Zhou, B., Bau, D., Oliva, A., and Torralba, A. (2017).
Interpreting Deep Visual Representations via Network Dissection.
*IEEE transactions on pattern analysis and machine intelligence*.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2015).
Object detectors emerge in deep scene CNNs.
In *ICLR*, pages 1–12.

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A. (2018a).
Places: A 10 Million Image Database for Scene Recognition.
*IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhou, B., Sun, Y., Bau, D., and Torralba, A. (2018b).
Revisiting the Importance of Individual Units in CNNs via Ablation.
*arXiv preprint arXiv:1806.02891*.

Zhuang, C., Yan, S., Nayebi, A., Schrimpf, M., Frank, M. C., DiCarlo, J. J., and Yamins, D. L. K. (2021).
Unsupervised neural network models of the ventral visual stream.
*Proceedings of the National Academy of Sciences of the United States of America*, 118(3).

Zylberberg, J. (2018).
The role of untuned neurons in sensory information coding.
*bioRxiv*, pages 1–23.