Exploring out of distribution: Deep neural networks and the human brain

by

Zubia Mansoor

B.Sc., St. Xavier's College, 2018

Project Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
Department of Statistics & Actuarial Science
Faculty of Science

Declaration of Committee

Name:	**Zubia Mansoor**
Degree:	**Master of Science**
Thesis title:	**Exploring out of distribution: Deep neural networks and the human brain**
Committee:	**Chair:** Haolun Shi Assistant Professor, Statistics and Actuarial Science
	Lloyd T. Elliott Supervisor Assistant Professor, Statistics and Actuarial Science
	Derek Bingham Committee Member Professor, Statistics and Actuarial Science
	Jiguo Cao Examiner Professor, Statistics and Actuarial Science

Abstract

Deep neural networks have achieved state-of-the-art performance across a wide range of tasks. Convolutional neural networks, with their ability to learn complex spatial features, have surpassed human-level accuracy on many image classification problems. However, these architectures are still often unable to make accurate predictions when the test data distribution differs from that of the training data. In contrast, humans naturally excel at such out-of-distribution generalizations. Novel solutions have been developed to improve a deep neural net's ability to handle out-of-distribution data. The advent of methods such as Push-Pull and AugMix have improved model robustness and generalization. We are interested in assessing whether or not such models achieve the most human-like generalization across a wide variety of image classification tasks. We identify AugMix as the most human-like deep neural network under our set of benchmarks. Identifying such models sheds light on human cognition and the analogy between neural nets and the human brain. We also show that, contrary to our intuition, transfer learning worsens the performance of Push-Pull.

Keywords: Out-of-distribution, deep learning, convolutional neural networks, cognitive science

Dedication

To Mom and Dad,
thank you for giving Raina, Shariah and me the greatest gift in life—an education.

Acknowledgements

I would like to dedicate my first acknowledgement to the most hands-on and supportive supervisor I have ever had—Dr. Lloyd T. Elliott. It is only because he took a chance on me that I am sitting here today, typing out my M.Sc. thesis to be submitted to SFU soon. There have been several instances where Lloyd has gone above and beyond his role as a supervisor for my well-being. I still remember I got caught up in one of the scam calls during my first week in Canada[1]. Lloyd sat down with me and wrote out all the steps I needed to take to ensure my security. Granted that Sonny and I were his first Master's students, but in reality, he was our first ever supervisor. We have had the special opportunity to try out Lloyd's exceptional food, hang out with his family and friends and participate in fun lab activities. All of this and more has made these two years nothing short of a memorable experience! I would also like to thank Derek Bingham, Jiguo Cao and Haolun Shi for their time reviewing this project and being on my committee.

I am still in mild awe that we all made it through the pandemic. At first, the thought that 80% of my graduate degree has been spent in a single studio up on an isolated mountain would be super disheartening. Nevertheless, being stuck with four other fellow students in the same place made it a lot better. This one is for my fellow Hamilton Hall friends—Louis, Sonny and Peter and Renny (for feeding me Indian food and letting me give tight hugs to his cat Nadia). I want to thank the Statistics department for providing every one of us with a homely environment. Even though our time together in person was limited, I loved going to the Meet-and-Greet, Burnaby Mountain picnic, Christmas party and Sadika's retirement party at Elia!

I feel so grateful sometimes to have met a fantastic group of friends in Canada and to have my best friend Uzma back home. Thank you for always taking my calls; I know the 12.5 hour time zone difference did not make that any easier. To Raaj for taking me to random events during my first year to meet new people. To Sumreen, Gabe and Sky for being the best hiking buddies. To Rahul, Suraj and Shashank for being the wise ones. To Riya and

[1]Almost had a mini-meltdown

Julia for being their absolute best selves. A special shoutout to those friends who went out of their way to help me with my professional endeavours. Dani, Kristen and Zaid—you are the best mentors and friends one could ever ask for.

I also want to have a paragraph dedicated to Abdullah. This project would not have come to fruition without your backing at every step of the way. I thoroughly enjoyed having late-night coding sessions, intellectually stimulating arguments and everything in between. Thanks for always being there when I had periodic meltdowns; I could not have asked for a better study buddy!

Lastly, I want to say that even words could not express the role played by my family in everything I have accomplished to date. Their constant support and love keep me going even on the darkest of days. The only bread-earner and an amazing dad of three girls, my dad is my inspiration. On the other hand, my mom has a strength unparalleled to anyone else's I have ever witnessed. To my baby sisters, the age gap is too much not to justify paying all your future bills. Being a Muslim minority woman came with its harsh realities. The good and the bad experiences combined gave me the confidence to leave the comfort of my own home and seek a new life here in Canada. My time here in this beautiful country has given me tremendous growth and a fresh perspective on life.

A special thanks to Compute Canada for providing us with the computational resources to conduct our research in a timely manner.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Advances in deep learning have created a notable impact on several areas of applications such as self-driving cars [11], object recognition [21], speech recognition [10], and complex strategy games like Go [30]. The advent of deep neural networks (DNNs) has ushered in a new era of state-of-the-art computer vision algorithms that can tackle image recognition challenges with machine observers (i.e. systems that automatically solve vision tasks [21]). These classes of algorithms are particularly adept at these tasks because they leverage concepts of hierarchical feature extraction and translational invariance to improve efficiency [9]. These networks have even surpassed human performance on multiple tasks [12, 30].

However, when we go beyond the usual assumption of independent and identically distributed (*i.i.d.*) test set samples, these models do not generalize well [35]. This paradigm beyond *i.i.d.* data involves training data that are not representative of the test data distribution. Such test sets are referred to as out-of-distribution data (OOD). Good performance on OOD data (generalization behaviour) is desirable because evaluation datasets in the real world rarely resemble the training universe [15]. One would therefore like to achieve state-of-the-art DNN performance in these realistic settings as well. For example, imagine a DNN that detects brain tumours based on patients' magnetic resonance imaging (MRI) scans. Once this model is deployed, suppose one of the scans it encounters is blurry (or has scanner-specific artifacts in the hospital where it is deployed). Even if the network was not exposed to blurred MRI scans during training, we would still like to have a model that can achieve benchmark performance.

Early works have attempted to improve OOD performance using data augmentation and processing techniques like rotations, scaling, cropping, etc. [21]. These approaches enable the network to learn deviations and scenarios during the training phase that may be expected to appear in the test set. A glaring drawback of conventional data augmentation, however, is that the model becomes robust exclusively to the distortions it was trained on [8]. A growing number of studies are now looking into alternative ways of improving generalization while maintaining state-of-the-art accuracy. One such technique was proposed by Hendrycks et al. called AugMix [17] that uses a random mixture of different augmentation techniques to expose the model to diverse images during training. It does so while maintaining the core visual content of the images.

On the other end of the spectrum, apart from data augmentation techniques, one may look at more intrinsic ways to improve generalization, that is, by designing network architectures that are more robust to OOD. A novel field of research has emerged, employing more brain-like DNNs to bridge this performance gap and inherit robust human visual features. The human brain can recognize objects rapidly and with precision [1]. Furthermore, human observers are unaffected by irregularities in images (such as distortions) and can identify objects presented from unusual angles or orientations [1]. As humans, we experience a variety of different visual scenarios like lighting settings (e.g., day, night), weather types (e.g., snow, rain), and environments (e.g., beach, hills). This prepares us to recognize objects across a wide range of conditions and can be equivalent to the training process for a DNN. This is of particular interest to us because of how humans perceive their surroundings. Our ability to accurately recognize objects is not hindered when these objects are placed in new, unfamiliar environments [1, 8]. For example, even if someone has never seen snow before, they are still able to identify objects in a snowy area. This property forms the crux of biologically-inspired DNNs, yielding a paradigm for better-equipped networks to handle OOD data.

Recent literature has looked at the similarities between human and DNN performance to come up with better models for object recognition [4, 8, 26, 29, 27]. For example, Strisciuglio et al. [32] propose a new type of neural network layer called the Push-Pull layer in their DNN architecture. This layer mimics the functions of the V1 (the primary visual) area [19] of the human brain [33]. Neurons in this Push-Pull inhibition layer are more sensitive to visual cues even when they are contaminated with noise [5]. Networks employing Push-Pull layers for vision tasks replace the first layer of their DNN with the Push-Pull layer and improve their robustness in the presence of OOD data [32].

Following the discussion above, the main goal in this work is to compare the robustness of deep neural networks with humans for image classification on data that these networks have not been trained on (OOD). We aim to explore how deep networks behave on OOD data and to determine if their performance is similar or different to that of humans. We thus ask which DNN architecture or framework most closely resembles human performance. With this project, we hope to shed light on notable behavioral comparisons between human performance and deep learning models and the overall generalization capabilities of deep learning models in computer vision tasks.

The main contributions of this work are summarized as follows:

1. We follow the paradigm for comparison of humans and DNNs by Geirhos et al. [8], making use of the data on human performance that they have publicly released. In that work, the authors examine a standard deep neural network (the ResNet architecture). We apply their paradigm to networks that are specifically built to handle OOD: AugMix and Push-Pull. A novel aspect of our work lies in designing a new custom network: a combination of AugMix and Push-Pull (AugMix+Push-Pull).

2. We seek to answer questions about which architecture or framework closely resembles human performance. We identify AugMix as the most human-like deep neural network under our current set of benchmarks.

3. With this work, we aim to advance our understanding of the analogy between deep learning systems and the human brain. We find that AugMix comes closest and that combining it with Push-Pull layers (AugMix+Push-Pull) pulls the network away from human-like performance. We also show that transfer learning worsens the performance of Push-Pull. Identifying these nuances in DNN performance could lead to better models of human object recognition.

This thesis is organized as follows. In Chapter 2, we review prior work on deep neural nets and performance on OOD data. We briefly explore the literature on human cognitive systems and fundamental insights from them that have ushered in a series of brain-inspired DNNs. Following this, in Chapter 3, we introduce the different methods used in this work: baseline ResNet, Push-Pull, AugMix, and AugMix+Push-Pull. We provide a detailed description of the implementation of each of these frameworks. Thereafter, we present our findings and results based on experiments carried out in Chapter 4. Finally, we conclude in Chapter 5 with a discussion of our results and outline exciting areas of future work.

Chapter 2

Related Work

This chapter will briefly explore the history of deep convolutional neural networks and look at different approaches that make them more robust to OOD data. Furthermore, we review the literature on human cognitive systems and how we derive inspiration from them to build state-of-the-art DNN architectures.

2.1 Deep convolutional neural networks

A “deep” neural network has a wide architecture corresponding to multiple hidden layers between the input and output. Convolutional Neural Networks (CNNs) are a class of deep neural networks that are popularly used for visual recognition tasks [9]. The convolution and pooling operations within these networks make them robust to translations and less prone to overfitting. The activation layer introduces non-linearities, making them capable of learning and performing more complex tasks. Originally formulated in 1998 by LeCun et al. [24], they were later popularized in 2012 by the SuperVision group who introduced a very similar but deeper architecture with more stacked layers called AlexNet [21].

In its inception, AlexNet won the ILSVRC 2010 challenge (top-1 error rate of 37.5%) and became a popular benchmark model in computer vision. Since then, several variants of convolutional neural networks have surfaced with deeper architectures and modifications to the learning process, achieving even higher accuracy. Four years later, the Visual Geometry Group (VGG) at the University of Oxford introduced VGG [31] which won second place in ILSVRC 2014 (top-1 error rate of 24.8%). This network aimed to increase its depth by using smaller-sized filters in the convolution layers. However, after VGG, many attempts to create deeper neural networks were met with a loss in accuracy and vanishing gradients. Around that time, He et al. [13] won the ILSVRC 2015 (top-1 error rate of 19.38%) with their revolutionary Residual Networks (ResNet). The ResNet model overcame the limitations of regular DNNs by skipping learning in some of the layers, allowing us to make deeper networks. Another notable class of neural network architectures is Densely Connected Convolutional Networks (DenseNet) [18]. These are a type of convolutional neural network with dense connections so that each layer can utilize information from previous layers. Due to this influx of additional information, CNNs can be compact and have fewer parameters.

2.2 Robustness to out-of-distribution data

Deep learning continues to be a growing field of research with applications ranging from computer vision to time series analysis. DNNs trained on millions of images can now achieve human-level performance when classifying images in their natural scenes, and when the distribution of the training images matches the distribution of the testing images [21]. A limitation of these networks is their ability to generalize well on out-of-distribution data. This condition is quite restrictive and does not reflect how humans learn about the world through their visual system. This limitation thus represents a barrier to artificial general intelligence: As humans, we are still able to accurately identify objects even when we are placed in new and unfamiliar environments.

Improving DNN generalization on out-of-distribution data is an extensive field of study. Researchers today are experimenting with different approaches to tackle this area by providing more robust models. Transfer learning is a common practice wherein models that have been already trained (i.e., pretrained) on massive amounts of data are reused for another related task. Researchers have leveraged this information gain and fine-tuned it for their purposes. For example, Hendrycks et. al [16] turn to pretraining and transfer learning as a way to improve DNN robustness. Their experimental setup includes using ResNets and testing it on the Canadian Institute For Advanced Research (CIFAR) dataset. Their findings suggest that pretrained models outperform their non-pretrained counterparts when faced with corrupted data and adversarial attacks. Following the authors' recommendation, we incorporate pretraining in our regular networks for enhanced model robustness.

Geirhos et al. [8] tested the generalization capabilities of DNNs by applying distortions to their test sets. They studied the classification accuracy of ResNet-50 (a ResNet architecture with 50 layers) under different training scenarios. They found that when ResNet was trained on data similar to the data it would be tested on, its accuracy was on par or slightly higher than human-level accuracy. This was also true when training was done on distorted data sets. When tested on data that was noisy or dissimilar to the training datasets, their performance dropped substantially in relation to humans (i.e. the extent to which human performance dropped for out-of-distribution data was less than the extent to which the DNN's performance dropped). Training the network on a single distortion and using it to classify other distortion types did not boost performance by much. However, training these networks longer (i.e., with more training iterations or epochs) generally improved performance. They also observed that when this architecture was trained on all but one of the distortions and tested on the remaining distortions, its testing performance on most held-out distortion categories improved. However, their performance was still worse than humans on the categories that were left out. They conclude by remarking that there are significant discrepancies in the generalization capabilities of DNNs. They, therefore, showed that the gap between a human and a DNN's ability to identify images robustly could not be closed by training on distorted datasets alone. We aim to explore alternative ways of closing this performance gap through our work than training on corruptions alone.

Along the lines of data augmentation, AugMix is a technique proposed by Hendrycks et al. [17] to improve the generalization of DNNs. This method uses a combination of simple augmentation techniques chosen at random. In this way, the model is exposed to highly diverse images during training. We discussed how DNNs tend to memorize specific distortions used in training [8]. Through this procedure, the output image from the AugMix layer now incorporates several layers of randomness. This feature sets it apart from fixed augmentation techniques like CutMix [34]. Hendrycks et al. test their training framework on CIFAR-10, CIFAR-100 and ImageNet data. Their results demonstrate that their training scheme achieves a substantially lower loss on corrupted data sets than other frameworks designed to improve classification robustness. This thesis leverages the AugMix framework in DNNs as a means to improve generalization abilities.

Most neural network architectures used today can trace their history to cognitive studies and neuroscience experiments. A natural strategy was adopted to emulate the human visual system in order to construct more robust DNNs. One of the first biologically motivated neural networks is called Neocognitron [6]. This model was presented with visual patterns repeatedly, and it learned to recognize them based on their geometric similarities. Inspired by this, Strisciuglio et al. [32] made progress in this direction with the invention of the Push-Pull layer. This novel layer is designed to mimic the human brain, making it robust to visual patterns even when they are perturbed with noise [5]. The authors implemented this paradigm using LeNet on Mixed National Institute of Standards and Technology (MNIST) data and ResNet, DenseNet on CIFAR data. This included testing different image distortions like Gaussian blur, impulse noise, frost, brightness, etc. In both cases, the Push-Pull networks consistently outperformed the regular networks. They also note that the accuracy on the clean (undistorted) testing set is unaffected by the Push-Pull layer. In our research, the aim is to better utilize the full potential of this framework with large capacity models, i.e., deeper networks with many parameters.

2.3 Models of human object recognition

Since its inception, machine learning models have been competing to outperform humans on virtually every task. However, we have also seen that humans possess excellent generalization abilities and (at least in the field of object recognition) are more robust [8]. Several lines of research have been developed to compare machine learning models with humans in order to gain insights to close the performance gap [4, 29, 26, 27, 22]. Deep neural networks are often criticized as being black box models that are difficult to interpret. Thus, efforts to investigate the similarities and differences between DNNs and humans can get us a step closer to better models of human recognition. However, the realization of brain-inspired DNNs does not necessarily mean that we exactly reproduce the complex architecture of the human brain. Instead, in our work, we try to capture important details that make humans more robust at visual recognition tasks and incorporate those details in our network architectures.

For example, Rosenfeld et al. [27] investigate how deep neural networks learn to represent images. They sought to determine if these representations are similar to those used by humans when classifying images. To do so, they carried out a series of experiments that compared humans against machines. These experiments aimed to identify the most similar image (one amongst them was indeed the 'most' identical) from a list of five candidate images. The diverse nature of this data set enabled the authors to measure if the features learned by a neural network's penultimate layers were sufficient to help it identify similar images in non-trivial settings. To do so, they extracted a series of features from an image. Different neural networks used these features to then pick an image from amongst the five candidate images. On average, they found that humans were far better at choosing the correct "similar" image than their deep neural counterparts. The authors contended that pretraining these networks on a subset of the data could reduce the performance gap. However, they concluded that the feature set learned by these networks was not generic enough to enable the kinds of learning humans do readily across domains.

Pramod et al. [26] conducted a similar experiment. In their paradigm, humans identify an image most dissimilar from a collection of pictures from the same image class in their setup. The dissimilarity measure is the time taken for humans to find this odd-one-out image. They use principal components analysis (PCA) to identify the most critical features required to determine how different one image is from another. They then use a simple linear regression model to predict the observed human dissimilarity measure through a linear combination of the top 100 PCA features. They found that CNNs outperformed all other machine learning models when accurately identifying the odd image in the lot. However, they also observed systemic differences in the representation of these images between neural networks and humans. In particular, they found that humans identify symmetric images as more distinctive in perception while those that share features to be more similar. Network models capture neither of these intricacies. The authors thus recommend that building networks that embed some of these features could result in networks with image representations that are much closer to those formed by humans.

In a survey paper, Lake et al. [22] review the fields of deep learning and cognitive science to find links between human understanding and their biologically inspired counterparts. They identify critical concepts like intuitive physics and intuitive psychology as crucial building blocks that enable children to transfer knowledge across a wide range of domains. Furthermore, they argue that a paradigm shift needs to occur within the realm of machine learning for these algorithms to become more human-like. Building causal models of the world to understand, query and make decisions about interacting with their environment is paramount. The authors make an interesting observation: While most deep learning networks can supersede human performance on many tasks, they take substantially longer to learn these patterns. Furthermore, these networks are often limited in utilizing knowledge learned in one domain to speed up learning in another. However, recent developments provide promising approaches to enable faster learning in unfamiliar terrain. Attention is one such development. Drawing inspiration from the human perceptual system, deep networks embedded with attention has shown substantial improvements within fields like machine translation. This has also allowed researchers to cut down training times significantly, as not all network parameters need to be updated (matching our intuition about attention).

Several approaches have been developed to measure the extent to which an artificial neural network is similar in behaviour to the human brain. In [29], the authors develop a scoring mechanism called 'Brain Score.' This measure determines resemblance using two criteria: neural metrics and behavioral metrics. The neural benchmarks compare intrinsic similarities between the activations in DNNs and the activations in the brain. Behavioral metrics examine the similarity in the responses of ANNs with humans. They find that DenseNet-169 and ResNet-101 are the most brain-like ANNs. They show a correlation between high-performing DNNs on ImageNet classification tasks and a higher Brain Score. However, this correlation exists only up to a certain threshold. Beyond this threshold, networks tend to vary considerably in brain score.

In conclusion, we have surveyed different DNN architectures, various methods that increase model robustness and models of human object recognition. The upcoming chapters, therefore, aim to study neural networks derived from these different approaches. Then we compare its performance on the distortions developed by Geirhos et al. to that of published publicly available human performance datasets using statistical hypothesis testing.

Chapter 3

Methods

This chapter describes the dataset, image distortions, the DNNs we employ, and human experiments. In terms of DNNs, we employ three established DNNs: ResNet, AugMix and Push-Pull [13, 17, 32]. We also develop a novel network: a combination of AugMix and Push-Pull (AugMix+Push-Pull). This lays the groundwork for our experiments and results.

3.1 Study dataset

In order to assess and make robust comparisons of deep learning models, we need massive high-quality and curated benchmark datasets. In the field of computer vision and object recognition, ImageNet [28] is a notable benchmark dataset with over 14 million images categorized into 22,000 categories. This database contains hand-annotated images arranged according to the hierarchy of WordNet [25]. WordNet is a lexical database of English words where each category represents synonym sets (often called 'synsets'). Each synset is a single concept characterized by these synonymous words, for example: 'dogsled', 'dog sled', 'dog sleigh.' There are over 100,000 synsets in WordNet, and the ImageNet database contains around 1,000 images for each such synset.

In its early stages around 2009, ImageNet was mainly populated based on captions or tags using search engines. This resulted in a significant number of mislabelled images. To overcome this, researchers used Amazon Mechanical Turk extensively to obtain more accurate annotations of the images. Due to the volume and wide variety of images, this visual database has been instrumental in advancing research in computer vision, and the development of methods in big data [2]. It is popularly used in smaller sizes. Take, for instance, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), where a subset of 1,000 categories from this dataset is used. The ILSVRC consists of an annual competition and a benchmark dataset in object detection and recognition. To provide contrast, this downsampled version has only three people categories: scuba diver, bridegroom, and baseball player, compared to 2,832 people categories under the person subtree in the complete ImageNet data. In 2012, ILSVRC made the ImageNet dataset publicly available for non-commercial use.

For our research, we make use of the ILSVRC 2012 database [28] that has 1,000 categories, with some being fine-grained like complex dog breeds such as Staffordshire bullterrier or Yorkshire terrier. This could present difficulties in carrying out human experiments as they tend to pick broader categories like cat or dog than name a specific breed. This was the same paradigm followed by Geirhos et al. [8] where they created a dataset called '16-class-ImageNet' consisting of 16 categories. They mapped these entry-level categories to the corresponding ImageNet categories using the WordNet hierarchy. The different categories they used were airplane, bicycle, boat, car, chair, dog, keyboard, bear, bird, bottle, cat, clock, elephant, knife, truck, oven. The main crux of this thesis is to compare the performance of our deep neural networks to human experiments carried out by Geirhos et al. [8]. Thus, to make fair comparisons, we create a new dataset called '15-class-ImageNet' that groups a subset of ImageNet classes into 15 entry-level categories. This dataset has all the same categories as Geirhos et al. (airplane, bicycle, boat, car, chair, dog, keyboard, bear, bird, bottle, cat, clock, elephant, knife, truck) except for oven. For a brief overview, Figure 3.1 displays one example image per category.

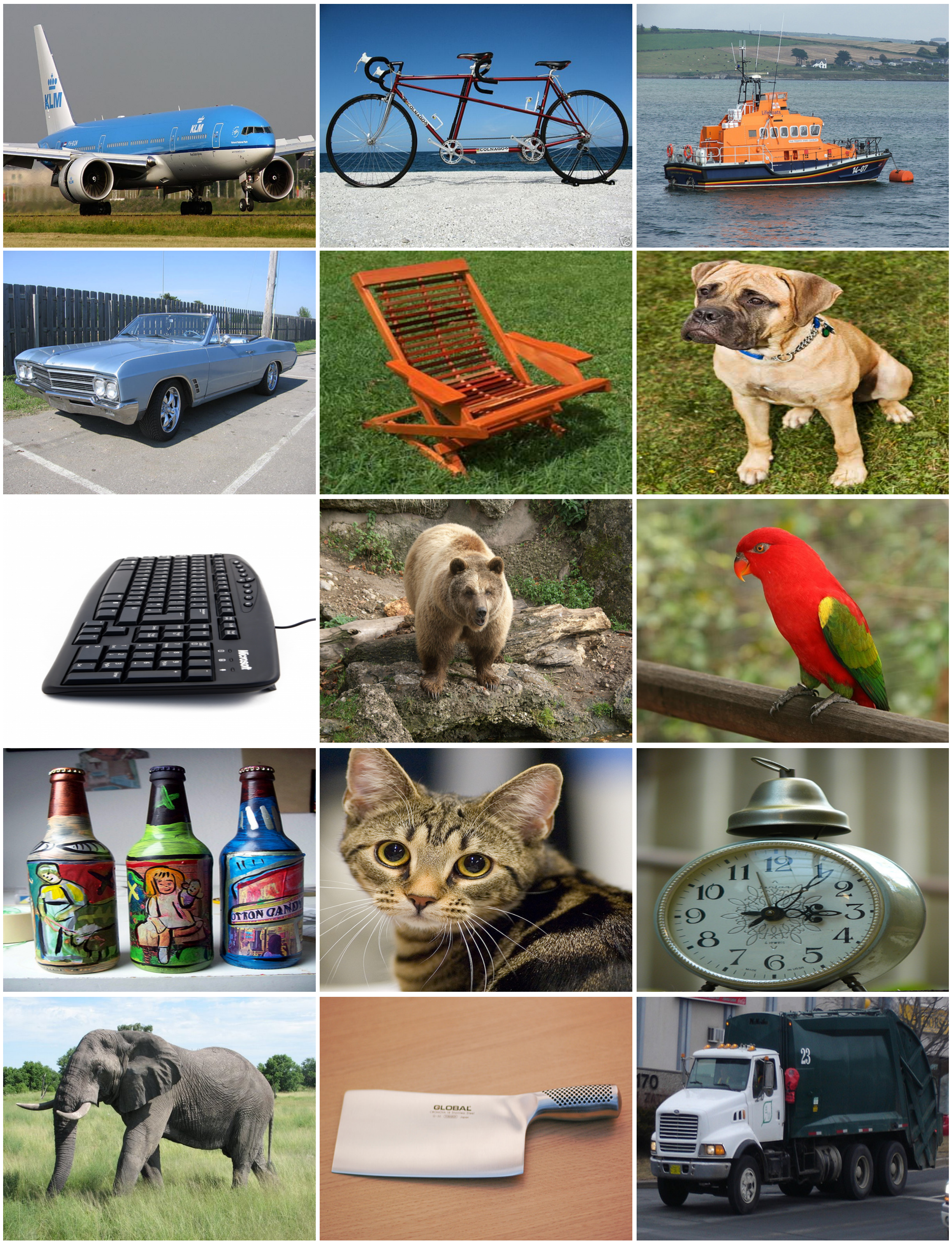

Figure 3.1: Example stimulus image from each of the 15 classes. From left to right, the classes are: airplane, bicycle, boat. Second row: car, chair, dog. Third row: keyboard, bear, bird. Fourth row: bottle, cat, clock. Bottom row: elephant, knife, truck [28].

3.2 Image pre-processing and distortions

The complete ImageNet data is 1.3 terabytes in size. To deal with a dataset of this volume, we depend heavily on supercomputing, cloud services, and GPUs throughout this work. Like some real-world data analysis tasks with big data, a portion of our time and efforts were focused on extracting and preparing the data in the proper format. The first step in the data collection involved acquiring this large dataset. We used Amazon Web Services (AWS) to download the ImageNet data and transfer it onto our Compute Canada servers. The entire dataset came in the form of one large tar file, and within that sizeable file, we had multiple unlabelled tar files, each corresponding to a specific category. The next step required launching a bash shell script on our compute servers to extract the desired images and group them into different classes. This constitutes the clean (undistorted) dataset with a total of 130,450 images. This is the only data filtering that we carry out, and we assume that most of the images are correctly labelled with the dominant object in the image. Figure 3.2 is a visual representation of the distribution of images across different categories, where we observe class imbalance.

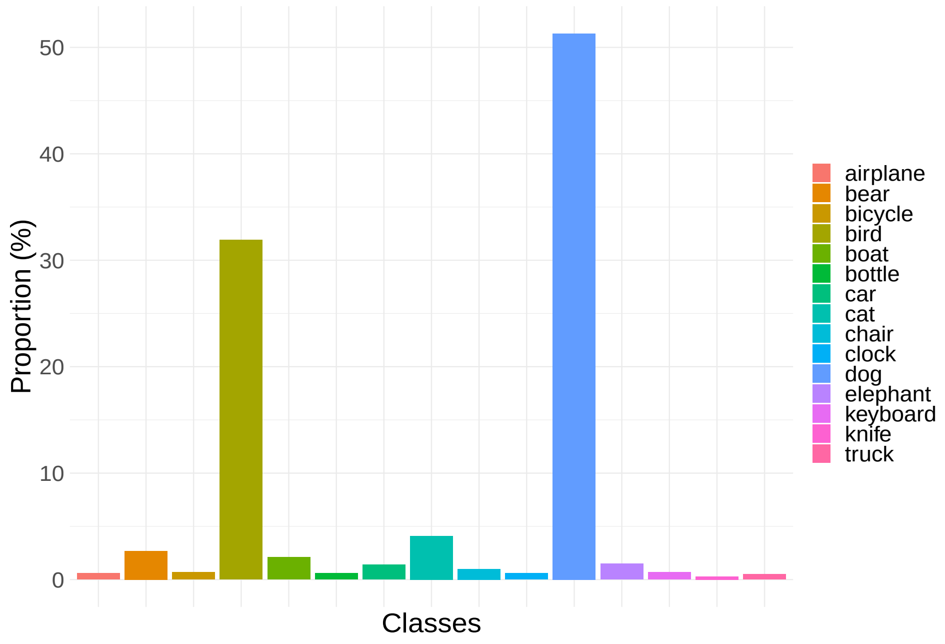

Figure 3.2: Distribution of images across the different classes.

In image classification tasks, each greyscale image is a two-dimensional matrix of pixels that each takes a value between 0 and 255. Values closer to zero are darker patches, while values closer to 255 represents brightness. A color image has separate red, blue, and green channels (RGB) represented by three of these two-dimensional matrices stacked on top of one another. We used Python (version 3.6.3) for all data pre-processing related tasks. We used the `skimage` module to resize the dimensions of the images to (224 x 224 x 3), which denotes its height, width, and depth. This preserves the amount of information and is the input shape expected by most current DNNs [13].

The final step in this process included applying parametric image manipulations on the clean (undistorted) dataset to generate different sets of distorted images. Code for these distortions is provided by Geirhos et al. [8] under an open-source license. However, they do not provide the distorted images themselves, so we recreated them using their code. In this study, we will consider six image distortions: greyscale, contrast, rotation, uniform noise, low-pass and salt-and-pepper noise. These distortions are described below. Figures 3.2, 3.3, and 3.4 depict examples of the stimuli image across three arbitrary categories (`bird, dog, car`) for all distortion types.

This selection is a subset of the image distortions used by Geirhos et al. [8] (technical issues prevented us from using the remaining distortions: phase noise and high-pass). This enables us to compare our findings with that of human performance assessed by Geirhos et al. across these distortions. Moreover, this set encompasses a wide range of perturbations that may occur as a result of real-world conditions like faulty machinery, say. This will help us make more robust comparisons that simulate practical settings. We made use of Compute Canada to generate the six sets of corrupted images. This allowed us to parallelize the computational load using a series of python scripts for each distortion and store six times the original data size on the server.

Greyscale. In greyscale images, each pixel value only represents the amount of intensity that ranges from black (low intensity) to white (high intensity). We used the `rgb2gray` method in the Python `skimage` module to convert images into greyscale.

Contrast. Contrast refers to the differences in color or grayscale between different aspects of an image. Having a high level of contrast makes objects in an image more discernible than a low level. Applying a contrast level k (in percent) on a normalized image ($[0, 1]$ range) gives us the scaled image with each pixel value given as below. For our experiments, we scale the images to a contrast level of 5%, according to the following equation from [8]:

$$\text{scaled\_value} = \frac{1 - \frac{k}{100\%}}{2} + \text{initial\_value} \cdot \frac{k}{100\%}$$

Rotation. Rotation is a standard geometric distortion as well as a data augmentation technique. It turns an image in a clockwise or counterclockwise direction by an angle θ. In our case, we rotate the images in the anti-clockwise direction by 90°. We do this by transposing the original image matrix and reversing the column order.

Uniform noise. Images can be noisy from using a low-resolution camera lens. This distortion can be modelled as uniform noise. It has also been shown that degradation in DNN performance is particularly evident for additive noise or blur distortions [3]. The width parameter w is adjusted to tune the range $[-w, w]$ of additive uniform noise added to each pixel separately. Here, we first apply a contrast level of $k = 30\%$ on the original image and then white uniform noise with $w = 0.1$ is added pixel-wise (i.e., to each pixel).

Low-pass (blurring). Blurring is a common image distortion that can occur due to a camera being out of focus or moving objects. Moreover, it can simulate DNN performance on distant objects captured using poor camera sensors. Low pass or Gaussian filtering provides a useful model for this type of image perturbation. It is controlled using the standard deviation parameter of the Gaussian filter. We used the `scipy.ndimage.filters.gaussian_filter` function with std = 7 for our low-pass experiments.

Salt-and-pepper noise. An image corrupted by this noise has dark pixels in the brighter areas and bright pixels in the darker patches giving the semblance of sprinkled salt and pepper. Also referred to as impulse noise, this may happen during image processing or disruption in the transmission process. This image manipulation involves setting pixels to black or white determined by the probability parameter p. First, we scale the original image to a contrast level of 30% and then apply salt-and-pepper noise with $p = 0.2$.

Although these distortions encompass many real-world scenarios, it is possible to encounter new, unfamiliar image corruptions under a different task setting. For instance, the athletic apparel company Lululemon may want to identify their logo across images scraped off the web. However, their logo might be crushed or scrunched up in some of the candidate images under study. Our models, however, are not exposed to these types of natural and realistic distortions. They were only trained on well-studied distortions to assess their performance with those of humans fairly. Hence, our current image perturbations are not fully representative of all practical distortions, which is a potential limitation of our work.

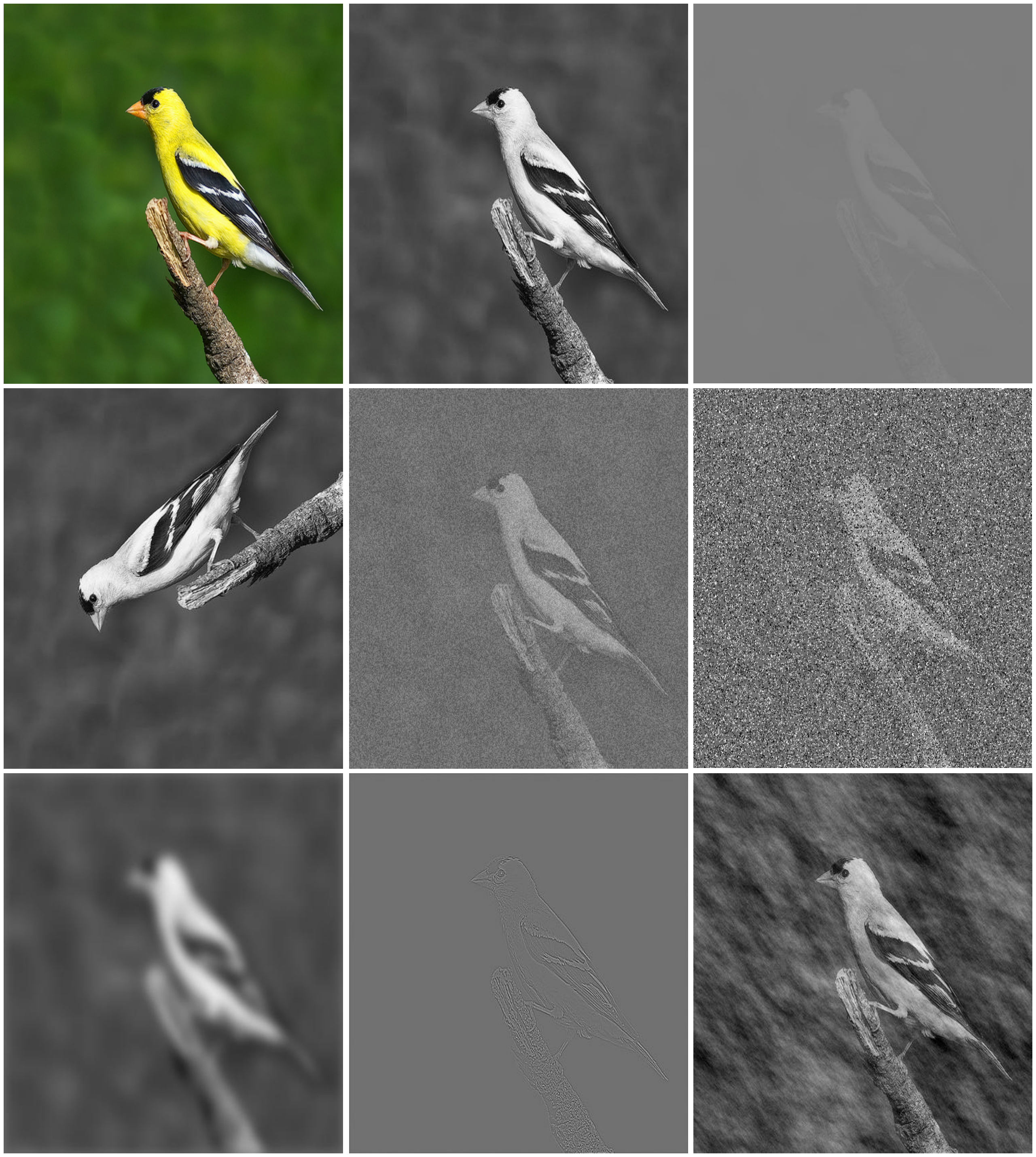

Figure 3.3: Example stimulus image of class `bird` across all distortion types. From left to right, image manipulations are: color (undistorted), greyscale, low contrast. Middle row: rotation, uniform noise, salt-and-pepper noise. Bottom row: low-pass (blurring), high-pass, phase noise [28].

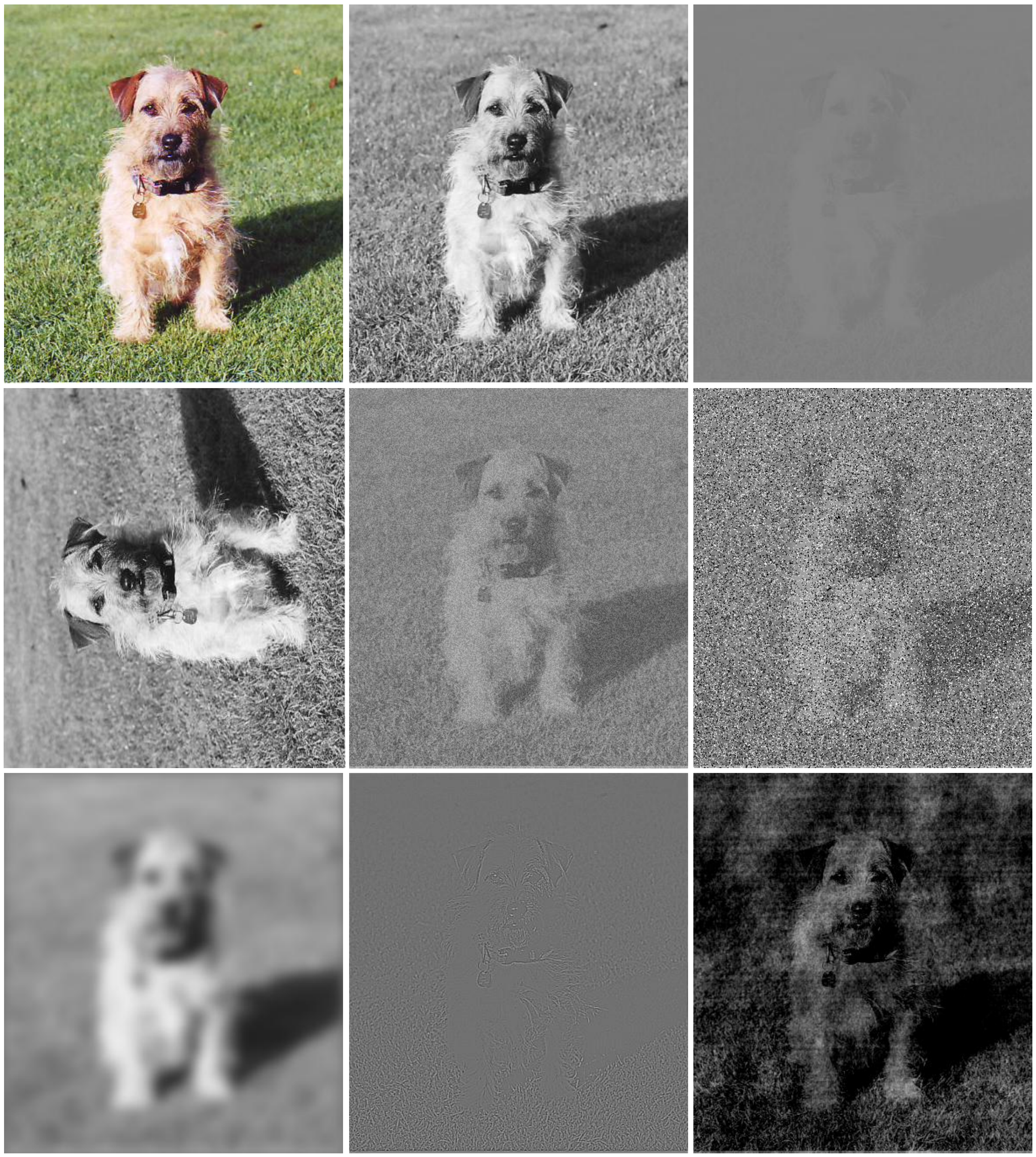

Figure 3.4: Example stimulus image of class `dog` across all distortion types. From left to right, image manipulations are: color (undistorted), greyscale, low contrast. Middle row: rotation, uniform noise, salt-and-pepper noise. Bottom row: low-pass (blurring), high-pass, phase noise [28].

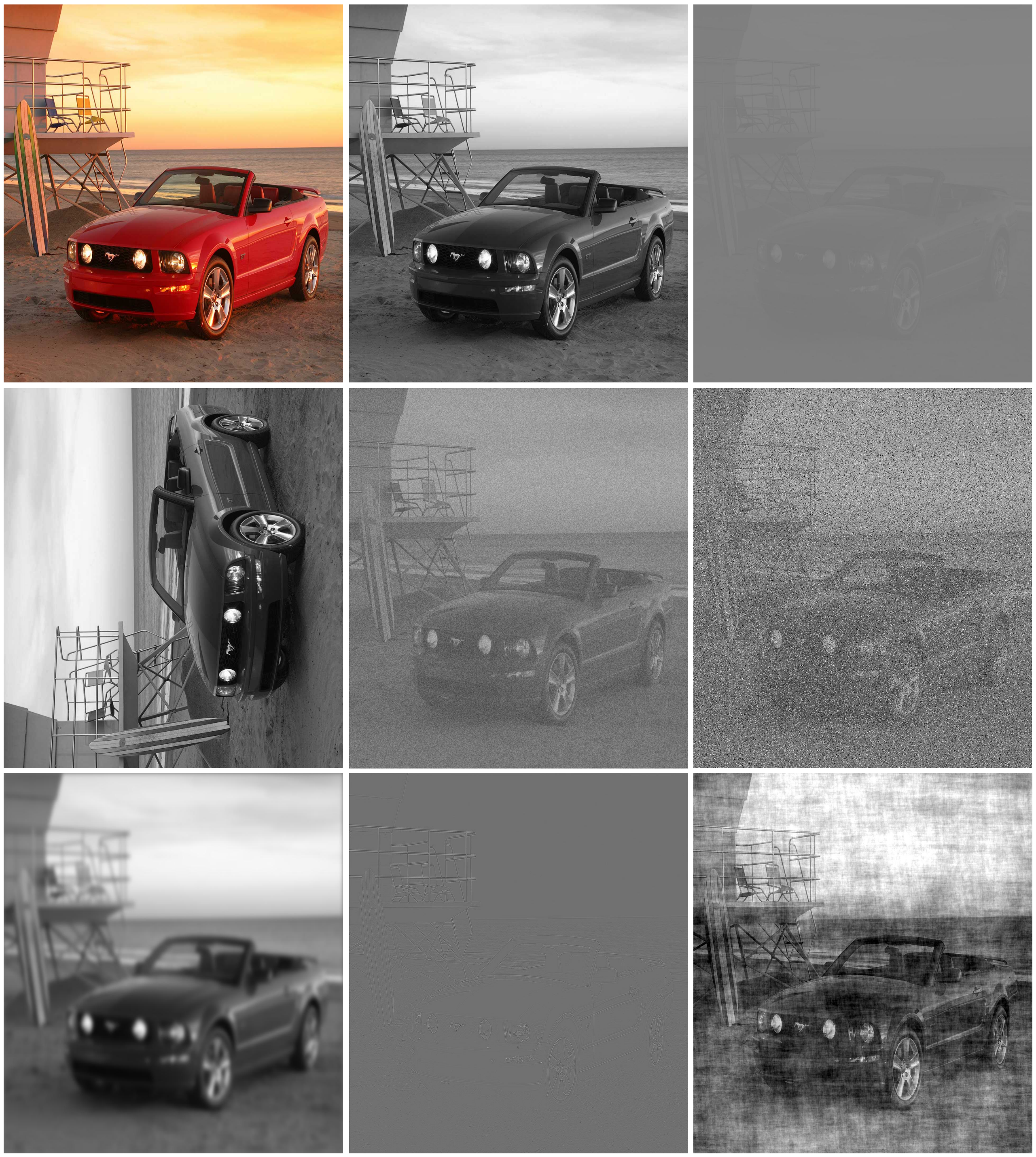

Figure 3.5: Example stimulus image of class `car` across all distortion types. From left to right, image manipulations are: color (undistorted), greyscale, low contrast. Middle row: rotation, uniform noise, salt-and-pepper noise. Bottom row: low-pass (blurring), high-pass, phase noise [28].

3.3 Residual neural networks

Deep Neural Networks (DNNs) are a class of neural networks that have several layers between the input and output layers, hence the name "deep". Typically for complex tasks, DNNs leverage hierarchal feature extraction to learn low/mid/high-level representations [9]. Unsurprisingly, deeper networks achieve high accuracy on a wide range of datasets [31]. Following this rationale, a natural question arises —"Will adding more and more layers always lead to improved performance?" He et al. [13] found that as they increased the network depth, the accuracy reaches a plateau and then suffers a steep decline. This phenomenon was not a result of overfitting whereby the model learns the training data well (low training error) but is unable to generalize to unseen data (higher test error). In their case, they observed a high training error and consequently a higher test error.

In an attempt to tackle this, He et al. [13] proposed that one can create a deeper network by taking a shallow network and adding extra layers that simply learn the identity function. The resulting architecture should then give the same output as the shallow model and have at least the same accuracy. However, their experiments showed that it was difficult for the networks to learn the identity function and not improve performance. Instead, they hypothesized that it was easier for the network to learn the deviations from the identity mapping, i.e., residuals, than the function itself. In other words, it is easier to come up with a solution like $F(x) = 0$ rather than $F(x) = x$ using stacks of convolution and activation layers. Here, the layers are trying to learn the residuals and are collectively termed as a *residual block* (ResBlock; [13]). These residual blocks form the crux of the deep learning residual framework, and the networks following this paradigm are called *Residual Neural Networks* (ResNets; [13]).

3.3.1 Residual blocks

To better understand the ResNet architecture, we need to first dig deeper into the building blocks of this network shown in Figure 3.5. Let us call our input x. In a regular DNN (shown on the left side), this input x passes through all the layers, and the weights for the output $f(x)$ are learned directly for each layer. On the right portion of the figure, the ResNet 'skips' the learning for $f(x)$ in the dotted region and instead needs to learn the residuals $f(x) - x$. The input x is then added to the residuals $f(x) - x$ to get the desired output $f(x)$. This is called a residual or skip connection.

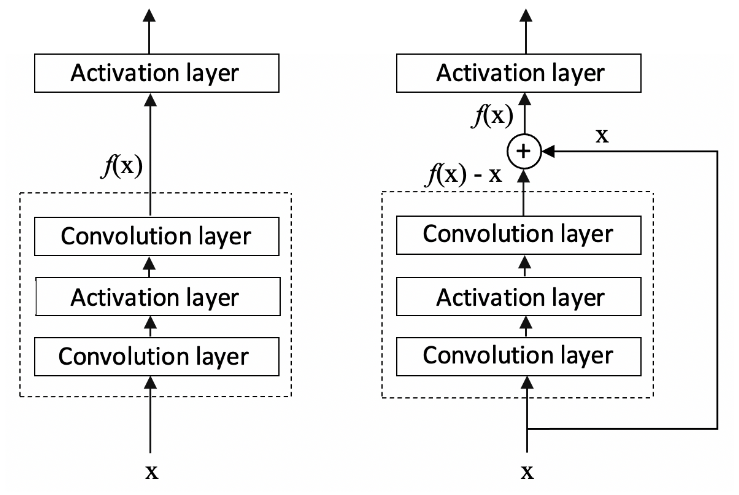

Figure 3.6: From left to right: a regular block and a residual block (ResBlock).

As mentioned earlier, a deeper network can be constructed by adding layers with their output equal to their input. This allows the network to maintain the same accuracy. Thus, if the goal is to get $f(x) = x$, then the ResNet needs to learn the residuals $f(x) - x = 0$. This implies pushing the weights and biases within the dotted region of the figure to zero, which is easier than learning a specific mapping. This is the fundamental concept that ResNets are built upon, with each skip connection constituting a residual block.

The addition of skip connections also helps the network overcome the infamous vanishing gradient problem. In the absence of this shortcut, the gradients tend to zero due to repeated chain rule during backpropagation. In contrast to this, ResNets mitigate this issue by allowing the gradients to skip layers.

3.3.2 ResNet-50 Implementation

Let us briefly explore the architecture of ResNet-50 network as shown in Figure 3.6. This will serve as the baseline model for our experiments. The “50” indicates that it is composed of 50 layers and has over 23 million trainable parameters [13]. In addition to that, it has been pretrained on millions of images from ImageNet. It is a widely employed network for image recognition tasks due to its deep architecture and transfer learning features. The same architecture was also used by Geirhos et al. [8] to compare with human performance. All of these aspects collectively make it a good benchmark model.

The learning is done in 5 stages [13]. The first stage is a convolution layer followed by a pooling layer. Each of the next four stages corresponds to a residual block. Every residual block has 3 layers with both 1×1 and 3×3 convolutions. The convolutional layer is followed by a common non-linear activation function called Rectified Linear Unit (ReLU). It is defined as $g(z) = max(0, z)$ that sets negative inputs to zero and anything greater than 0 remains the same. ReLU activation function also overcomes the vanishing gradient problem, allowing models to learn more efficiently and perform better. Within each residual block, we skip three layers and then add the input before the final ReLU function.

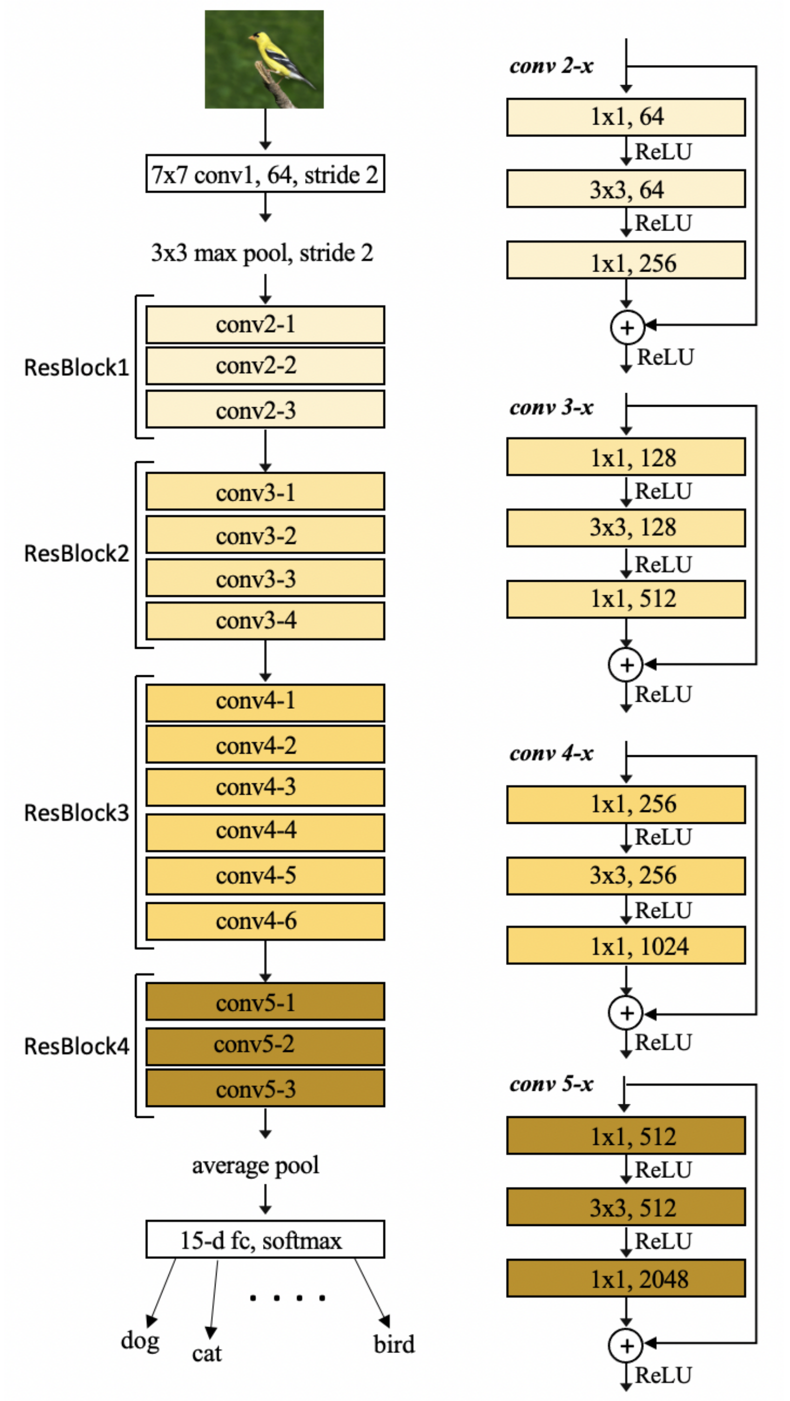

Figure 3.7: The ResNet-50 architecture.

3.4 AugMix networks

AugMix is a framework designed to train DNNs to be more robust to out-of-distribution data. AugMix takes as training input an image, a set of transformations and a number k. Here, k denotes the number of repetitions of the augmentation procedure. Next, it initializes a matrix of zeros (called augmentation matrix) and draws k observations from a Dirichlet distribution with parameter α (affecting the skewness of the distribution over the different types of operations).

For each repetition, AugMix randomly chooses three operations from a transformation set. It uses these operations, either by themselves or by composing them together to create a transformation chain. By default, the algorithm creates three transformation chains. Each of these chains can vary in length (number of compositions). Then, one chain from amongst them is uniformly selected at random. The augmentation matrix is populated with the product of the weight times the output formed by feeding the original image into the selected augmentation chain.

Once all iterations are complete, another sampling step is performed. A value m is randomly drawn from a beta distribution. This beta distribution is also parameterized by the same α that the Dirichlet distribution is parameterized by. This mixing proportion m combines the original image with the augmented image from the chain. The image output by this algorithm is a weighted combination of the original image and augmented image (represented by the augmentation matrix). The output image from the AugMix layer now incorporates several aspects of randomness ranging from the type and intensity of operations, length of the chain and the mixing proportion. Figure 3.7 is an example realization of the AugMix process described above.

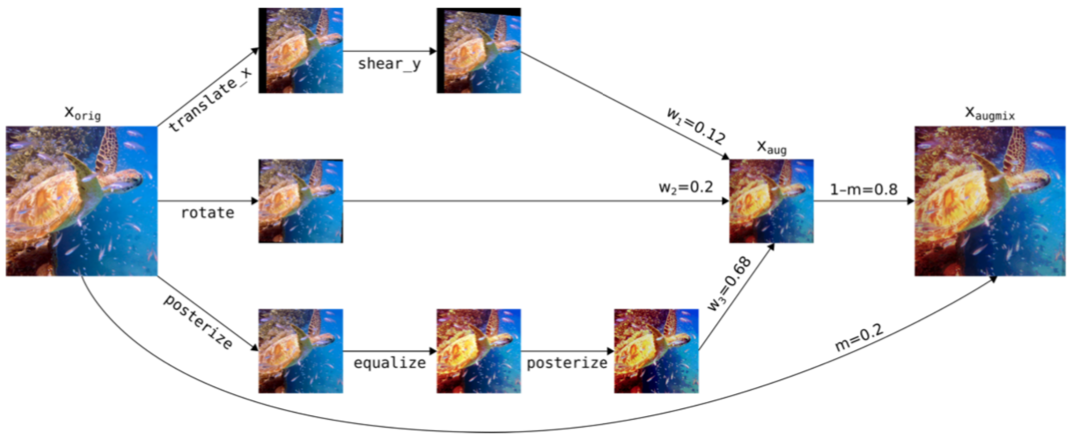

Figure 3.8: An example of AugMix. The final image is a weighted combination of the original and the augmented images. Diagram taken from [17], used with permission.

To train a neural network on these transformations, Hendrycks et al. compute the Jenson-Shannon Divergence Consistency (JSD) loss. These images (original and augmented versions) can be thought of as inputs into a posterior distribution. These distributions assign a probability to a particular class label, given the input image received. The JSD loss function seeks to minimize the difference in the posterior distributions induced by the original image, and its augmented variant [17]. This process ensures that models learn to become insensitive to a diverse range of variations in the input when predicting class labels.

3.5 Push-Pull networks

The Push-Pull layer [32] is a novel layer designed to simulate the functions of the V1 (the primary visual) area [19] of the human brain [33]. The primary goal is to capture a phenomenon known as push-pull inhibition. Here, a specific part of the brain called the excitatory receptive field is excited by a positive stimulus (push). Another area called the inhibitory receptive field attempts to hinder this response with a negative stimulus (pull). Due to this natural response mechanism, the human visual system can respond to visual stimuli even in the presence of noise like snow, fog, blur, etc.

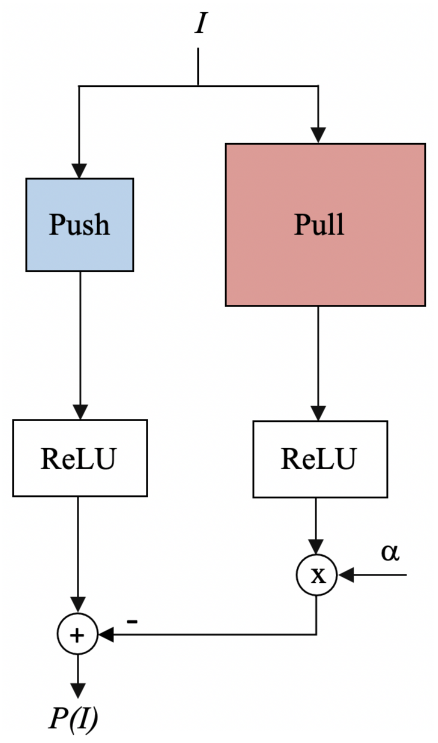

Figure 3.9: Schematic representation of the Push-Pull inhibition. The push kernel represents the excitatory receptive field, whereas the pull kernel mimics the inhibitory receptive field.

In order to model these processes, the Push-Pull layer has two convolution kernels (or filters) called the push and the pull kernels as shown in Figure 3.8. Analogous to the human brain, the push kernel represents the excitatory receptive field while the pull kernel mimics the inhibitory receptive field. In practice, the pull kernel is larger than the push kernel. The weights for the push kernel are learnable parameters. For the pull kernel, we compute the weights by inverting the pull weights and upsampling by a factor h. The response from each of these kernels is activated non-linearly using ReLU (explained in Subsection 3.3.2). The output of this layer is computed by subtracting a fraction α of the pull response from that of the push response as summarized in Equation 3.1 from [32]. We used the default settings for $h = 2$ and $\alpha = 1$ [32].

$$P(I) = \theta(k * I) - \alpha \cdot \theta(-k_{\uparrow h} * I) \tag{3.1}$$

Here, θ is a rectified linear unit (ReLU) activation function, α is the inhibition strength of the pull kernel and h denotes the upsampling factor of the push kernel k, where $h > 1$. To implement the Push-Pull network, one can replace the first convolution layer in existing DNN architectures like ResNet-50 as shown in Figure 3.9. Although this layer could be placed virtually anywhere in the network, this particular setting is recommended to better model the initial stages of brain activity where low-level visual features are processed.

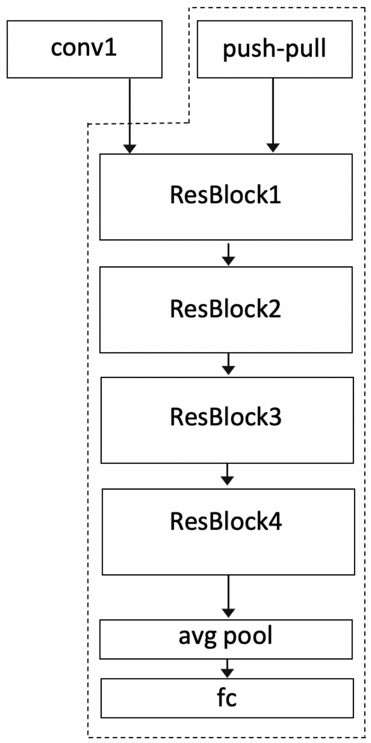

Figure 3.10: Modified ResNet-50 architecture with a Push-Pull layer added.

3.6 Human performance from Geirhos et al.

Our main goal is to compare the performance of deep networks to humans on classifying OOD data. We use human performance data from Geirhos et al. [8] and closely follow the conditions of their experiments. This is done to perform a fair comparison of their results with those from our experiments using AugMix, Push-Pull, and our novel AugMix+Push-Pull. Geirhos et al. provide their data under a Creative Commons Attribution 4.0 International license, and we acquired these data from GitHub[1]. These data are publicly available, and we rely exclusively on secondary use of these anonymous data. Geirhos et al. chose to carry out a controlled lab experiment instead of using third-party services like Amazon Mechanical Turk to enroll human participants. This gives them the flexibility to design and control the conditions of these experiments to make fair comparisons with deep neural networks.

Geirhos et al. make use of the ImageNet Large Scale Visual Recognition Challenge (ILSRVR) 2012 database [28] that has millions of images with over a thousand categories. However, in order to draw close behavioral comparisons with humans, this ImageNet data was down-sampled to 16 distinct and broader-level categories as mentioned in Section 3.1. A total of twelve distortion types were applied to record human accuracies on this data. Some were a set of binary image manipulations like color vs. greyscale, true color vs. false color, original vs. equalised power spectrum. Other common image distortions included contrast, rotation, phase noise, uniform noise, low pass, high pass, and salt-and-pepper noise. Additionally, they considered three Eidolon stimuli [20] which correspond to a type of parametric image distortion. Note that there is no human data available for salt-and-pepper noise.

In Geirhos et al., three observers participated in the colour and contrast experiments. Six observers were selected for the opponent colour, high-pass filter, low-pass filter, phase noise, and power equalisation experiments. In the uniform noise and Eidolon experiments, five observers had participated. For every human experiment, a blank screen with a central fixation square was presented for 300s. The image then appeared on the screen for another 200s, followed by a noise mask that lasted 200s. Finally, the observers were given 1500s to make their choice. A forced-choice image categorization paradigm was implemented where the participants are asked to click on the category they thought the displayed image resembled the most. The rationale behind the background noise mask is to prevent the feedback loops in the human brain from being activated. Deep neural networks rely on feedforward connections, whereas the human brain is more dynamic in nature, having recurrent loops of information [23]. For more details about this paradigm, see [8].

[1]`https://github.com/rgeirhos/generalisation-humans-DNNs/`

In order to make comparisons, Geirhos et al. trained different networks on clean (undistorted) data as well as on each of the distortions. For our comparative analysis, we choose the paradigm where networks are trained on clean (undistorted) data only and tested on different types of distortions. We believe that this setting is most representative of the real world where machine learning models are deployed. It is also a better test of model robustness since most tasks are trained on clean images, and we are uncertain of the exact distortion we might encounter. Extending Geirhos et al. we perform this classification and experiment and compare to their human data using networks designed to handle OOD data (instead of just the ResNet, as was done in Geirhos et al.).

As we conclude this chapter, we have covered the foundational concepts of the ResNet-50 architecture, AugMix framework and Push-Pull network. We also briefly outlined the essential details of the human experiments carried out by Geirhos. There is a whole host of model robustness techniques, and we will focus on these methods for the experiments discussed in the next chapter.

Chapter 4

Experiments and results

In this chapter, we outline the steps taken for training our current set of networks. We then present our results using non-pretrained and pretrained versions of these experiments. Thereafter, we discuss how we compare the performance of DNNs with humans using different statistically rigorous testing procedures.

4.1 Training procedures

We carried out extensive experiments with four types of DNNs (baseline ResNet, AugMix, Push-Pull and AugMix+Push-Pull). Each DNN has an underlying ResNet-50 architecture with 15 output neurons to match the number of categories in the dataset. Besides the baseline ResNet, we make modifications to the other DNNs as described in Chapter 3. AugMix includes a custom data augmentation scheme during the training stage. The Push-Pull DNN replaces the first convolution layer in ResNet-50 with a Push-Pull layer. We include a fourth network that is a combination of AugMix and Push-Pull. We also examine a novel merger of these two paradigms: (AugMix+Push-Pull).

In this work, we make use of a single train/test split in a 75:25 ratio. For each network, the training is done on the clean (undistorted) training set. We choose the default values specified by Geirhos et al. [8] for most of the hyperparameters. To maintain consistency, we employ the same configurations for each architecture summarized in Table 4.1. Also, we use stochastic gradient descent (SGD) as our optimizer. We will now explain each of these training choices briefly.

Table 4.1: Hyperparameter configurations used during training.

Hyperparameter	**Configuration**
Epochs	15
Batch size	32
Learning rate	0.1
Momentum	0.9
Weight decay	0.0001

Batch size is the number of training examples used in a single iteration (one forward/backward pass), whereas an **epoch** is one complete pass through the entire training dataset.

Gradient descent is an optimization algorithm that determines the weights of a network that minimize the error function. It does so by taking a step in the direction towards the lowest point of this function. This direction is opposite to the gradient (or slope) hence the name 'gradient descent'. Gradient descent uses all of the training examples in each iteration to compute gradients and update the model's weights. **Stochastic gradient descent**, on the other hand, introduces randomness in the gradient descent process by selecting one sample point (or a subset of points) at each step.

A hyperparameter called the **learning rate** determines the magnitude of the descent, i.e., step size. In other words, the learning rate controls how much a model's weight changes during training. A large learning rate might miss local minima but converges quicker. Conversely, a small learning rate avoids skipping over any local minima but takes longer to converge.

Classical SGD usually finds it difficult to navigate towards the optimum when stuck in a ravine. Ravines are regions where the surface curves more steeply in one direction than in another [9]. It tends to oscillate in those regions and ends up finding a local minima. Here, the error is low but not the lowest. **Momentum** overcomes this phenomenon by pushing the gradients in the right direction and out of the regions it was stuck oscillating in.

Weight decay or regularization is applied to a model's weights in order to penalize its complexity. This technique avoids overfitting by constraining a network's weights and forcing it to learn mostly generalizable features.

We used the following hyperparameter values as listed in Table 4.1. We set our batch size to 32. We thus used 32 training examples are used to compute the error gradient before updating our model's weights. We trained all of our models for 15 epochs. Each network, therefore, makes 15 complete passes through the training dataset. We observed validation accuracy plateauing after 12 epochs (shown in Figure 4.1). Hence, we found 15 epochs to be sufficient for training our models. We were also bound by time and budget constraints in terms of model training. A learning rate of 0.1 is employed so that the model can learn well within our training time. We pick a momentum value of 0.9, adding 90% of the previous gradient update to the new gradient update. We use a weight decay of 0.0001. This value forces different networks under consideration to generalize well during our limited training time. Once training is complete, we evaluate each model on unseen test data for each distortion type. Note that the test set for each distortion contains the same images distributed across different classes. We evaluate model performance using `top1` accuracy in PyTorch, i.e., the model's highest probability class equals the correct target label. This metric is the same as base classification accuracy.

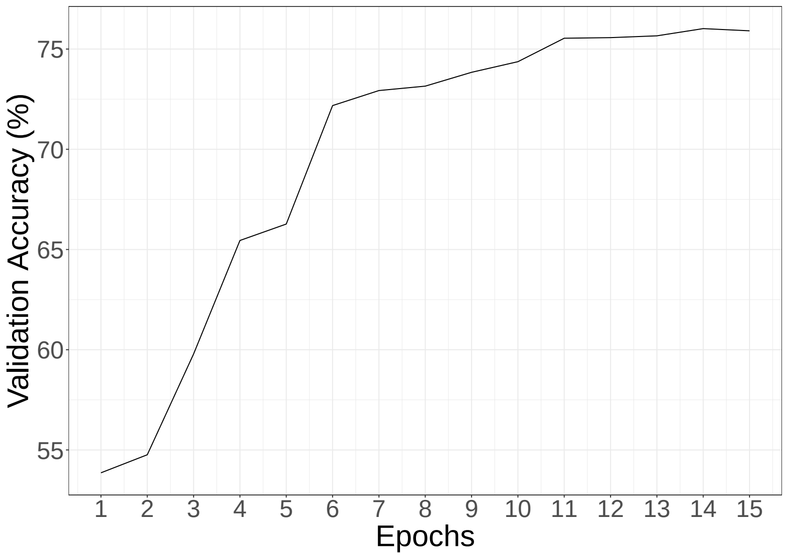

Figure 4.1: Validation accuracy versus epochs for baseline ResNet. The accuracy plateaus after 12 epochs, indicating convergence of the training.

These experiments are done for both the non-pretrained and pretrained versions of the networks. In case of no pretraining, we trained the networks from scratch using our subset of ImageNet (15 categories). In the case of pretraining, we use weights for all the layers (except the fully connected layers) that have been published for ResNet and trained using all of ImageNet. These weights are publicly released along with the PyTorch package[1]. This is done to adjust for the 15 classes in our dataset as ResNet was originally trained on 1,000 categories of ImageNet. For Push-Pull, we also train the Push-Pull layer and the first two residual blocks in addition to the fully connected layers. This ensures that residual blocks after the Push-Pull layer can adapt to the new responses, as the output from a

[1]`https://pytorch.org/hub/pytorch_vision_resnet/`

Push-Pull layer is different from that of a usual convolution layer. All of our networks were implemented in PyTorch. We conducted our analysis using high-capacity NVIDIA Tesla V100 SXM2 GPUs provided by Compute Canada.

4.2 Results

We examined the performance of several variations of the ResNet architecture with pretraining and no pretraining. In Table 4.2, we report the classification accuracy (in percent) for all the models and humans across the different image manipulations on the 15-class ImageNet dataset.

Table 4.2: Classification accuracy (in %) for pretrained as well as non-pretrained networks are included, along with human accuracies from [8].

Distortion	**Baseline ResNet**		**AugMix**		**Push-Pull**		**AugMix + Push-Pull**		**Humans**
	Non-Pretrained	Pretrained	Non-Pretrained	Pretrained	Non-Pretrained	Pretrained	Non-Pretrained	Pretrained	
Clean (undistorted)	76.02	84.58	88.59	99.50	85.71	79.39	82.01	98.19	88.5
Greyscale	70.94	80.20	84.59	97.94	83.04	74.85	77.54	95.56	86.6
Contrast (5%)	32.05	32.04	38.28	55.75	32.05	31.43	32.08	47.62	47.6
Rotation (90°)	55.25	64.31	64.87	85.44	65.48	64.86	58.18	76.94	78.5
Uniform (0.1)	41.8	49.27	57.68	69.25	53.00	54.06	67.03	81.98	45.6
Low-pass (std=7)	47.73	34.31	62.00	59.41	46.51	34.01	71.04	74.89	48.5
Salt-and-pepper (0.2)	32.36	32.19	53.20	51.05	32.76	45.89	44.07	41.27	NA

4.2.1 Non-pretrained networks

In Figure 4.2, we show the mean classification accuracy (mcA) for the non-pretrained networks across all distortions (excluding clean data). We see that humans continue to substantially outperform the baseline ResNet as has been shown in previous papers [8]. However, the strongest model in our current set of DNNs is AugMix. We observe that AugMix exhibits the highest mean corruption accuracy and is comparable to humans that are fairly robust to distortions. This indicates that, on average, AugMix is able to generalize in a more human-like way than other networks. We note that AugMix is closely followed by the combined network AugMix+Push-Pull with competing mean accuracy to humans. Although Push-Pull has a lower accuracy than humans, it still shows a considerable improvement over the baseline ResNet. The baseline ResNet has the worst performance overall on distorted data. These findings are particularly interesting as AugMix and Push-Pull are DNNs designed to handle OOD. These modified network architectures do indeed provide an accuracy boost to regular DNNs in case of image distortions.

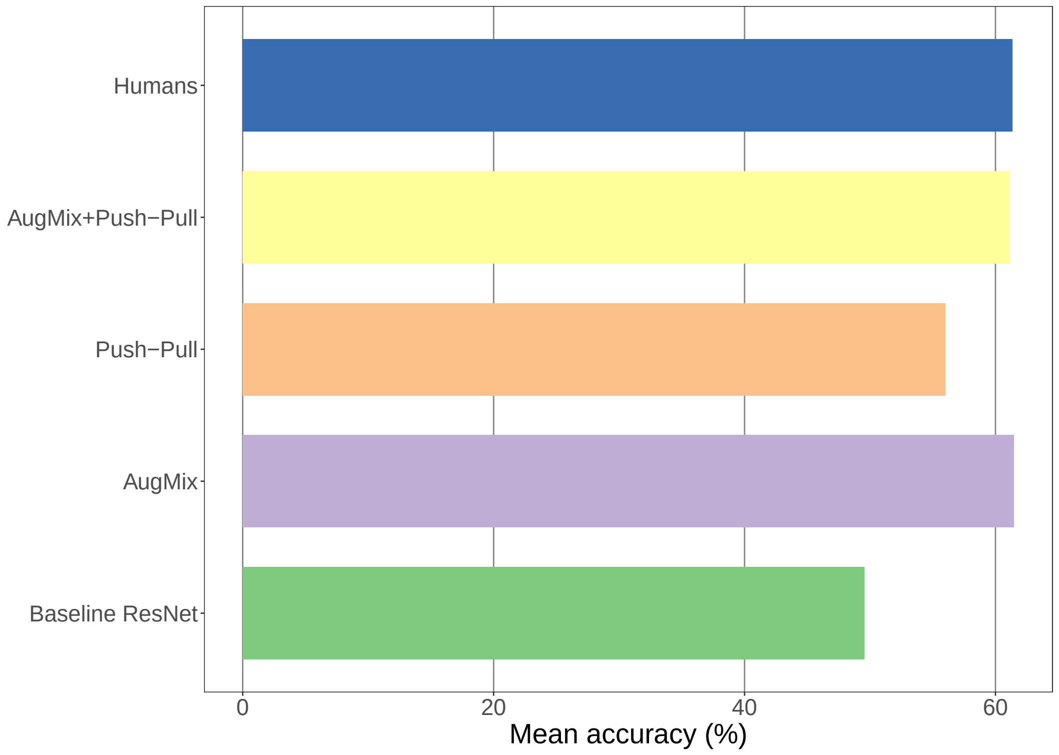

Figure 4.2: Mean corruption accuracy of DNNs with no pretraining and humans across all image distortion types. AugMix and AugMix+Push-Pull are on par with humans.

Figure 4.3 shows a comparison of the classification accuracy on individual distortions for the human subjects and the deep network models. In the case of non-pretrained networks, both AugMix and AugMix+Push-Pull have comparable performance to humans, even surpassing it in two cases: low-pass and uniform noise, with the AugMix+Push-Pull being the better performing network. It is fascinating to highlight these two use cases that show more robust performance of DNNs. Firstly, low-pass (or blurring) has real-world implications as it is highly likely for images to suffer from this as a result of low-quality camera sensors. Also, Geirhos et al. [8] findings showed that regular DNNs found it more difficult to generalize to uniform noise as compared to other distortions. Even though we do not have human data for salt-and-pepper noise, we notice a similar pattern on this distortion, with AugMix and AugMix+Push-Pull being the front runners. For all other image manipulations, humans continued to outperform DNNs. We also note that the Push-Pull layer significantly improves over the baseline ResNet for all corruptions except low-pass, where it is only slightly worse. It appeared to be more tolerant of image manipulations like rotation and greyscale.

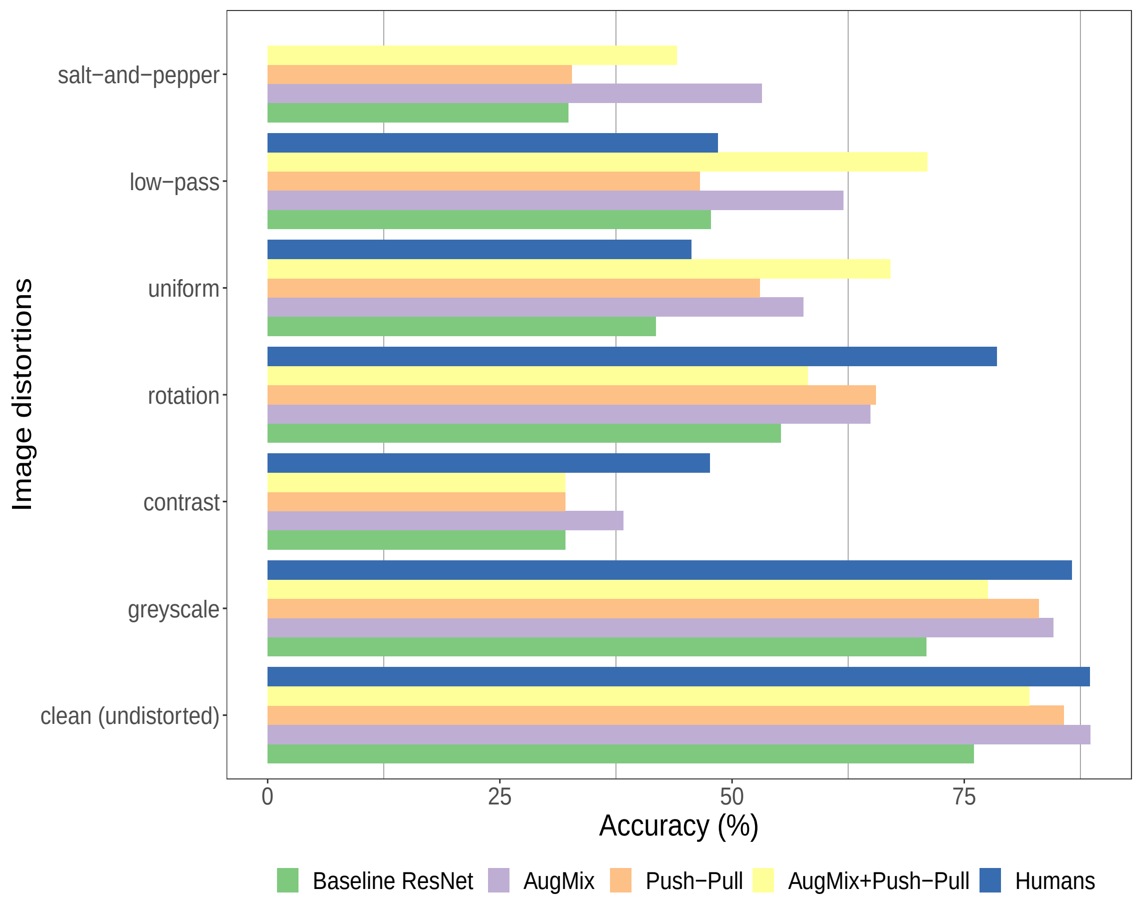

Figure 4.3: Classification accuracy of DNNs with no pretraining and humans on clean and distorted data. AugMix and AugMix+Push-Pull outperform humans on low-pass and uniform noise. Push-Pull performs better than baseline ResNet on all noises except low-pass.

The results suggest that the modified DNNs (AugMix, Push-Pull, AugMix+Push-Pull) are considerably more robust to corruptions than the baseline ResNet. The human visual system is considered highly robust to different types of distortions and noises. AugMix and AugMix+Push-Pull matching and, in some cases surpassing human performance is a crucial step towards robust machine learning and the creation of better models of human object recognition.

4.2.2 Pretrained networks

A similar pattern to the non-pretrained experiments emerges in terms of performance for pretrained DNNs, as shown in Figure 4.4. AugMix continues to be the best performing network on average. However, with pretraining, AugMix surpasses human performance by a large margin. AugMix+Push-Pull is the only other pretrained network to outperform humans and is only slightly worse than AugMix. We also notice an increased mean accuracy for the baseline ResNet upon pretraining. Looking at the errors made by the pretrained networks and humans in Figure 4.5 we notice that for pretraining, DNNs outperform humans on all the distortions. For instance, AugMix exhibits the best performance for all distortions except for low-pass and uniform noise. AugMix+Push-Pull beats every other network (and even humans) for these two distortions, which is consistent with what we saw for the non-pretrained architectures.

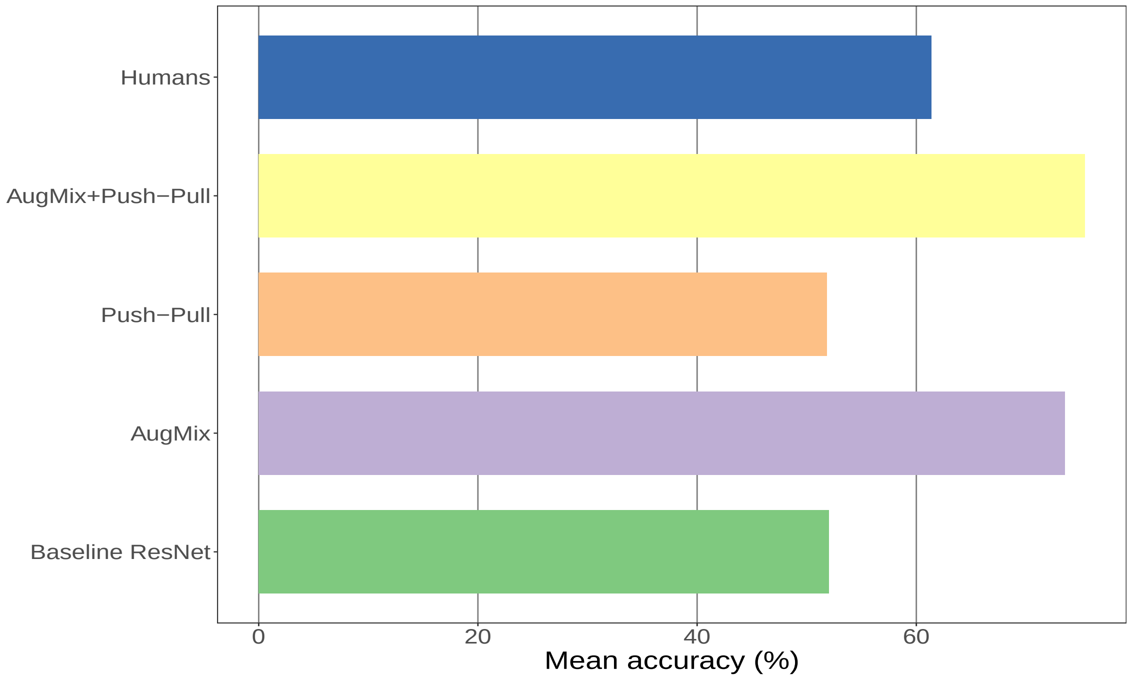

Figure 4.4: Mean corruption accuracy of pretrained DNNs and humans across all image distortion types. AugMix and AugMix+Push-Pull surpass human performance.

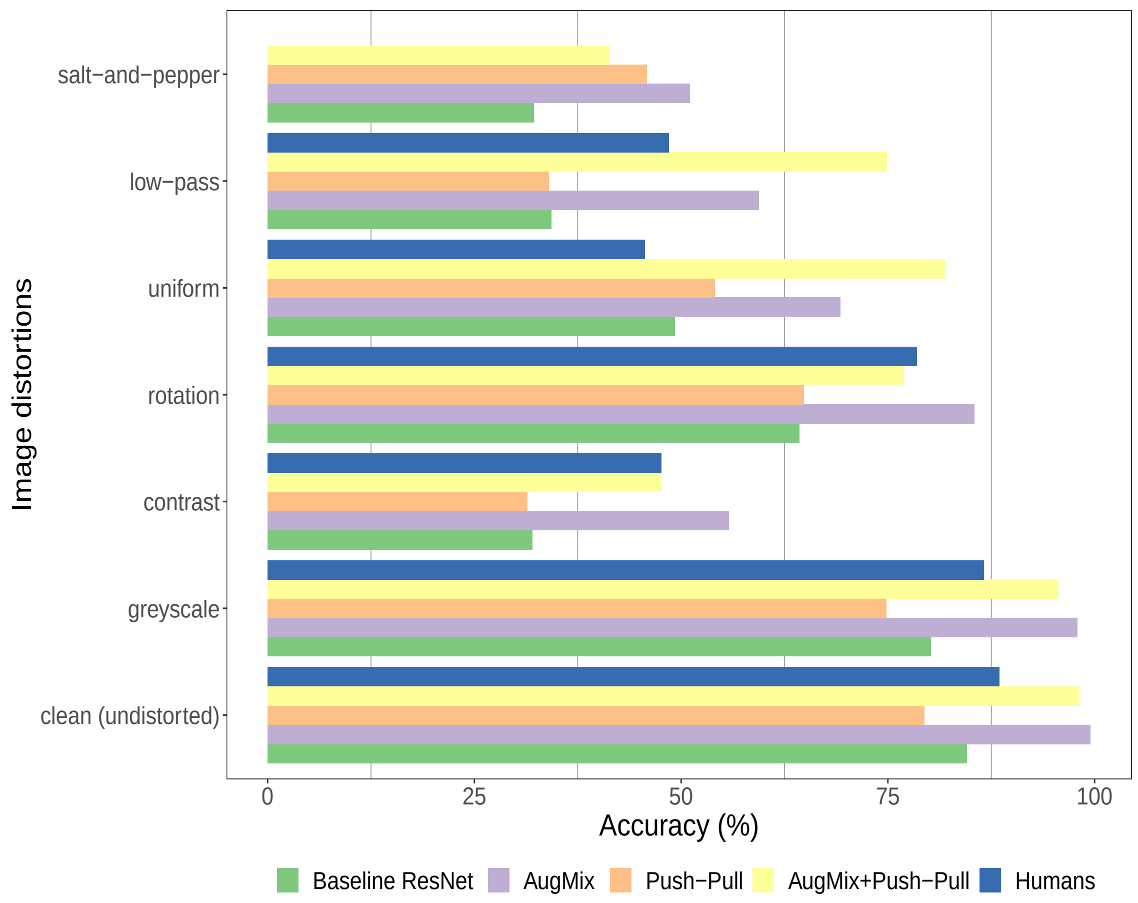

Figure 4.5: Classification accuracy of pretrained DNNs and humans on clean and distorted data. AugMix and AugMix+Push-Pull together outperform humans on all distortions. The pretrained Push-Pull performs worse than its non-pretrained counterpart.

The pretraining of the CNN layers contributes to an improvement over the non-pretrained counterparts for all networks except for the Push-Pull network. The pretrained Push-Pull network has roughly the same mean accuracy as baseline ResNet in this paradigm and even performs worse in case of greyscale distortion. In the original paper on the Push-Pull framework, Strisciuglio et al. [32] train all the networks from scratch. However, they suggest that one can replace the convolution layers of an already trained model with the Push–Pull layers. As mentioned in Section 3.5, the output from a Push-Pull layer is different from that of a usual convolutional layer. Strisciuglio et al. suggest that to combine the layers, we need to apply certain fine-tuning procedures in case of pretraining so that the layers after the Push-Pull layer can adjust to these new responses.

Based on our results, using a pretrained Push-Pull model does not seem feasible. In light of the fine-tuning recommendation made by the authors, we changed our default learning rate (0.1) and experimented with a smaller value (0.01). This gave us an increase in test accuracy on clean (undistorted) data from 69.1% to 79.3%. We conclude that there might be subtle intricacies in how the pretrained Push-Pull network should be fine-tuned. Deploying the Push–Pull layer changes the learning dynamics of these classification models, making the optimization process less straightforward. Since it is a novel layer, this problem may not be feasibly addressed by simply applying general rules of thumb for tuning. This is an interesting area for future research. We cannot justify the degradation in the performance suffered by the Push-Pull for the pretrained criterion. Hence, we compare with human performance only the non-pretrained DNNs in our study.

4.3 Comparing with human performance

Comparing the classification accuracies on different distortion types in Figure 4.6, we see that there is no consistent pattern in how humans and models behave. In some cases, DNNs surpass human performance, while in others they do not generalize so well.

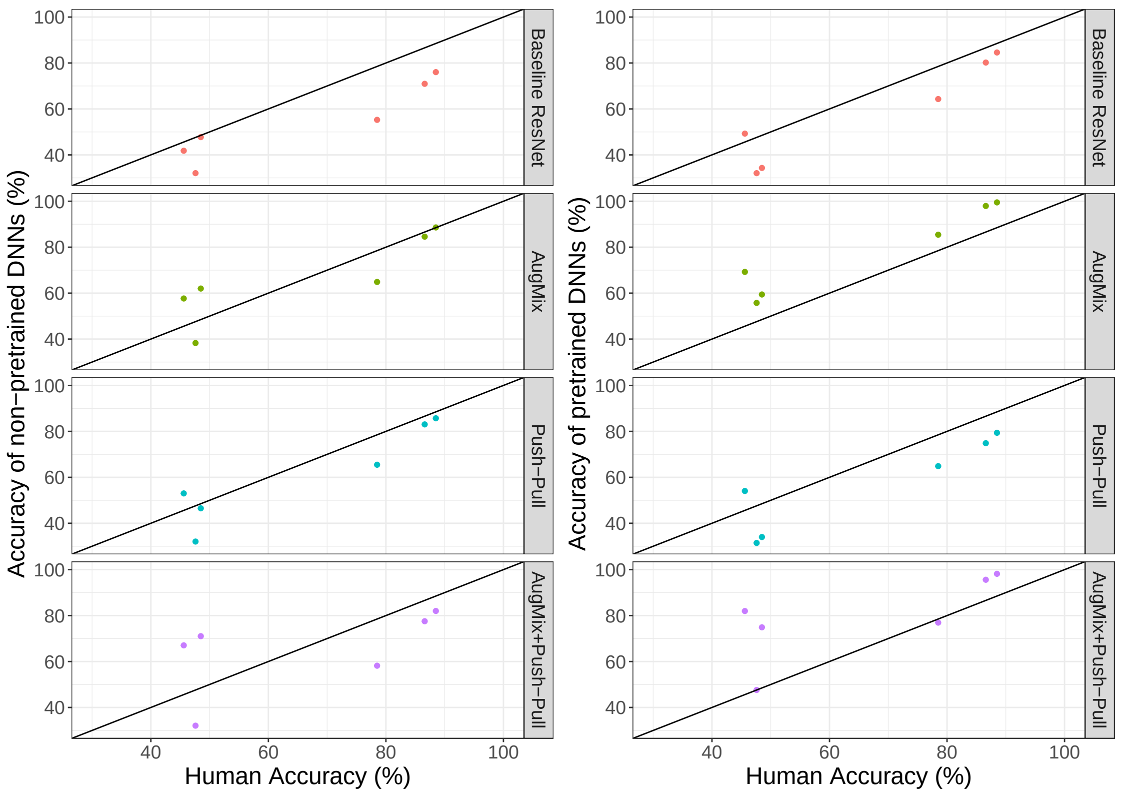

Figure 4.6: DNN versus human accuracy on different distortion types. AugMix and Push-Pull lie closer to the 45° line in both cases, suggesting resemblance to human performance.

Thus, we need more concrete measures to identify these underlying patterns and closeness in relationship. In order to find the most human-like DNN architecture, we will test for similarity between their performances in two ways:

- **Distance-based similarity.** One aspect of closeness can be defined as the smallest difference between human and DNN performance.
- **Pattern based similarity.** Another aspect of similarity can be operationalized by looking at patterns across the different noises between humans and a particular DNN.

4.3.1 Distance-based similarity measures

For distance-based similarity measures, we use t-tests to assess the similarity of each network with humans and sign tests to draw pairwise comparisons between networks.

t-tests for similarity with humans

Let us take the difference between the accuracy of a DNN, say a_i, for the i^{th} distortion with respect to humans h_i. Let d_{resnet}, d_{augmix}, $d_{\text{push-pull}}$ and $d_{\text{augmix+push-pull}}$ denote the difference for baseline ResNet, AugMix, Push-Pull and Augmix+Push-Pull respectively. For example, $d_{\text{resnet}} = \text{resnet}_i - h_i = (-12.476, -15.662, -15.555, -23.250, -3.797, -0.767)$, where each data point corresponds to one of the six distortions (including clean data) used in our study. Let $\bar{d}_{\text{resnet}}$ then be the mean difference between DNN and human accuracy across the different data points.

We make use of one sample t-tests to determine if there is a statistically significant difference between the mean accuracy of a DNN versus humans. For instance, our hypothesis for baseline ResNet is stated as:

$$H_o : \bar{d}_{\text{resnet}} = 0 \text{ against } H_a : \text{Not } H_o.$$

Here, our null hypothesis states that the mean accuracy of humans and DNN is the same, whereas our alternative hypothesis tests if they are different. We carry out these tests for each DNN in a similar fashion and obtain the results as shown below.

```
   One Sample t-test

data:  baseline_resent
t = -3.5079, df = 5, p-value = 0.01714
```

```
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -20.651084  -3.184582
sample estimates:
mean of x
-11.91783
```

```

  One Sample t-test

data:  augmix
t = 0.025873, df = 5, p-value = 0.9804
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -11.42531  11.65764
sample estimates:
mean of x
0.1161667
```

```

  One Sample t-test

data:  push_pull
t = -1.4494, df = 5, p-value = 0.2069
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -13.640585   3.804251
sample estimates:
mean of x
-4.918167
```

```

  One Sample t-test

data:  augmix_with_push_pull
t = -0.16281, df = 5, p-value = 0.877
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -20.79583  18.31849
sample estimates:
mean of x
-1.238667
```

Our preliminary analysis using a t-test shows that we only reject the null hypothesis for the baseline ResNet network. This implies that there is a statistically significant difference between how baseline ResNet performs and how humans perform. We do not reject the null hypothesis for AugMix, Push-Pull and AugMix+Push-Pull. Hence, there is no evidence that the performance of these networks on distorted data is not similar to human performance.

However, we would still like to find the architecture that most closely resembles the functioning of a human brain (failure to reject the null does now allow us to accept the null). Motivated by these results, we conduct a sign test to make pairwise comparisons across DNNs. In other words, we try to answer the question if a particular DNN is closer to humans when compared with another DNN. This will help us make more direct comparisons to find the most human-like architecture (i.e., the pairwise sign tests allow us to reformulate the null so that rejecting the null tells us which network is more human-like).

Sign tests for pairwise comparisons between networks

We have recorded the p-values for twelve sign tests corresponding to each combination as shown in Table 4.3.

Let us now discuss in detail the specifics of the sign test with respect to an arbitrary DNN-DNN pair, say AugMix compared with AugMix+Push-Pull. Let w_i be a variable denoting the closeness to human performance defined as:

$$w_i = \begin{cases} 1, & \text{if AugMix is closer to humans than AugMix+Push-Pull for the } i^{th} \text{ distortion} \\ 0, & \text{otherwise} \end{cases}$$

Now, w follows a Bernoulli distribution with probability of success p, where success is defined as AugMix being closer to humans than AugMix+Push-Pull. We make use of one sided (directional) sign tests to determine which network is closer. Our hypothesis for this test is stated as:

$$H_o : p = 0.5 \text{ against } H_a : p > 0.5$$

In other words, our null hypothesis says that both AugMix and AugMix+Push-Pull are equally likely to be closer to humans, whereas the alternative hypothesis tests if AugMix is closer. We carry out the tests for every DNN-DNN combination in a similar fashion and obtain the results as shown below in Table 4.3. Note that the diagonal elements will be empty cells.

Table 4.3: P-values obtained using sign test for every DNN-DNN pair.

	Baseline ResNet	**AugMix**	**Push-Pull**	**Augmix+Push-Pull**
Baseline ResNet		0.8906	0.8906	0.8906
AugMix	0.3437		0.6562	0.01563
Push-Pull	0.3437	0.6562		0.1094
Augmix+Push-Pull	0.3437	1	0.9844	

We get a significant p-value at nominal significance highlighted in blue, and we only reject the null hypothesis for the AugMix and AugMix+Push-Pull pair. The results suggest that AugMix is closer to human performance for OOD than AugMix+Push-Pull. This implies that adding the Push-Pull layer to AugMix is pulling it away from human-like performance rather than making it more brain-like.

4.3.2 Pattern-based similarity measures

In the case of pattern-based similarity, we use two measures. First, we rank the networks for each distortion depending on how close it is to humans. Secondly, we use Spearman correlation to look for monotonic patterns between DNN performance and that of humans.

Ranking closeness to humans

In this subsection, we are interested in looking for similarity in patterns between how a human brain perceives distortions and how a DNN does. For each distortion, we rank the networks based on the absolute distance from human accuracy. The closest DNN will have the least absolute difference between their accuracies. Since we have four networks under consideration, rank 1 is the closest to human performance and rank 4 being the furthest. Figure 4.7 is a stacked visual representation of these rankings across different image distortions. Based on this chart, we can see that AugMix has the highest proportion of 1's, indicating that it is closer to humans more often than other networks. It is also interesting to note that although Push-Pull, which is designed to mimic humans dealing with distortions, does come in second most of the time, if not first. The network AugMix+Push-Pull is rated second, third, and fourth in balanced proportions. The baseline ResNet does not make it to the top three for most distortions.

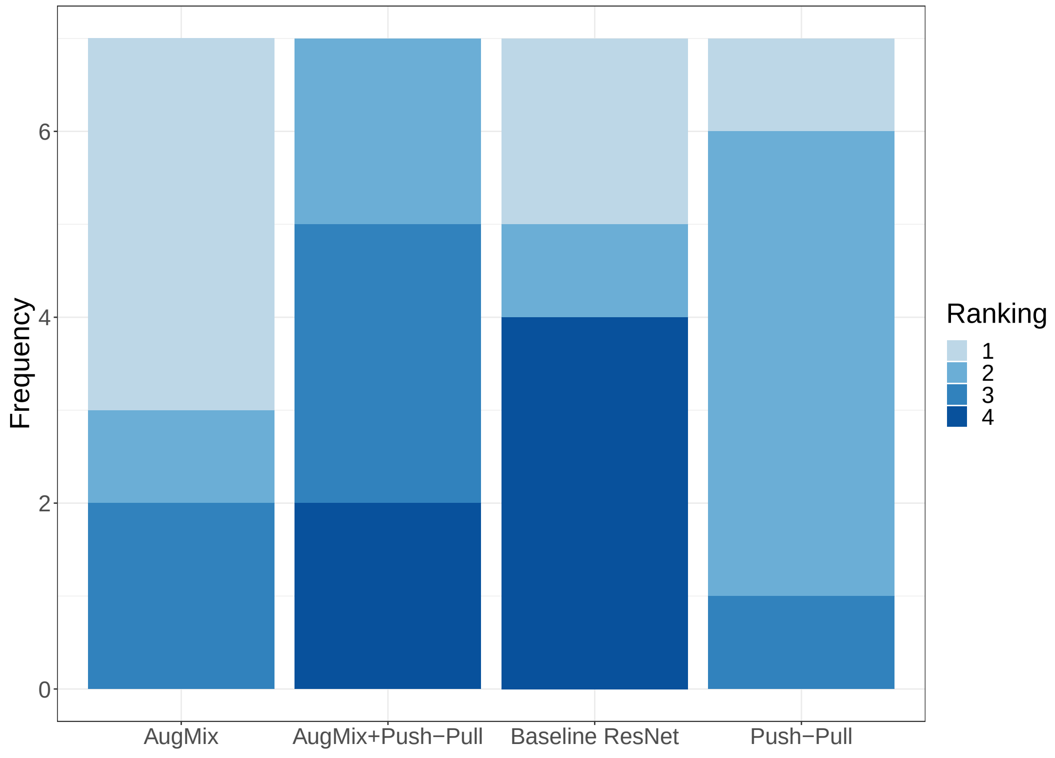

Figure 4.7: Ranking closeness of DNNs to humans. AugMix is the majority holder of rank 1. Push-Pull ranks second most of the time.

Spearman correlation

Earlier, we looked at the ranks of different DNNs based on how close they were to humans. Another lens through which we can test for patterns across distortions is through Spearman correlation. This measure tests how well the relationship between two variables can be characterized through a monotonic function. For each DNN, we analyze the correlation between their ranked accuracies and that of humans. Thus, this correlation will be high if both the DNN and humans exhibit similar performance patterns when we move from one distortion to the next. Similarly, we will get a low value when the two are not monotonically related. For instance, say human performance on uniform noise was ranked last. If AugMix's performance on uniform noise was also ranked last, we would expect AugMix and humans to be strongly correlated.

We find that baseline ResNet and AugMix have a high correlation value as depicted by Figure 4.8. Based on our previous distance-based similarity measures, this result is consistent in showing that AugMix is the most human-like architecture. However, it is interesting to note that at least in terms of behavorial patterns across different distortions, baseline ResNet mimics some aspects of human performance.

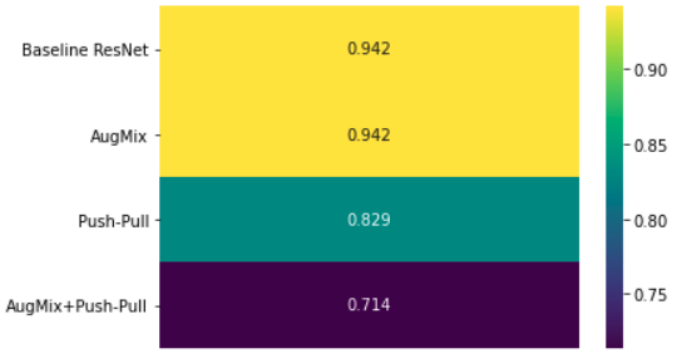

Figure 4.8: Spearman correlation between DNNs and human performance. AugMix and baseline ResNet have the highest correlation values indicating strong monotonic relationships with humans.

As we end this chapter, we observe that the above results suggest that AugMix is the most human-like architecture compared to the other DNNs under consideration. These findings do not support our initial hypotheses that a Push-Pull paradigm for OOD would be the most brain-like network as it is designed to mimic the functions of the human brain. Instead, we find that AugMix is closest and that combining it with Push-Pull layers (AugMix+Push-Pull) actually pulls the network away from human-like performance. It is possible that there could be some underlying representational capacity present in the human brain that is lacking in Push-Pull. Conversely, this could indicate that some of the internal mechanisms of the AugMix network are more similar to the human visual system. These findings can be used to advance our understanding and guide future research into more robust machine learning models. It may be useful to derive inspiration from the human visual system to build better models of object recognition that achieve state-of-the-art performance even on distorted data, but despite Push-Pull's inspiration from the V1, we find evidence that this inspiration is not translated into human-like behaviour.

Chapter 5

Conclusions and Future Work

DNNs are garnering more and more attention due to their state-of-the-art precision, which surpasses human performance in many areas. In this work, we were interested in the paradigm wherein these models encounter data they have not seen during training, termed as out-of-distribution data. More specifically, we focused on image recognition tasks with a wide variety of applications in medicine, self-driving cars, social media networks, etc. We compared object recognition by DNNs with how humans recognize objects in an image. Due to its inherent ability to withstand noisy settings, the human visual system possesses desirable properties that can be leveraged for image recognition tasks. To achieve human-like artificial intelligence, we would like to deploy a real-world model that can mimic these generalization capabilities of the human brain, or at least come close to it. This sets in motion the search for tools needed to investigate this empirically.

To that end, we explored deep neural networks that are designed to be robust to out-of-distribution and distorted data and compared them with human performance. These methods are motivated by a need to adapt and overcome limitations of existing DNNs, for instance, ResNet-50. We approached this problem in two ways. The first way explored an external method using a regular DNN coupled with a stochastic data augmentation technique called AugMix. The other way looked at making internal modifications to the regular DNN architecture. Stepping back into early neuroscience models naturally gave way to the bedrock of the methodology behind the Push-Pull layer. The alterations made by the Push-Pull layer are inspired by the functioning of the human brain (V1) and were therefore hypothesized to be more human-like. We also considered a blend of this intrinsic and extrinsic approach by combining AugMix and Push-Pull into a novel network. We considered six different types of distortions for our analysis to cover a wide range of practical scenarios. Furthermore, we conducted our experiments on a subset of the ImageNet dataset, a popular benchmark in object recognition. We used statistically rigorous testing procedures to draw comparisons with humans as opposed to standard error-based rules of thumb [8].

The purpose of this research is to offer insights into two broad questions: What is the most human-like DNN architecture? Our results demonstrated that AugMix tends to be functionally similar to the human visual system, as measured by classification accuracy. This was an interesting result as it also beat the Push-Pull network, which is designed to capture the essence of the human brain when faced with noisy images. We validated our findings by looking at this similarity through various lenses and metrics like hypothesis testing and ranked correlation. We found that contrary to our intuition, pretraining actually worsened the performance of the Push-Pull network. We could only provide our own intuitive explanations for this contradiction, and thus it requires further investigation.

The second question we tackled was: How do we advance our understanding of the analogy between deep learning networks and the human brain in the pursuit of building better models of human object recognition? Identifying DNNs that best emulate the mechanisms of the brain becomes the current best understanding of how the brain functions and lays the foundation for future experiments.

In the ever-expanding literature of robust machine inference and cognitive sciences, human-inspired models of deep learning have carved out a fairly prominent niche—particularly for object recognition tasks. We summarize a few areas of future work here:

1. We carry out our experiments for a single test/train split. An immediate extension would be to repeat it for multiple such splits. This would give us the flexibility to access more metrics that test similarity with humans.
2. We found that pretraining for the Push-Pull network requires further attention to the fine-tuning procedures. Future work around this could lead to improved performance and comparison between humans and pretrained networks included in this study.
3. For our experiments, we considered only one paradigm from Geirhos et al. [8] where we train on clean (undistorted) data only and test on the different types of distortions. It would be interesting to simulate the other scenarios using our current set of DNNs. In other words, what happens when customized networks like AugMix and Push-Pull are trained on one or more distortions and then tested on other noises? Does this translate to improved performance on unseen distortions for these custom networks as compared to regular DNNs? Does it make the networks more or less human-like?
4. We explored four types of DNN architectures to handle OOD. However, this is a mere drop in an ocean full of artificial neural networks. A future endeavour useful for researchers would be to add more baseline and customized DNNs to expand our

comparative study and benchmark human performance. There exist several other models like Stylized ImageNet [7] that use shape-based feature representations for neural network learning. There is also DeepAugment [14] which is a data augmentation technique that perturbs images by introducing distortions during forward propagation. This is different from AugMix (AugMix applies simple augmentation operations only to the raw images themselves).

5. We only explored image data. Future work could include behavioral studies in which the interaction between humans and their environment is compared to reinforcement learning.

Our comparative study examines human-like performance in four DNNs (Push-Pull, AugMix, ResNet and AugMix+Push-Pull). However, there are many more avenues to go down along the road to artificial intelligence, making this a fascinating and growing field to explore.

Bibliography

[1] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological review*, 94(2), 1987.

[2] F. X. Diebold. “Big Data” and its Origins. *arXiv preprint 2008.05835*, 2021.

[3] S. Dodge and L. Karam. Understanding how image quality affects deep neural networks. *International Conference on Quality of Multimedia Experience (QoMEX)*, 2015.

[4] S. Dodge and L. Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–7, 2017.

[5] T. C. B. Freeman, S. Durand, D. C. Kiper, and M. Carandini. Suppression without inhibition in visual cortex. *Neuron*, 35(4), 2002.

[6] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[7] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.

[8] R. Geirhos, C. R. M. Temme, J. Rauber, H. H. Schütt, M. Bethge, and F. A. Wichmann. Generalisation in humans and deep neural networks. In *Proceedings of the 31st Conference on Advances in Neural Information Processing Systems*, 2018.

[9] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[10] A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. *arXiv preprint 1303.5778*, 2013.

[11] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3), 2020.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[14] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, D. Song, J. Steinhardt, and J. Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv 2006.16241*, 2021.

[15] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint 1610.02136*, 2018.

[16] D. Hendrycks, K. Lee, and M. Mazeika. Using pre-training can improve model robustness and uncertainty. *arXiv preprint 1901.09960*, 2019.

[17] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. AugMix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint 1912.02781*, 2020.

[18] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. *Computer Vision and Pattern Recognition (CVPR)*, page 2261–2269, 2018.

[19] E. R. Kandel, J. H. Schwartz, and T. M. Jessell. *Principles of neural science.* New York: McGraw-Hill, Health Professions Division, 2000.

[20] J. Koenderink, M. Valsecchi, A. van Doorn, J. Wagemans, and K. Gegenfurtner. Eidolons: Novel stimuli for vision research. *Journal of Vision*, 17(2), 2013.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th Conference on Advances in Neural Information Processing Systems*, 2012.

[22] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *arXiv preprint 1604.00289*, 2016.

[23] V. AF Lamme, H. Supèr, and H. Spekreijse. Feedforward, horizontal, and feedback processing in the visual cortex. *Current Opinion in Neurobiology*, 8(4), 1998.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[25] G. A. Miller. WordNet: A lexical database for English. *Communications of the Association for Computing Machinery*, 38(11), 1995.

[26] R. T. Pramod and S. P. Arun. Do computational models differ systematically from human object perception? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1601–1609, 2016.

[27] A. Rosenfeld, M. D. Solbach, and J. K. Tsotsos. Totally looks like - how humans compare, compared to machines. *arXiv preprint 1803.01485*, 2018.

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3), 2015.

[29] M. Schrimpf, J. Kubilius, H. Hong, N. J. Majaj, R. Rajalingham, E. B. Issa, K. Kar, P. Bashivan, J. Prescott-Roy, F. Geiger, K. Schmidt, D. L. K. Yamins, and J. J. DiCarlo. Brain-score: Which artificial neural network for object recognition is most brain-like? *bioRxiv preprint 10.1101/407007v2*, 2020.

[30] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587), 2016.

[31] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015.

[32] N. Strisciuglio, M. Lopez-Antequera, and N. Petkov. Enhanced robustness of convolutional networks with a push–pull inhibition layer. *Neural Computing and Applications*, 32(24), 2020.

[33] M. M. Taylor, M. Sedigh-Sarvestani, L. Vigeland, L. A. Palmer, and D. Contreras. Inhibition in simple cell receptive fields is broad and OFF-subregion biased. *Journal of Neuroscience*, 38(3), 2018.

[34] S. Yun, D. Han, S. Joon Oh, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *arXiv preprint 1905.04899*, 2019.

[35] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the robustness of deep neural networks via stability training. *arXiv preprint 1604.04326*, 2016.