Learning Facet-Specific Entity Embeddings

A thesis submitted in partial fulfilment

of the requirement for the degree of Doctor of Philosophy

Rana Abdulhadi Alshaikh

April 2021

Cardiff University
School of Computer Science & Informatics

To the soul of my father,

To my mother, my husband, and my children.

Abstract

An entity embedding is a vector space representation of entities in which similar entities have similar representations. However, similarity is a multi-faceted notion; for example, a person may be similar to one group of people because they graduated from the same university and similar to another group through having the same nationality or playing the same sport. Our hypothesis in this thesis is that learning a single entity embedding is a sub-optimal way to faithfully capture these different facets of similarity. Therefore, this thesis aims to learn facet-specific entity embeddings that capture different facets of similarity, taking inspiration from a framework widely known in cognitive science called *conceptual spaces* framework.

Conceptual spaces [48] are vector space models designed to represent entities of a given kind (e.g. movies), together with their associated properties (e.g. scary), and concepts (e.g. thrillers). As such, they are similar in spirit to the vector space models that have been proposed in natural language processing, but there are also notable differences. First, the dimensions of conceptual spaces, referred to as *quality dimensions*, are interpretable, as they correspond to semantically meaningful features. Second, conceptual spaces are organized into sets of semantic domains or facets (e.g. genre, language), which are formed by grouping the quality dimensions. Each facet is associated with its own low-dimensional vector space, which intuitively captures similarity with respect to the corresponding facet. For instance, the vector space for the budget facet would only capture whether two movies had similar budgets. From an application point of view, the fact that conceptual spaces are structured into facets is appealing because this

allows us to model the different facets of similarity in a more flexible and cognitively more plausible way. Based on this, we hypothesize that learning facet-specific entity embeddings that are similar in spirit to conceptual spaces will allow us to predict the properties and categories of entities more reliably than from standard single space representations. Learning data-driven conceptual spaces, especially in an unsupervised way, has received very limited attention to date.

Therefore, in this thesis, we will learn facet-specific entity embeddings that is similar in spirit to conceptual spaces. This includes learning quality dimensions and then grouping them into facets. In particular, in this thesis, we propose three unsupervised models to learn this type of vector space representations for a set of entities using their textual descriptions. In two of these models, we convert traditional vector space embeddings into facet-specific entity embeddings, using quality dimensions-like features. In these cases, we rely on an existing method to learn these features. In our first proposed model, we structured the vector space representations implicitly into meaningful facets by identifying the quality dimensions in a two-level hierarchy: The first level corresponds to the facets, and the second level corresponds to the facet-specific features. In our second developed model, using the quality dimensions and pre-trained word embeddings, we decompose the vector space representations into low-dimensional facets in an incremental way. In both of these models, we depend on clustering algorithms to find facet-specific features. In contrast, our third proposed model uses a mixture-of-experts formulation to find the features that describe each facet and it simultaneously learns the facet-specific embeddings directly from the bag-of-words.

We evaluate our models on several datasets, each of which contains a set of entities with their textual descriptions and a number of classification tasks, using a range of different classifiers. The experimental results support our hypothesis that, by capturing different facets of similarity, facet-specific vector space representations improve a model's ability to predict the categories and properties of entities.

Acknowledgements

First and foremost, I thank God, who gave me the patience and strength needed to write this thesis. I would also like to thank the people whose presence and support enabled me to complete this journey.

I am indebted to my supervisor, Professor Steven Schockaert, for his unlimited support, guidance, and encouragement. During the past four years, he could not have been more generous with his time, his knowledgeable advice, and his thought-provoking questions, which contributed to the development of my skills and overcame my PhD difficulties. I would also like to thank my second supervisor, Dr.Zied Bouraoui, who assisted me and was always available when I needed his support.

I extend my heartfelt appreciation to my family and friends: to my husband, who stood beside me, endured, and sacrificed much so that I could finish my PhD; to my children, Naif and Khalid, for their tremendous support and prayers, which helped make this journey possible. To my father's soul, to my mother for her continuous prayers and encouragement, and to my sisters and brothers for their support. To my friends and colleagues in Saudi Arabia and the UK for being a source of positive energy and encouragement.

Finally, I would like to express my sincere gratitude to the Ministry of Education, King Abdulaziz University, and the Saudi Arabian Cultural Bureau in London for the generous scholarship that provided my family and me with unlimited support and enabled me to devote myself to this research.

Contents

List of Publications

The work introduced in this thesis is based on the following publication:

1. Rana Alshaikh, Zied Bouraoui, and Steven Schockaert. Learning conceptual spaces with disentangled facets. In Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), pages 131–139, Hong Kong, China, Nov2019. Association for Computational Linguistics. [6]

2. Rana Alshaikh, Zied Bouraoui, and Steven Schockaert. Hierarchical linear disentanglement of data-driven conceptual spaces. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI), pages 3573–3579, Yokohama, Japan (Online), Jul 2020. IJCAI.org. [8]

3. Rana Alshaikh, Zied Bouraoui, Shelan Jeawak, and Steven Schockaert. A mixture-of-experts model for learning multi-facet entity embeddings. In Proceedings of the 28th International Conference on Computational Linguistics (COLING), pages 5124–5135, Barcelona, Spain (Online), Dec 2020. International Committee on Computational Linguistics. [7]

List of Figures

List of Tables

List of Acronyms

BOW Bag-Of-Words

CBOW Continuous Bag-of-Words

GloVe Global Vectors for Word Representation

NLP Natural Language Processing

ML Machine Learning

TP True Positive

TN True Negative

FP False Positive

FN False Negative

SVM Support Vector Machines

kNN k-Nearest Neighbors

Chapter 1

Introduction

1.1 Background and Motivation

In our modern and complex world, the volume of knowledge is multiplying rapidly. Thousands of entities (i.e. identifiable things with separate existence, such as movies, restaurants, books, organisations, buildings, events, and many more), are created every day. For most of these entities, there is a mass of information in the form of picture, text, audio, and video content. This information comes, among others, from the contributions of many users through social networks, online encyclopedias, blogs, shopping sites, and other sources. All this information needs to be systematized and represented in a way that helps computer systems to find patterns, extract valuable information, categorize, reason, and much more. Representational learning for entities, which aims to do exactly this, is therefore an active area in the field of artificial intelligence (AI). One of the techniques used, known as entity embedding, embeds entities in a continuous vector space. This type of entity representation plays a central role in the fields of information retrieval [34], natural language processing [105] and machine learning [113], cognitive science [48], and others. Entity embeddings are learned using a variety of different inputs, including human similarity judgments, text descriptions, and images. Regardless of how they are learned, entity embeddings can essentially be viewed as compact encodings of a similarity relation. Indeed, while many embeddings exhibit a variety of interesting linear regularities, such regularities are the result of the structure of the similarity relation that is used for learning the embedding [5].

Similarity is inherently multi-faceted, with the importance of the different facets being context-dependent. For instance, two movies can be similar because they belong to the same genre or because they are about the same historical event, among many others. However, these different facets of similarity are not reflected in the structure of standard entity embeddings; at least, not in an explicit way. To see why this is sub-optimal, consider the problem of concept induction: given a small set of entities $e_1, ..., e_k$, identify other entities that are of the same kind. For instance, given the examples *Barcelona*, *Madrid*, *Alicante*, valid completions would be other Spanish cities. The problem of concept induction underpins many of the applications in which entity embeddings are used, including knowledge base completion and recommendation. In cases where the given set of entities is small, the result will strongly depend on the similarity relation encoded by the given entity embedding: given a few entities, we can do little else than selecting the nearest neighbors of their (averaged) entity vectors. However, if we have several embeddings of the considered entities, each capturing different facets, then we can solve the concept induction task by first identifying the most relevant facet(s), and thus rely on a form of similarity that is relevant for the given concept. Another task that could benefit from such a representation is modelling the semantics of web tables, which has been getting quite some attention lately. Web tables have valuable information but making sense of such information means that we need to understand the meaning of each row and each column, which we need to learn from few examples.

Another important advantage of representing the different facets of similarity is that this allows for more fine-grained explanations in unsupervised settings. In particular, it can be said that two entities are similar in a specific set of features if they are similar in the facets that encode those features, which is more informative than a statement about the general similarity induced from traditional entity embedding. For example, if there are two buildings that have high cosine similarity in the architectural style facet and low similarity in the location facet, it is easily be induced that the appearance of the buildings is very similar, but they are in different locations. However, inducing such

information is not possible if there is only one vector space representation. Although in this thesis the aim is to improve the interpretability of static vectors representations, in recent years there has been a shift from static vectors representation to contextualize language models such as Bidirectional Encoder Representations from Transformers (BERT) [38], which dynamically learn representations of entities in context. This shift has made the representations used in NLP even less interpretable. This means that the aim of this work is still valid, and understanding what features language models are using to make predictions is, as a result, rapidly gaining in importance, even though the techniques used in this thesis will need to be adapted to work with language models.

The problem of learning separate facet-specific embeddings is related to disentangled representation learning. While this latter problem has already received considerable attention, most existing work has focused on (semi)-supervised settings, primarily in the visual domain. Unsupervised approaches for disentangled representation learning generally need strong inductive biases [94]. In the text domain, most work has focused on separating style or sentiment from content.

The idea that a single unstructured vector space is insufficient for modeling similarity has been widely studied in cognitive science. In particular, this idea is closely related to the distinction between so-called *integral* and *separable* dimensions, which plays a central role in cognitive models of categorization [48]. Dimensions, in this context, refer to elementary cognitive features and are known as quality dimensions. This stands in stark contrast to most methods for learning entity embeddings, which only aim to capture similarity, where the dimensions do not typically have any particular meaning. Two dimensions are intuitively separable if they can be considered in isolation (e.g. *size* and *hue*), and integral otherwise (e.g. *hue*, *saturation*, and *luminosity*, which are jointly perceived as colour). Psychological studies have shown that the way in which humans generalize from examples is affected by the nature of the underlying dimensions [51, 114]. The theory of conceptual spaces [48] is a popular cognitive model

that takes this distinction between integral and separable dimensions into account by organizing dimensions into facets [1]. Dimensions from the same facet are assumed to be integral (i.e describe the same aspect of similarity) whereas those from different facets are assumed to be separable. Each facet is associated with a metric space. Given a conceptual space, the dissimilarity between two objects is determined (i) by computing their (usually Euclidean) distance in each of the facet-specific spaces, and (ii) by taking a weighted average of these distances. This is in accordance with empirical findings, which suggest that Euclidean distance is predictive of human similarity judgments in the case of integral dimensions, whereas such judgments are a function of Manhattan distance in the case of separable dimensions.

The problem of learning conceptual spaces from data has only received limited attention. Inspired by conceptual spaces, [36] introduces a method for structuring a given entity embedding using interpretable (but non-orthogonal) dimensions. Somewhat related, [119] propose a supervised method to decompose word embeddings into subspaces that capture particular aspects of word meaning, such as sentiment polarity or part-of-speech. The idea of decomposing a word embedding space into subspaces is also central to the method from [4], which is aimed at distinguishing synonyms from antonyms. Within a broader context, [10] propose a method to group numerical features into facets, with the aim of generating better linguistic descriptions of numerical data. Based on all these researches, we think the conceptual space framework can address the aforementioned limitations of entity embeddings.

Therefore, in this thesis, we consider the problem of learning entity embeddings that are more similar in spirit to conceptual spaces, in an unsupervised way, for a set of entities from a certain domain where each entity is associated with textual description. To illustrate, if we have a set of buildings where each building has a textual description, we expect that the descriptions include information about different facets (e.g. location, architectural style) the purpose of the building, and many more. Each facet captures a

[1] These facets are often referred to as *domains* in the context of conceptual spaces. However, we will use the term facets to avoid confusion with domains of discourse.

different facet of similarity; for example, Cardiff Castle and the Millennium Stadium could be similar because they are both in Cardiff, but dissimilar in terms of their architectural style. We can think of conceptual spaces as being structured in two levels. At the top level, a conceptual space is organized into facets, such as architectural style and geographic locations of buildings. At the second level, each facet is organized using a number of features that are specific to that facet, such as *modern* and *historical* for the architectural style facet. Our aim in this research is to identify such facets and represent them as low-dimensional vector spaces. In particular, there are two important properties of conceptual spaces that we want the learned embedding to adopt: 1) the quality dimensions of conceptual spaces represent the important semantic features of the domain, and 2) these quality dimensions are grouped into facets, where each facet is associated with a low-dimensional vector space. Based on this view, the process of learning conceptual spaces from data can be seen as consisting of the following steps:

1. We need to identify the salient semantic features for the given domain.
2. We need to group these features into facets.
3. We need to learn a low-dimensional vector representation of the entities of interest for each given facet.
4. We need to model the semantic features in the corresponding facet-specific vector space.

Learning such representations is challenging due to the fact that we do not have any prior knowledge about what facets are covered in the text descriptions of the entities, and what are the semantic features that define a particular facet.

To address these challenges, we propose a number of different approaches for learning facet-specific representations. Our proposed approaches range from obtaining facet-specific representations by converting a given traditional entity embedding (using an existing method for learning quality dimensions) to learning facet-specific representations directly from the data. To simplify the problem, as a starting point we used the method from Derrac and Schockaert [36] to learn interpretable dimensions that are similar to quality dimensions of the conceptual space (i.e. Step 1). This method assumes no prior knowledge of the salient features of the considered domain. Thus, each term w that appears sufficiently in the bag-of-word (BoW) representation is considered to be a candidate salient feature of the domain. Then, in a given entity embedding space, Derrac and Schockaert [36] train for each word w a linear classifier, which tries to predict from the embedding of an entity whether the word w occurs in its description. The words $w_1, ..., w_n$ for which this classifier performs sufficiently well are then used as basic features. Each of the basic features w is associated with a corresponding vector $\mathbf{d_w}$ (i.e. the normal vector of the separating hyperplane that is learned by the classifier). The learned vectors will be referred to as *feature directions* as each direction can be used as a scale to rank the entities based on how much they have the corresponding feature. Figure 1.1 shows a two-dimensional representations of the movies domain, spanned by two feature directions. We rely on this method to learn quality-dimensions-like directions in two of our three proposed approaches.

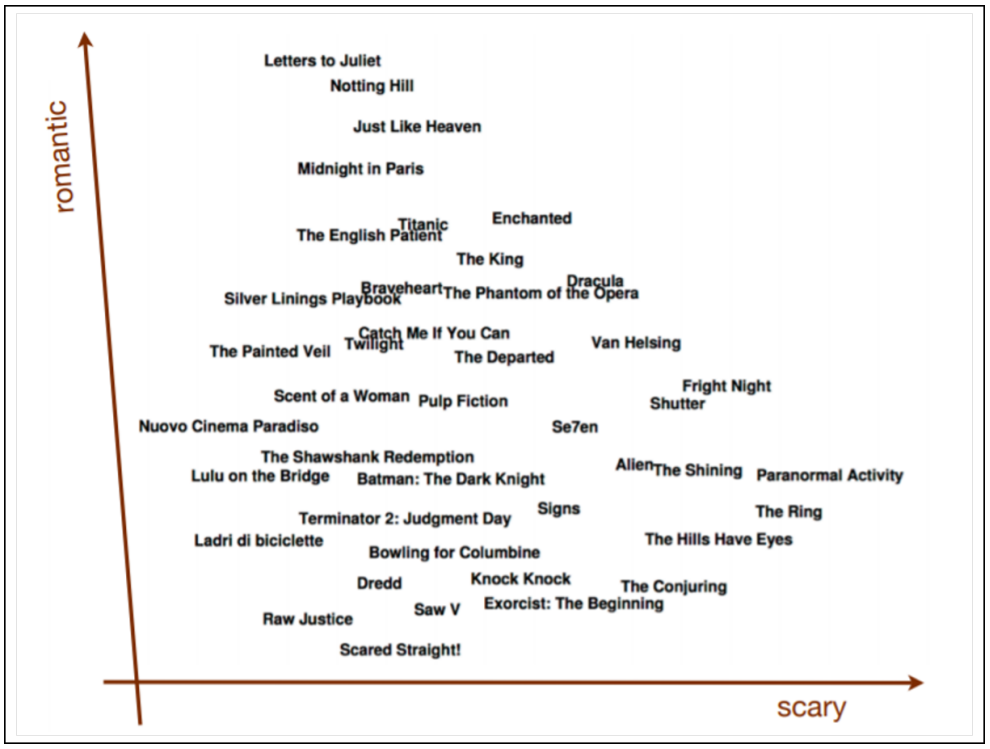


Figure 1.1: Directions modeling two salient features of the movies domain, *scary* and *romantic*, in a two-dimensional projection of a 100-dimensional space [124].

1.2 Hypothesis and Research Questions

The initial vector space embeddings generated by traditional methods are not interpretable, lack clear structure, and do not reflect the different facets of similarity in their structure. Previous research, especially that inspired by the conceptual spaces framework, has shown that it is possible to learn a semantic representation of a vector space by learning interpretable dimensions (i.e. to identify semantic features as directions in the vector space). We hypothesize: 1) that by adapting the conceptual spaces framework and structuring the entity embedding space, either explicitly or implicitly, into separate facets, the quality of these learned semantic features will improve; also, 2) that by having multiple low-dimensional space representations of entities, where each

one specializes in one of the facets (i.e. facet-specific representations), the properties and categories of entities can be predicted more reliably than from standard representations.

In order to verify this hypothesis, the following research questions are addressed:

Research Question 1: *For a given entity embedding, can we discover a variety of meaningful facets by clustering the learned feature directions, or is an external supervision signal required for this purpose?*

Research Question 2: *Since the conceptual space structure can be viewed as a hierarchical one, can we structure, i.e. implicitly decompose, the given vector space into facets by identifying the features directions in a hierarchical fashion?*

Research Question 3: *How, and to what extent, can we, in an unsupervised way, explicitly decompose a given vector space embedding into low-dimensional vector spaces that capture meaningful facets?*

Research Question 4: *To what extent can a higher-quality representation be obtained by directly learning facet-specific spaces from BoW representations, rather than by decomposing an existing vector space?*

1.3 Contributions

The primary aim of this thesis is to learn facet-specific low-dimensional entity embeddings. The contributions made through this thesis are:

- Proposing a simple and effective method for dividing the entity embedding implicitly into separate facets which are based on hierarchically learning quality dimensions. We show that this model learns higher quality dimensions than the standard approach proposed by [36]. Furthermore, we present evidence that the

linear regularities that can be found in entity embeddings tend to have a hierarchical structure. The hierarchical approach and its relevant analysis were published at IJCAI 2020 [8]. (This contribution relates to Research Questions 1 and 2.)

- Introducing a model to decompose a vector space embedding into meaningful facets in an unsupervised fashion. The model uses pre-trained word embedding as an inductive bias, which allows us to find non-thematic facets. The incremental nature of the model reduces the impact of the dominant facet, i.e. finds diverse facets, rather than redundant facets that are mostly related to the most dominant facet. The model was published at CoNLL 2019 [6]. (This contribution relates to Research Question 3.)

- Developing a scalable and unsupervised method for learning several facet-specific low-dimensional entity embeddings from BoW representations. This model combines the mixture of experts (MoE) model with GloVe model in order to learn multiple low-dimensional entity embeddings, each of which specializes in one facet of the data. The model was published at COLING 2020 [7]. (This contribution relates to Research Question 4.)

1.4 Thesis Structure

The remainder of this thesis is organized as follows:

- Chapter 2 – Background and Related Work – provides an in-depth review of the literature of entity embeddings, conceptual spaces, and some of the other related areas. Additionally, this chapter explains and defines some of the fundamental and basic methods in machine learning that we used in this research.

- Chapter 3 – Data Acquisition and Preprocessing – presents the datasets that we used to evaluate our proposed methods. Information regarding how these datasets were obtained and preprocessed, the number of entities in each dataset,

the size of the vocabulary, the classification tasks, how we represent each dataset, and how we learn the quality dimensions-like directions are covered in this chapter. This chapter presents an explanation of the method used by Derrac and Schockaert in [36], which is the model that represents the basis of our work.

- Chapter 4 – Hierarchical Linear Disentanglement of Data-Driven Conceptual Spaces – describes a model that tries to mitigate the limitations of learning quality dimensions in single-space representations by identifying features in a hierarchical fashion. A full description of the evaluation of the model in comparison with several baselines is also introduced in this chapter.

- Chapter 5 – Learning Conceptual Spaces with Disentangled Facets – introduces an in-depth analysis of several methods proposed to decompose a given vector space embedding into meaningful facets in an unsupervised fashion. This analysis includes the quantitative evaluation, which is a set of experiments that uses several classifiers to investigate the quality of the resulting facet-specific embeddings produced by these methods, and an analysis of the qualitative aspect.

- Chapter 6 – A Mixture-of-Experts Model for Learning Multi-Facet Entity Embeddings – describes the model that learns facet-specific embeddings using a mixture-of-experts formulation and a variant of GloVe. This chapter presents the detailed experiments conducted to evaluate the performance of the model, quantitatively and qualitatively, in comparison with several baselines.

- Chapter 7 – Conclusion and Future Work – concludes the thesis by expounding on how the models in the previous chapters fulfill the general research aims and contribute to the area of knowledge. This chapter additionally highlights potential future work.

1.5 Summary

In this chapter, we have introduced our considered problem, which is learning a facet-specific representations that are similar to conceptual spaces. We have also discussed the motivation behind the choice of this topic, with a brief overview of the background. Additionally, we have discussed our hypothesis, research questions, contributions, and the thesis structure. The next chapter will focus on the literature of the considered topic with an explanation of some basic machine learning algorithms used in this research.

Chapter 2

Background and Related Work

2.1 Introduction

In this thesis, similar to Named Entity Recognition (NER), entities are considered as identifiable, named individuals with separate existences, such as movies, restaurants, and books, and each entity is described by a set of words that forms a document (mostly a long document). The words will be used to find the salient features of the domain that the entities belong to, and the documents will be used to represent these entities. Therefore, in this chapter, a review of the literature in learning word and entity embeddings is included. Moreover, our aim in this thesis intersects, in principle, with different active areas within the fields of AI and cognitive science. As a result, this chapter comprises a review of the literature in these areas, namely, conceptual spaces, disentangled representation learning, and subspace clustering.

This chapter is divided into five sections. Section 2.2 gives a general overview of some of the state-of-the-art models for learning word embedding. Entity embeddings are discussed in 2.3 as we take a broad look at the work that has been done in this area, with more details provided regarding the approaches that we adopt in our models. Section 2.4 focuses on explaining the conceptual spaces framework and presents some of the recent works that aim to explore and learn such representation. In Section 2.5, we discuss disentangled representation learning and explore whether disentangled representation learning approaches can be used to learn facet-specific spaces. In Section 2.6,

we explain what the subspace clustering problem is and we discuss the different types of the subspace clustering algorithms. Finally, a summary of the chapter is presented in Section 2.7.

2.2 Word Embeddings

Word embedding models are ways of learning words representation as a dense vectors in a continuous vector space. These models can capture many semantic relations as well as syntactic relations. Moreover, the vector representations learned by these word embedding models have been proven to perform well in a wide range of tasks, such as named entity recognition [54], sentiment analysis [149], and information retrieval [110, 107].

Word2vec [104] is one of the most famous word embedding models. Using neural networks, word2vec represents each word in a manner that pushes concurrent words with similar contexts closer together in the vector space. In word2vec, word embeddings can be learned using one of two ways (or tasks): Continuous bag-of-words (CBOW) and Skip-gram. In CBOW, the embedding is learned by predicting the target word using its context words. At each time step, a window of n words around the target word is used as the input to the model, which is passed to an embedding layer. Then, the word embeddings are passed to a hidden layer to calculate the average vectors, without taking the order into consideration, and transferred to a softmax layer to predict the target word.

Skip-gram, on the other hand, contrasts entirely from the CBOW in its aims. The structure of the skip-gram model utilizes the target word to estimate neighbouring words. To illustrate, the input to the model is a one-hot vector of the target word, which is passed to a hidden layer of size $V \times d$, where V is the size of the vocabulary and d is the dimension of the embedding, which is a hyperparameter. The hidden layer output is then passed to a softmax layer, which generates a probability distribution over the

vocabulary. While Skip-gram represents words very well, it is not efficient enough for large training sets. Therefore, in [105] several approaches are proposed to make the skip-gram model more efficient. One of those approaches is the skip-gram with a negative sampling (SGNS). Rather than updating all the weights for the negative words, i.e the non-context words to the target word, the model updates a sample from all the negative words.

GloVe is another word embedding model [117]. GloVe is designed with the motivation to capture the linear regularities and semantic relationships from the ratios of word–word co-occurrence counts by minimizing the following objective function:

$$G = \sum_{i,j=1}^{V} f(x_{ij})(w_i.\tilde{w}_j + b_i + \tilde{b}_j - \log x_{ij})^2 \tag{2.1}$$

where w_i and $\tilde{w}_j$ are the vectors for the word i (as a target word) and the word j (as a context word), respectively, while b_i and $\tilde{b_j}$ are the biases. Furthermore, X_{ij} is the number of times that the words i and j co-occur. The function $f(X_{ij})$ is a weighting function that tries to minimize the effect of rare words. It is defined as $f(X_{ij}) = 1$ if $x > x_{max}$ and $(\frac{x}{x_{max}})^\alpha$ otherwise, where α is often fixed to 0.75 and x_{max} is the cut-off, which is normally fixed to 10 or 100.

A large number of studies have put forward extensions, modifications and evaluations of these embedding models. As an example, the authors in [111] were motivated by the fact that in natural language, a word can have different meanings based on the context; thus, they extended the skip-gram model so that it would learn multiple vector representations for each word that would correspond to its multiple meanings. The different definitions for each word are obtained by clustering the embeddings of the context words around the target word. After that, in order to predict the sense, the context vector is formed by taking the average of the embeddings of the contexts words and then assigning the sense to the cluster that is closest to that context vector.

Moving beyond linear context, Levy and Goldberg [82] extend SGNS by considering dependency-based context, i.e. instead of using a window of n words, the con-

text is derived based on syntactic relations. They show that varied contexts generate noticeably different embeddings. Their results show that dependency-based contexts produced more functional similarities than linear contexts, which had more topical similarities. Ethayarajh et al. [45] proposed a method to remove undesirable word association; specifically, their method aims to remove gender-biased analogies, such as *doctor:nurse::man:woman*, from the word embedding space, while at the same time preserving gender-appropriate ones, like *king:queen::man:woman*. Wang et al. [141] conducted in-depth evaluation of a number of word embedding models using both intrinsic and extrinsic evaluation methods. The intrinsic evaluation methods were used for evaluating the quality of the embedding based on some criteria, such as reliability, and robustness against lexical ambiguity, while the extrinsic evaluation methods were used to evaluate the impact of using word embeddings as input features to NLP tasks, such as named-entity recognition.

In general, according to the analysis in [11], word embedding models outperform the traditional count-based methods in many lexical semantics tasks. However, Levy and Goldberg [83] have proved that the mathematical foundations of word embedding models (in particular SGNS) are implicitly based upon matrix factorization, meaning these models are similar to traditional distributional models. Moreover, in [84], the authors demonstrate that the word embedding models' hyperparameters, such as the size of the negative sample, are one of the principal reasons behind the performance quality of these models. Additionally, they prove that transferring these hyperparameters to count-based methods also improved the performance of these methods.

2.3 Entity Embeddings

Entity embeddings are vector space representations of the entities. An entity embedding is a mapping from a set of entities (e.g. documents, users, movies, or concepts) into a vector space, such that each entity is represented as a dense vector. Such representations are commonly used in cognitive science, where they are referred to as semantic spaces or conceptual spaces [48]. As another example, the field of information retrieval also has a long tradition of using vector space representations to represent entities [121, 34]. In the field of NLP, recent years have witnessed an explosion of applications that rely on entity embeddings. For instance, entity embeddings are now commonly used for injecting background knowledge [95, 90], and as core representations for recommender systems [151] and entity-focused search [138, 66, 152].

Accordingly, a wide variety of approaches have already been proposed for learning such representations, most of which essentially try to learn vectors that represent the similarity structure of the given domain. Some approaches learn the representations using structured data, such as relational data extracted from a knowledge base. This is the case for knowledge graph embedding models, e.g. TransE [17]. Another way to represent entities is to start from textual descriptions, i.e. viewing each entity as a document and using document representation methods such as Doc2Vec [81]. We refer to such approaches as word-based entity embedding models. In addition to these types of embeddings, there are methods that combine the two resources (i.e. structured data and textual description) to learn entity embeddings [96, 64]. Since the goal of this research is to learn conceptual spaces from textual data, we use only word-based entity embedding approaches. These approaches are based on the principle of compositionality, which states that the meaning of a longer text (such as a document) depends on the meaning of its components (i.e. words and sentences) [132]; the next section will describe some of these methods in more detail.

2.3.1 Document Embedding Approaches

2.3.1.1 Classic Techniques

The bag-of-words (BoW) model is a well-known and basic method for representing text documents [97]. For a collection of documents and a vocabulary set consisting of n words extracted from this collection, BoW represents each document as a vector of length n that encodes how many times each word in the vocabulary appears in the document. Thus, BoW is a count-based approach that neglects the order of the words within the document. Despite the fact that BoW is used with great success in many tasks in text mining, NLP, and information retrieval, there are many issues associated with this model. One crucial issue is that BoW does not distinguish between common, uninformative, words and informative words, i.e. it treats all words equally. To deal with this issue, many weighting functions have been used in place of the original frequency count, such as term frequency–inverse document frequency (TF–IDF) [97] and pointwise mutual information (PMI) [135]. In addition to this issue, BoW is not optimal for capturing document similarity, as documents on similar topics but with different words will have very different vectors [154]. High dimensionality is another issue of BOW, therefore, some kind of dimensionality reduction is commonly used to acquire vectors whose segments correspond to concepts rather than words. An example of this type of model is latent semantic analysis (LSA) [34]. LSA tries to find semantic relationships between words and documents; between documents, based on the words in each document; and among words, based on the context in which those words appear in documents [41]. This is achieved by first constructing a document–term matrix, using either the basic BoW model or TF–IDF. Then, to reduce sparsity, truncated singular value decomposition (SVD) is applied, where each document is represented by the principal components from the initial vector space. The resulting space is then used to calculate similarity among entities. Multidimensional scaling (MDS) [32] is another classical statistical dimensionality reduction technique, which simultaneously reduces the dimensionality and maintains dissimilarity between entities by building a vector

space representation from a pairwise similarity matrix. Broadly speaking, there are two types of MDS. First, in metric MDS, the distance between entities in the resulting embedding and their original distance are required to stay as close to each other as possible; we refer to this type as MDS. Second, non-metric MDS has a less restrictive objective, requiring merely that the distance between the entities in the new embedding should match a monotonic transformation of their original distances. MDS takes a distance matrix as input (which could consist of human-generated similarity judgements, pairwise Euclidean distances, angular differences, etc.). The aim of MDS is then to minimize an objective function commonly called $stress$ function:

$$stress = \left(\sum_{i,j \in \{1,\ldots,N\}} (d_{ij} - \tilde{d_{ij}})^2 \right)^{\frac{1}{2}} \tag{2.2}$$

where d_{ij} is the distance between i and j from the input matrix, and $\tilde{d_{ij}}$ is the Euclidean distance in the new embedding. In non-metric MDS, d_{ij} is replaced by an unknown monotonic transformation function $f(d_{ij})$. MDS can be used for different purposes; in cognitive science, it is commonly used to visualize high-dimensional data and interpret similarity between entities, [115, 60], or to learn conceptual spaces [36, 48]. MDS is also used in many psychological domains to uncover the dimensions affecting certain psychological perceptions and behaviours, which is useful to describe the data [130, 67] and to help in revealing its hidden structure [49].

Probabilistic topic models, such as Latent Dirichlet allocation (LDA) [15], are another common class of methods for representing documents. The LDA method learns a representation in which each text document is represented as a multinomial distribution over latent topics. The topics themselves are multinomial distributions over words. Some authors have also tried to combine ideas from topic modelling with word embeddings, such as [33], where topics are represented as Gaussian distributions over a word embedding space instead of the usual multinomial distributions. They also found that their approach gave qualitatively different topics compared to LDA. Hierarchical LDA (hLDA) model [52] is an extension of the LDA where topics are represented as a tree structure and documents are then assigned a path of that tree.

2.3.1.2 Unsupervised Document Embedding Techniques

In recent years and following the success of word embedding models, a large number of studies have proposed utilization, extension, or modification of the word embedding models, with the aim of learning document representation. The authors in [154], motivated by the belief that word embeddings encode semantic similarity more accurately than word occurrence statistics, improved the BoW representation by substituting the hard mapping used in BoW with a fuzzy mapping using word embedding similarity, i.e. the connections between words in documents in the resulting space are fuzzy numbers. Another way to use word embeddings to extend BoW model representation is proposed in [77], where the word embedding vectors are clustered to form concepts through which a bag-of-concepts representation is constructed. Averaging word vectors [129] is another utilization of word embedding, which represents the document by averaging the word vectors for all the words belonging to that document. In [128], word2vec pre-trained vectors and word sense vectors, learned using BabelNet, were both used to learn rich document representation. Le and Mikolov propose Doc2Vec [81], which is a generalization of word2vec that intends to project a document into a latent d-dimensional space. Two variants are proposed: a distributed memory model (dmpv) and distributed bag of words (dbow). The dmpv model tries to predict the next word, giving a number of contexts with fixed length, sampled from the document. The document vector is shared across all the words in the document, while the word vectors are shared across all the documents. In dmpv, the order information is preserved, unlike in dbow, where the model is forced to predict randomly sampled words from the documents without the use of the context words in the input. Doc2vecC [27] is similar in architecture to Doc2Vec, except that in training the model the document vector is the average of the vectors of randomly sampled words from that document, which helps in capturing the semantics of the document. To speed up the training process and enhance performance, Doc2VecC corrupts the document during the learning by randomly removing words from the document. Additionally, the authors in [133] ar-

gue that Doc2Vec and similar methods can be improved by replacing dot product with cosine similarity, due to its ability to work as a regularization component to prevent the model from increasing the magnitude of the vectors to increase the similarity.
Similar to word2vec, GloVe can also be adapted to learn document representation by replacing word–word co-occurrence with document–word co-occurrence. The GloVe model as a document embedding model can be formulated as follows:

$$G = \sum_{j=1}^{m} G_j \quad G_j = \sum_{i:x_{ij}>0} f(x_{ij}) \left(\mathbf{e_i} \cdot \tilde{\mathbf{w}}_\mathbf{j} + b_i + \tilde{b}_j - \log x_{ij} \right)^2 \quad f(x_{ij}) = \left(\frac{x_{ij}}{100} \right)^{0.75} \tag{2.3}$$

Here $\mathbf{e_i}$ represents the embedding of document e_i, and x_{ij} is the number of occurrences of w_j in the BoW representation of e_i. The term G_j captures to what extent the document embedding is capable of modelling the word w_j. An advantage of this formulation is that, in the case of entity embeddings, we can easily combine the GloVe objective with additional terms that rely on structured data. Therefore, [66] proposes this generalization of GloVe to learn low–dimensional subspaces to represent entities using textual descriptions and structured data. A variation of GloVe is also used in [69] to learn a geographic location embedding, using both textual description and structured information, where the textual description consists of Flickr tags of photos taken close to the location.

2.3.1.3 Supervised Document Embedding Techniques

Another paradigm of work is the use of a supervision signal (usually classification labels) to learn document representation. Broadly speaking, most supervised models that have recently been proposed to learn document representations use neural network architectures. In [35], convolutional neural networks (CNN) are used to learn word embeddings, sentence embeddings, and document embeddings simultaneously, to be able to map the meanings of words to those of documents. Similarly, in [132], CNN or long short-term memory (LSTM) networks are used to transform word vectors into

sentence vectors. Then a gated recurrent neural network (RNN) is used to learn the representation of the document from the sentence embeddings. A generalization of LSTM models is proposed in [91] to deal with long text more efficiently. This is because LSTM models need to keep the important discovered features and transmit these features through the sequence, hence when the text is long, some of this information may be lost during the transmission. The solution proposed in [91] consists of multiple LSTMs trained at different time periods which allow the information to be kept for long distances; as a result of this, the modelling of long documents is improved. These methods use single classification tasks to train the model; other methods learn the representation across multiple classification tasks [92, 93].

In this thesis, we rely on entity embeddings that are based on BoW representations, more precisely, GloVe and MDS.

2.4 Conceptual Spaces

Motivated by the way humans learn concepts very quickly, Gärdenfors in [48] proposed a new framework to learn concept representation. He argues that the two representations mostly used in cognitive science (i.e. the symbolic approaches and the sub-symbolic approaches, such as neural networks) fail to model quick learning of concepts. This is because concept learning heavily depends on similarity judgements, which is not possible in symbolic approaches as similarity cannot expressed using logical notation. The neural network, on the other hand, suffers from two main limitations when learning concepts: 1) it is very difficult to interpret the resulting representation, and 2) it is required to use a large number of examples to learn concepts[1]. Therefore, Gärdenfors proposed the conceptual space framework to represent entities as a metric space with geometric structure. This structure and the fact that we can measure distance in the space gives the representation the ability to model similarity more naturally.

Gärdenfors defines a conceptual space as a high-dimensional vector space that is spanned by a set of **quality dimensions**. Each quality dimension represents a measurable and cognitively meaningful feature in the space, i.e. the quality dimension assigns a feature to the entities. We can then measure to what extent two entities are similar or different based on their distance with respect to that quality dimension. Features that describe temperature, weight, and height are examples of quality dimensions. The relation between two quality dimensions is either integral (if they inherently belong to the same aspect, e.g. *hue* and *brightness*), or separable (if they are meaningful in isolation from each, e.g. *size* and *brightness*). Based on these relations, the conceptual space is structured as a set of low-dimensional **facets**, each of which represents one aspect of the entities, e.g. color, and shape. These facets are formed by a set of integral quality dimensions that are separable from other dimensions. In principle, building these

[1]Recent Language models such as [21] can generalize from a few examples, but the interpretability is still an issue.

facets is the fundamental purpose of the quality dimensions. The assumption behind the decomposition of the conceptual space into facets is that we can describe an entity using a subset of the features independently from the other features. For example, we can describe the color of an object independently from its size. Generally, we can see that the conceptual space framework combines the concepts of semantic space and feature representations into one united method; therefore, it allows for a richer, more informative representation [136].

Properties, which are mostly described by adjectives (e.g. red, green, cold, warm), are represented as a convex region in a single facet. Concepts are then represented as a number of convex regions, in different facet spaces. For example, a dog is a concept that can be defined by the correlation between a number of facets, e.g. colour, shape, sound, and temperature space. These facets are not equally important. As an example, the shape facet is more important than the temperature facet for identifying a dog. The convexity assumption is related to the betweenness relationship, which is very important in measuring similarity and subsequently in learning concepts. The betweenness relationship, simply means that if two points in the space have a certain property, then all the points in between have this property as well.

Based on a variety of psychological evidence, Gärdenfors argues that a weighted Euclidean metric should be used to measure the facet-specific similarity, and a weighted Manhattan metric should be used to aggregate the facet-specific (dis)similarities. The conceptual space is then defined as the product space of its constituent facets' spaces. In this thesis, we take a broader view of conceptual spaces, viewing them as geometric representations of entities, in terms of (i) having interpretable quality dimensions (ii) that are organized into facets.

The conceptual spaces framework has been useful in many areas in AI, cognitive science, and computational linguistics. For instance, the authors in [87, 88] argue that a conceptual space, as an intermediate level between symbolic and sub-symbolic representations, is needed to improve cognitive architecture systems. In [85], Lewis and

Lawry used conceptual spaces to model concept combinations. Bechberger and Kühnberger [12] propose a formalization of conceptual spaces that is able to model correlations between different facets. They argue that because of the convexity assumption, it is not possible to model the correlation between facets in a geometric way. For this reason, they replaced the convex sets with star-shaped sets. Within AI, conceptual spaces are used in many areas, such as computer vision [25], robotics [26], and plausible reasoning [125]. In linguistics, several authors suggest that the adaptation of conceptual spaces will lead to enhanced semantic inference and meaning representation. As an example, conceptual spaces are used by [16] in building a categorical compositional distributional model that can be used to infer the meanings of sentences from two sources: the meanings of their constitutive words and their grammatical structures.

The problem of learning conceptual spaces from data has only received limited attention to date. One exception is the work of [36], which we build on in this thesis. In their work, they proposed an unsupervised method that uses text descriptions of the considered entities to identify semantic features that can be characterized as directions. Their core assumption is that words describing semantically meaningful features can be identified by learning for each candidate word w a linear classifier (linear SVM) that separates the embeddings of entities that have w in their description from the others. The performance of the classifier for w then tells us to what extent w describes a semantically meaningful feature. Each candidate word (i.e. feature) w is represented by the normal vector of the hyperplane that was learned by the linear SVM classifier. This allows learning a linear transformation from the given embedding space to a "disentangled" representation, where entities are represented as vectors whose coordinates correspond to coherent semantic features. The dimensions of this disentangled representation can intuitively be interpreted as the quality dimensions of a conceptual space, in the sense that they allow us to rank entities according to how much they have the corresponding feature. In other words, the problem Derrac and Schockaert [36] consider is how to derive a *feature-based* (i.e. disentangled) description of a given set of entities. For instance, in a semantic space of movies, they found dimensions corres-

ponding to features such as *scary*, *horror*, and *zombie*. Note that because these features tend to be correlated, the corresponding dimensions are typically not orthogonal in the input vector space. For this reason, they refer to these dimensions as *interpretable directions.* More recently, [1] has proposed a post-processing method to fine-tune these interpretable directions. Using textual description of entities and structured information, in [64] and [66], entity embeddings were learned, where instances of semantic types were grouped and represented in a low-dimensional subspace. Such subspaces can be seen as approximate conceptual spaces that are embedded into a single higher-dimensional vector space. The interpretable quality dimensions that are obtained from these subspaces were found in [19] to be useful for inductive reasoning. In [18], the authors used this type of entity embedding to learn representation for concepts as soft regions. Other attempts to learn conceptual spaces are made in [86, 87], where they used linguistic resources such as BabelNet and NASARI.

Somewhat related to the approaches developed in this thesis, [119] proposes a supervised method to decompose word embeddings into orthogonal and uncorrelated subspaces that capture particular aspects of word meaning, namely, sentiment polarity, part-of-speech, concreteness, and frequency. They extracted labels (e.g. positive/negative or noun/verb) from a combination of lexicons. The transformation matrix was then learned by training a model to maximize the distances between all the word pairs with different labels and minimize the distances between word pairs with the same labels. The idea of decomposing a word embedding space into subspaces is also central to the supervised method of [4], which is aimed at distinguishing synonyms from antonyms. Within a broader context, [10] propose a supervised method to learn data-driven conceptual spaces, with the aim of generating better linguistic descriptions of numerical data. In their approach, feature selection methods were used to group quality dimensions into facets. Based on a set of labelled training examples, the quality dimensions that are predicative of particular class labels belong to the same facet.

2.5 Disentangled Representation Learning (DRL)

Generative models are an unsupervised learning class of models that try to learn the distribution of the input data in a way that enables the model to generate new data points from the same distribution. In the last few years, a large number of generative neural network models have been proposed, with variational autoencoders (VAEs) [78] and generative adversarial networks (GANs) [50] being the best-known examples. The main underlying idea behind these models is that high-dimensional data (e.g. images) can often be described in terms of a much lower-dimensional latent vector space. Each object can thus be compactly described by its latent code, i.e. the corresponding vector in this latent space. The problem of DRL is to learn a latent vector representation that is such that (groups of) the dimensions of the latent codes correspond to meaningful interpretable factors. Disentangled representation learning has mainly been studied in the context of images, where having a disentangled representation allows one to manipulate images in a given prescribed way (e.g. generating an image showing what a person would look like when wearing glasses). Apart from this particular use case, disentangled representations have also been said to lead to more robust models (e.g. being less susceptible to adversarial attacks), and help in transfer learning and few shot learning settings. A variety of unsupervised and semi-supervised approaches for learning such disentangled representations have been proposed, such as InfoGAN [30], which is based on a modification of the loss function for GANs, and [58, 76, 28], which instead uses VAEs as the base model. Conceptually, these approaches modify the loss function of a given generative model by insisting that the dimensions of the latent vector space are in some sense independent. For example, in the case of InfoGAN, this is accomplished by including a term in the loss function which aims to maximize the mutual information between different dimensions. Such models essentially try to find independent factors of variations in the dataset, which is most successful if there is a lot of regularity in the dataset. For instance, a typical application is to learn latent codes of facial images, where factors such as gender, the presence of glasses, or the rotation of

the head can be discovered. When learning entity embeddings from text, however, similar strategies tend to be far less successful. In preliminary experiments with InfoGAN, for instance, we were not able to identify any meaningful dimensions for the datasets considered in this research. In other settings, disentangled representation learning for text has proven more useful. For instance, several authors have focused on separating style (or sentiment) from content [70]. In general, most existing approaches for text use some kind of supervision signal aspect-specific similarity judgements [63]. In the context of sentiment analysis, some work has been done on learning representations that disentangle different aspects mentioned in reviews [120]. In [57], an autoencoder was used to discover aspects and learn disentangled representations for them, without supervision. The model input is a review sentence represented as a weighted summation of the pre-trained word embedding vectors of its words. The number of the aspect representation dimensions is the same as the number of dimensions of the pre-trained word embedding vectors. The discovered aspects are fine-grained and thematic. For example, in the restaurant domain, they discovered aspects for main dishes, desserts, and drinks; these aspects are manually grouped into the food aspect. In [44], a general unsupervised disentangled representation learning method is proposed, which is also applied to text. However, this method is only evaluated intrinsically based on statistical measures of disentanglement, which do not reveal how semantically meaningful the learned features are. The aforementioned methods focus on representations which are relatively low-dimensional (e.g. typically involving 100 to 300 dimensions). Besides such methods, several approaches have been proposed for learning disentangled text representations which are much higher-dimensional. For example, Gabrilovich and Markovitch [47] learn document vectors with one coordinate for each Wikipedia page, encoding how related the document is to the corresponding Wikipedia entity. Similarly, Camacho-Collados et al. [22] proposed entity representations in which dimensions correspond to BabelNet synsets. In principle, the latent vector spaces learned by DRL methods can be viewed as way to find quality dimensions. It is unclear, however, whether purely statistical measures of independence can be sufficient for learning

semantically meaningful factors. While this is related to our aims in this research, it should be noted that our focus is on finding sub-spaces that capture different facets of similarity, regardless of whether the individual dimensions are interpretable. While interesting results have been obtained for particular applications, after a thorough empirical analysis, Locatello et al. [94] concluded that such results were highly sensitive to the random initialization of the neural network models and the value of hyper-parameters. Their results suggest that, in the absence of a suitable supervision signal, high-quality factors can only be learned in the presence of a strong inductive bias.

2.6 Subspace Clustering

The curse of dimensionality is an obstacle when it comes to understanding and dealing with real-life data. Thus, many methods have been put forward to re-represent the high-dimensional data as low-dimensional subspaces. These methods were built based on the assumption that the high-dimensional space consists of the union of low-dimensional subspaces. One special case is that of principal component analysis (PCA), where the number of subspaces is assumed to be 1. Finding the low-dimensional subspaces within the high-dimensional data is called the subspace clustering problem. The aim here is very similar to our aim in this thesis, therefore in Chapter 5, we test whether we can learn meaningful facets using the subspace clustering algorithm. Decomposing a high-dimensional space into low-dimensional subspaces is challenging, especially for large datasets, because of the presence of noise in real-life data, and the fact that membership of a data point to the subspaces is mostly unknown. This problem has found numerous applications, mostly in computer vision. There are many algorithms to solve this problem. In general, they can be divided into groups as follows.

2.6.1 Algebraic Methods

There are two main types of algebraic methods. The first is the method based on matrix factorization, such as that proposed in [74], which factorizes the data to create a similarity matrix that is then used to cluster the data points into subspaces. In cases where the subspaces are independent, these methods successfully find the subspaces. However, they are likely to fail when the independence conditions are not met [43]. Second are the geometric-based approaches, such as Generalized Principal Component Analysis (GPCA) [139]. GPCA represents the subspaces with a set of polynomials, where the number of subspaces is the degree of those polynomials. While GPCA is able to find subspaces of different dimensions, it is sensitive to outliers and the complexity of the model increases with the number of dimensions and the number of subspaces [43].

2.6.2 Iterative Methods

Starting from initial segmentation, iterative methods find the subspaces by iteratively fitting PCA models to each group and then allocating points to the closest subspace that is founded upon minimizing the PCA reconstruction error of every cluster [101]. One example is the k-subspaces algorithm, which is a generalized version of the k-means algorithm, [142]. The main disadvantages of such methods are that the number of subspaces must be given, and that they are sensitive to initialization and outliers.

2.6.3 Statistical Methods

These methods assume that the data inside the subspace has a certain distribution, therefore it is more robust against outliers. One example is Mixtures of Probabilistic PCA [134], which assumes that the data distribution inside each subspace is Gaussian; the Expectation Maximization (EM) is then used to iterate between clustering the data

points and estimating the parameters of the mixture model. As with the iterative approaches, the main drawback is that the number of the subspaces, or at least (in some approaches) the number of dimensions, needs to be known.

2.6.4 Spectral Clustering-based Methods

These approaches find the subspaces through two main steps: first, an affinity matrix is constructed, then spectral clustering is applied. One example of these methods is the Local Subspace Affinity (LSA) [147]. The main assumption of LSA is that a point and its nearest neighbors (NNs), based on angular or Euclidean distance, usually belong to the same subspace. Using PCA, a local subspace can be fitted to each point and its k-NNs, where k is the number of neighbors. In order to build the affinity matrix, LSA measures the similarity between pairs of points based on the local information surrounding every point. Then, spectral clustering is applied to the affinity matrix in order to separate the data. LSA is robust against outliers because outliers are more likely to be excluded, as they diverge considerably from the other points. The main issue in LSA appears when engaging with points close to the intersection of two subspaces, because the neighbor of a point could then be a member in another subspace. Another issue is LSA's sensitivity to the correct choice of the neighbourhood size k.

In [43], the authors proposed the sparse subspace clustering method (SSC), which is based on the sparsity principle, such that if we have N points, the nearest neighbors of a point could be any of the $(N-1)$ remaining points, which is much larger than k in LSA. The assumption in SSC is that if a point is a member in a linear subspace with d dimensions then it can be written as a sparse linear combination of d data points. This sparse linear combination can be obtained by minimising the number of the nonzero coefficients, using ℓ_1 normalization. Then the resulting sparse representation is used to form the affinity matrix. As a final step, spectral clustering is applied to obtain the subspaces. Although SSC can handle noise and outliers, it is not yet sufficiently scalable to be suitable for large datasets. In light of this, You et al. [148] proposed a more

efficient variant called SSC-OMP, which uses orthogonal matching pursuit (OMP) to find a sparse representation, which is more efficient than the ℓ_1 normalization used in SSC.

2.7 Summary

In this chapter we have discussed the background knowledge related to entity embeddings and briefly explored some classical and recent techniques, focusing on the methods used in this thesis. Conceptual spaces and disentangled representations have also been discussed, with a review of some of the recent work in each area and how it relates to the goal of this thesis, to obtain a fuller understanding. We have also explored some of the machine learning basics used in this research. The next chapter will cover the datasets used in this thesis to evaluate the different methods we developed for learning facet-specific representations.

Chapter 3

Data Acquisition, Preprocessing, and Representation

3.1 Introduction

This chapter gives an overview of the datasets used in this thesis. The datasets were chosen based on the following requirements: 1) the dataset should comprise a set of entities that belong to one domain (e.g. a set of books or a set of cars); 2) each of these entities should have textual descriptions allowing the salient features of the domain (using the words) to be induced; 3) the entities should have a set of properties and categories that can be used to formulate classification tasks. These classification tasks are important to evaluate our models. Based on these requirements, seven datasets were used: Movies, Place Types[1], Locations, Buildings, Organisations, Bands, and Wikipedia[2]. Each dataset comprised a set of entities, with each entity associated with a textual description. All these datasets are domain-specific, except for the Wikipedia dataset. In Section 3.2, the source and acquisition process for each dataset will be described, as well as the classification problems associated with the dataset. Section 3.3 will cover in detail how the datasets were preprocessed and give statistics about each dataset (e.g. size of vocabulary and number of entities). Section 3.4 explains which vector space representations were used and how they were obtained. Finally, Section 3.5 explains

[1] `http://www.cs.cf.ac.uk/semanticspaces/`

[2] `https://github.com/rana-alshaikh`

how we learn interpretable dimensions that are similar to the quality dimensions of conceptual spaces.

3.2 Data Collection

This section highlights the datasets that we collected to evaluate our models. Of the seven datasets used in this thesis, three are published datasets (Movies, Place Types, and Locations), while the remainder (Buildings, Organisations, Bands, and Wikipedia) were created specifically for this research.

3.2.1 Open Source Data

- **Movies:** This dataset was taken from [36]. Reviews for the top 50,000 movies, based on the number of votes on IMDB, obtained from the following resources: IMDB, IMDB sentiment dataset, Rotten Tomatoes, and SNAP project's Amazon Reviews were collected. Each movie has a corresponding document integrating all reviews for that movie. The 15,000 movies with the longest documents (in terms of word count) were selected for the next stage.

  Classification tasks: In [36], three classification tasks were created to evaluate their methods. The first task is to predict the genres of each film from among 23 classes, based on IMDB classifications. The second task is to predict the IMDB plot keywords of each film from among 100 classes. The third task is to predict the age rating class for each film, from six classes based on certificates from the United Kingdom (UK) and the United States of America (US). All these classification tasks are multi-label tasks; for example, a movie can be both a romance and a comedy, and both USA-PG-PG13 and UK-12-12A.

- **Place Types:** This dataset was taken from [36]. To create this dataset, a set of place types were considered from three different place-type taxonomies: GeoNames (natural features and man-made taxonomy), Foursquare (a taxonomy that focuses on urban man-made places such as restaurants and shops), and OpenCYC (a common-sense knowledge base taxonomy). In particular, the entities in this case are fine-grained place types. The textual description for each place type was then derived from Flickr, a photo-sharing website, where users can add short textual-description tags to their photos. Where a photo was tagged with a given place type, the other tags of that photo were taken as a textual description of that place type; all tags co-occurring with a given place type were integrated into one document. In total, 22,816,139 photos were collected. Only the place types associated with 1,000 photos or more were considered.

  Classification tasks: Similar as for the movies domain, the authors of [36] created three classifications tasks from the natural categories and properties of the entities in this dataset to evaluate their methods. The categories to be predicted are the place types' parents in the taxonomies. For each of the three taxonomies used, one classification task was generated. The Foursquare classification task consisted of predicting the nine top-level categories. The GeoNames taxonomy included seven classes. The third classification task was extracted from OpenCYC and was limited to 20 classes. The Foursquare and GeoNames classification tasks are single-label tasks, while the OpenCYC classification task is a multi-label task.

- **Locations:** This dataset was taken from [68]. The entities in this dataset correspond to geographic locations across the UK, and the bag-of-words (BoW) representations are composed of the tags assigned to geotagged Flickr photos whose associated coordinates are near these locations. The number of location entities is 209,121.

  Classification tasks: The classification task associated with this dataset was to

predict the land cover classes extracted from the CORINE Land Cover (CLC) dataset, which is a European dataset that provides information on land cover with a spatial resolution of 100-metre[3]. The land cover classes were organized into three levels: level 1 (5 classes), level 2 (15 classes), and level 3 (44 classes).

Table 3.1 provides statistics related to the aforementioned datasets and information about the classification tasks associated with each dataset, including a sample of the classes from these classification tasks and whether each classification problem is multi-label (ML) or single-label (SL).

[3] http://www.eea.europa.eu/data-and-maps/data/corine-land-cover-2006-raster-2

Dataset	Entities	Attribute	ML/ SL	No. Classes	Sample of Classes
Locations	209121	CORINE level 1 (CL1)	SL	5	Artificial surfaces, Agricultural areas, Forest and semi natural, Wetlands, Water bodies areas.
		CORINE level2 (CL2)	SL	15	[Urban fabric], [Industrial, commercial, transport units], [Mine, dump, construction sites], ...
		CORINE level3 (CL3)	SL	44	[Continuous urban fabric, Discontinuous urban fabric], ...
Movies	13978	Plot keywords (KeyW)	ML	100	beach, beating, betrayal, doctor, escape,church, gore, gun, helicopter, hero, pregnancy, ...
		Genre	ML	23	Action, Adventure, Animation, Biography, Comedy, Crime, Documentary, Drama, Family,...
		Age Ratings (AR)	ML	6	USA-G, UK-12-12A, UK-15, UK-18, UK-PG, USA-PG-PG13, USA-R.
Place Types	1383	Foursquare (Fours.)	SL	9	Arts and Entertainment, Nightlife Spot, Food, College and University,...
		Geonames (Geo.)	SL	7	Mountain/Hill/Rock, Spot/Building/Farm, Road/Railroad, Undersea,Stream/Lake,...
		OpenCYC (OpenC.)	ML	20	Facility, Aqueduct, Military Place, Foreground, Border,...

Table 3.1: Overview of the Open Source Datasets.

3.2.2 Generated Data

To generate datasets, we chose to use Wikipedia [4] and Wikidata [5], as these respectively provided us with textual descriptions for the entities and a set of attributes from which we were able to generate classification tasks. Wikipedia is a free online encyclopaedia hosted by the Wikimedia Foundation, with content created and edited by volunteers all over the world. Wikidata is a collaborative, multilingual knowledge graph created

[4] https://en.wikipedia.org/wiki/Main_Pag

[5] https://www.wikidata.org/wiki/Wikidata:Main_Page

by the Wikimedia Foundation, which stores information about entities, i.e. concepts, objects, or topics. Each information entry includes a unique number (ID) preceded by the letter 'Q', a label (which is not necessarily unique), description text (which, in combination with the label, is unique), a list of property value pairs, and links to the Wikipedia page (or pages, where these exist in different languages) that describes the entity. Figure 3.1 shows this information. Every Wikipedia article has a unique Wikidata ID and a link to the Wikidata page of the entity described by the article.

Four further datasets were constructed from Wikipedia and Wikidata specifically for this research. Three of those datasets are domain-specific: Buildings, Organisations, and Bands, where the entities were selected based on Wikidata semantic types. The associated classification tasks for these datasets involved predicting the Wikidata attributes of the entities. To obtain natural classification tasks, we looked for Wikidata triplets in the form (h, a, t), where h is an instance of the considered semantic type, a is a certain attribute, and t is the corresponding attribute value. To obtain meaningful attributes, we selected only those attributes a for which a triplet of the form (h, a, t) existed for at least 10% of the entities h of the considered type. For each such attribute, we selected those values t that occurred for at least 100 entities. For example, for the buildings domain, there is a set of attributes provided by Wikidata, such as country, architectural style, color, etc. For each building that has a Wikipedia page, there is a Wikidata page that lists a subset of these attributes with a certain value for each one. To illustrate, for a building like **Cardiff Castle**, one of the attributes is ***Country*** which has the value (i.e. label) ***UK***. Based on Wikidata, the ***Country*** attribute has many labels (UK, USA, Italy...). The ***Country*** attribute is considered as a classification task if, 1) 10% of the set of buildings have this attribute in their Wikidata pages, 2) at least two of the attribute's labels (UK, USA, Italy, etc.) have no less than 100 buildings belonging to each one.

In general, as the attributes and their values are extracted from curated and structured resources, such as taxonomies and Wikidata, we restrict ourselves to the exact

attributes/values, and we do not consider any infliction or paraphrase for these attributes/values. Because the attributes' values provided by such resources are fine-grained and different (i.e. have no repetition), considering infliction or paraphrase for these attributes/value will not affect the classification results. To clarify, if a particular building has the pair (attribute: architecture style, value: modern), the synonyms of modern are not found as a value of the (architecture style) attribute in Wikidata.

- **Buildings:** The entities in this dataset are buildings that have Wikipedia pages. The dataset was created by retrieving all Wikipedia pages whose semantic type on Wikidata corresponds to 'building'. These Wikipedia articles were then taken as the text descriptions of the entities, and the sample was filtered to eliminate instances of Wikipedia pages that were too short, i.e. that contained fewer than 500 words.

  Classification tasks: We used two attributes from Wikidata (these being the only attributes for which a sufficient number of entities per attribute value was found): Country, with two classes, and Administrative Location, with two classes.

- **Organisations:** This dataset was created in a similar way to the Buildings dataset. Wikipedia pages whose semantic type on Wikidata corresponds to 'organisation' were retrieved, and entities with pages of fewer than 500 words were removed.

  Classification tasks: Two classification tasks are associated with this dataset, the first task is to predict the Country with four classes, and the second task is to predict the Headquarter Location attribute, with two classes.

- **Bands:** Using a similar method to that used for the previous datasets, the Wikidata entities with 'musical band' as the semantic type were retrieved, with their corresponding Wikipedia pages.

Figure 3.1: Wikidata General Components.

Classification tasks: The classification tasks for this dataset involved predicting: Genre, with 9 classes; Country of Origin, with 6 classes; and Location of Formation, with 4 classes.

- **Wikipedia:** We compiled this dataset from the English Wikipedia. Specifically, we selected the 100,000 Wikipedia entities with the longest articles, which approximately correspond to those entities whose Wikipedia articles contain more than 500 words before preprocessing.

  Classification tasks: As classification tasks for the Wikipedia dataset, we first considered the problem of predicting the Wikidata semantic types of the Wikipedia entities. We identified 13 semantic types that occurred sufficiently frequently, each having at least 2,000 instances in our collection. In addition to these semantic types, we extracted nine attributes from Wikidata, to each of which a value was ascribed for a sufficient number of instances: three single-label attributes for *movie* entities (Language, with three classes, Country, with eight classes, and Colour, with two classes), two attributes for *music* (Genre, which is multi-label, with thirteen classes, and Country, a single-label attribute, with nine classes), two attributes for *business* (Country, with three classes, and Legal Form, with seven classes) and two attributes for *human* (Gender, with two classes, and Country of Citizenship, with two classes).

We retrieved all the instances and their attributes using the WikiData SPARQL query service [6]. Table 3.2 provides statistics about the datasets and a sample of the classes, with the distribution of each class, associated with the classification tasks, specifying whether each classification problem is single-label (SL) or multi-label (ML).

[6]The crawler was implemented by Zied Bouraoui and is available at `https://github.com/zbghub/wrt`

Dataset	Entities	Vocabulary	Attribute	ML/ SL	No. entities that have the attribute	No. Classes	(Class, Percentage of positive intities) (Top 3 classes)
Buildings	3721	10641	Country	SL	3425	2	(USA, 60%), (UK,9.3%)
			Administrative Location(AL)	SL	2668	2	(Massachusetts, 4.8%), (California, 5%)
Organisation	11800	27321	Country	ML	3800	4	(USA, 56%), (UK, 8.3%), (Canada, 5.3%).
			Headquarter Location (HL)	SL	2930	2	(Washington, 5.8%), (London, 55.1%)
Bands	11448	41663	Genre	ML	9379	9	(thrash metal, 12.2%), (alternative rock, 11.9%), (indie rock, 6.4%).
			Country of origin (Geo.)	ML	6857	6	(USA, 68.5%), (UK, 6%), (Australia, 3.1%).
			Location of Formation (LoF.)	SL	4830	4	(Los Angeles, 7.3%), (New York City, 5.6%), (London, 4.8%).
Wikipedia	100000	235190	Semantic Type (SM)	ML	100000	13	(human, 10%), (business, 9.7%), (human settlement, 8.3%).
			Movies: Language (MoL)	ML	3183	3	(English, 23.3%), (Tamil, 18.7%),(Hendi, 15.3%)
			Movies: Color (Mocl)	ML	2200	2	(color, 89.7%), (black-and-white, 10%).
			Movies: Country (MoC)	ML	3265	8	(India, 58.4%), (USA, 16.7%), (UK, 3%).
			Music: Country (MuC)	ML	2544	9	(USA, 37.8%), (UK, 14.8%), (Germany, 4.9%).
			Music: Genre (MuG)	ML	2566	13	(alternative rock, 10.4%), (pop, 9.1%), (rock, 6%).
			Business: Country (BC)	ML	3644	3	(USA, 36.3%), (Germany, 10.8%), (UK, 7.2%).
			Business: Legal-form (BLF)	ML	2000	7	(joint-stock, 17.4%), (public company, 10.65%), (GmbH, 6.4%).
			Human: Gender (HG)	SL	10087	2	(Male, 79.4%), (Female, 20.5%).
			Human: Country of citizenship (HC)	ML	4704	2	(USA, 14.9%), (India, 9%).

Table 3.2: Overview of the Generated datasets.

3.3 Data Preprocessing and Technical Details

In this section, we describe the preprocessing approach for each dataset and highlight some technical details related to the vocabulary and how each dataset was split to train the classifiers.

The preprocessing approach taken aimed to stay broadly in line with the strategy used in [36]. The text was tokenized to words, and all words were converted to lower case. Non-alphanumeric characters and stop words were removed.

Regarding how the datasets were used for the classification tasks, in the Movies and Locations datasets and Wikipedia dataset, where more labeled data was available, we used fixed splits of 60% for training, 20% for tuning, and 20% for testing. For the remaining classification problems, where few positive examples were available, we split the labeled examples of each problem randomly into thirds, allocating two thirds for training and one third for testing. For tuning, we used five-fold cross-validation over the training set.

Movies: In [36], the dataset was preprocessed, and only adjectives and nouns were retained, as they were more likely to correspond to salient features. Only terms appearing in the reviews of at least 100 movies were considered. The resulting vocabulary comprised a total of 22,903 candidate words. In this research, we heavily depended on pre-trained word-embedding vectors. We used 50-dimensional GloVe word vectors[7], which were pre-trained on the English Wikipedia[8]. Words for which we did not have a pre-trained word vector were removed, resulting in 17,100 candidate terms. Finally, 1,022 duplicated documents found in the dataset were removed, further reducing the number of document entities to 13,978.

[7] In our initial experiments, the efficiency of some of the models that we tried (See Appendix A) strongly depended on the number of the dimensions of the pretrained GloVe model; therefore we chose 50-dimensional GloVe word vectors.

[8] The pre-trained vectors available at `https://nlp.stanford.edu/projects/glove`.

Place Types: In [36], the dataset was preprocessed, and only terms appearing in the tags of at least 50 place types were considered. The resulting vocabulary comprised a total of 21,833 candidate words. As the Place Types dataset contains more noise than the Movies dataset, we only considered tags that appeared with at least 100 place types; this reduced the vocabulary size to 17,545 words. The number of words for which we had a pre-trained word vector was 17,100. The final number of entities was 1,383.

Locations: The original vocabulary contained 978,534 tags assigned to 209,121 entities. Since Flickr tags often correspond to concatenations of different words, we tokenized the tags using Wordninja [9], which splits terms based on English Wikipedia unigram frequencies. We subsequently discarded stop words, using NLTK [14], as well as words for which we did not have a pre-trained word vector. The vocabulary after preprocessing contained 114,002 terms.

Buildings: BoW representations of the Wikipedia concepts were obtained using a standard preprocessing strategy (i.e. removing HTML tags and references), including stop word removal with NLTK [14]. We also POS tagged the documents and retained only the nouns and adjectives. Finally, frequent words, i.e. those that occurred in more than 60% of the Wikipedia articles in the dataset, were removed, as were words that occurred fewer than ten times. Entities whose description contained fewer than 200 words after preprocessing were removed. The final number of entities after preprocessing was 3,721 and the total number of words in the vocabulary was 10,641; we had pre-trained vector embedding for only 9,516 of these.

Organisations: BoW representations of the Organisations concepts were obtained in the same way as those of the Buildings dataset. The final number of entities after removing entities with fewer than 200 words in their description was 11,800. The vocabulary size was 27,327 candidate terms; we had pre-trained vector embedding for 12,852 of these terms.

Bands: BoW representations were obtained using the same preprocessing approach as for the Buildings and Organisations datasets. The number of entities after prepro-

cessing was 11,448 and the vocabulary consisted of 41,663 terms; pre-trained vector embedding was available for only 18,852 of these.

Wikipedia: The same preprocessing approach was used; however, here we used all terms, not only nouns and adjectives, as for this large sparse dataset a larger vocabulary was required to obtain good representations. The total number of entities was 100,000 and the vocabulary size was 235,190 words; a pre-trained word-embedding vector was available for only 173,732 of these.

3.4 Representations

In this section, we will explain how we obtained the vector space representations used in this research. Note that since not all the datasets were used for each experiment, and because of the nature of some of the embedding models, some of the following vector space embeddings were obtained only for a subset of the datasets. More details will be given in Chapters 4, 5, and 6.

MDS: Using MDS, 100-dimensional vector spaces were obtained for the relatively small datasets (Movies, Place Types, Buildings, Organisations, and Bands). For the remaining datasets, MDS representations could not be obtained because MDS is not sufficiently scalable to generate representations for relatively large datasets. To learn robust MDS representations, we used the entire vocabulary (including terms for which we did not have pre-trained word-embedding vectors). The dissimilarity matrix of size $n \times n$, where n is the number of the documents, i.e. the input of the MDS, was calculated as in [36], using the normalized angular difference between the BoW vectors of documents as follows:

$$ang(e_i, e_j) = \frac{2}{\pi} \arccos\left(\frac{v_{e_i} \cdot v_{e_j}}{\|v_{e_i}\| \cdot \|v_{e_j}\|}\right) \tag{3.1}$$

Doc2vec: Using Doc2vec, 100-dimensional vector spaces were obtained for the small datasets where we had the original text: Buildings, Bands, and Organisations. Regarding the hyper-parameters of the model, the window size value was chosen from $\{1, 3, 5\}$, based on the tuning data, while the number of epochs was set to 20, and the minimum frequency of terms was set to 10.

GloVe: Using GloVe, 100-dimensional vector spaces were obtained. A negative sample, which is a random sample of terms that did not appear in the document, was used. According to [69], the use of such a negative sample enhances the representation obtained by variants of GloVe. We set the size of the negative sample per entity to 2,000 for the Movies, Buildings, Place Types and, Wikipedia datasets and to 1,000 for the Locations dataset. The algorithm was trained using AdagradOptimizer, with a learning rate of 0.05 and mini-batch size of 1,000; the number of epochs was set to 10.

In initial experiments, we also considered document embeddings learned by the neural variational document model [103]. However, we found the resulting embeddings to be of much lower quality.

3.5 Learning Interpretable Dimensions

As we mentioned in Chapters 1 and 2, our aim is to learn a vector space representation that is similar to conceptual spaces in terms of having interpretable quality dimensions that are grouped into low-dimensional facets. In this section, we explain how we obtain interpretable quality dimensions that are similar to the dimensions of the conceptual spaces. These quality dimensions will play a crucial role in this thesis. One way to learn such dimensions from data is that proposed by Derrac and Schockaert [36], which we used as a starting point for two of our proposed models in Chapters 4 and 5.

To explain how these directions are obtained, let E be a set of entities of some particular domain (e.g. movies) for which a vector space embedding is given. Each entity is associated with a document containing the textual description, i.e. we have a document collection $\mathcal{D} = \{D_e \,|\, e \in E\}$. Let $V = \{w_1, ..., w_m\}$ be the vocabulary, i.e. the set of all words that occur sufficiently in the document collection (after preprocessing these documents, e.g. removing rare words). In the following, we will write $\mathbf{e} \in \mathbb{R}^n$ for the embedding of entity e, and $F = (f_1, ..., f_t)$ for the set of semantically meaningful features that describe the considered domain. In particular, we want to represent each entity e as a vector $(f_1^e, ..., f_t^e) \in \mathbb{R}^t$ such that each of its components corresponds to a semantically meaningful feature.

The first step is to find F, which is basically a set of words, each of which corresponds to a feature that can be modelled as a direction in the vector space. Then, for $f \in F$ we write $\mathbf{d_f}$ for the vector characterizing this direction. This amounts to identifying vectors $\mathbf{d}_1, ..., \mathbf{d}_t$, in the given embedding, which represent the most salient semantic features. The learned vectors will be referred to as *feature directions* to emphasize the fact that only the ordering induced by the dot product $\mathbf{d_i} \cdot \mathbf{e}$ matters. Formally, this means that $\mathbf{e_1} \cdot \mathbf{d_f} < \mathbf{e_2} \cdot \mathbf{d_f}$ iff e_2 has the feature f to a higher extent than e_1 (e.g. if f denotes the feature *scary*, then this would mean that movie e_2 is scarier than movie e_1). The feature-based representation of e is simply given by $(\mathbf{e} \cdot \mathbf{d}_1, ..., \mathbf{e} \cdot \mathbf{d}_t)$.

To find the feature directions, each word w occurs sufficiently frequently in the document collection D, i.e. all the words in V, is considered as a candidate feature. To determine whether w should be added as a feature, i.e. added to F, we train a linear classifier to separate the vector representations of the entities e for which w is mentioned in D_e from the vector representations of the other entities[9]. We used Logistic regression classifiers, without any regularisation, in our experimints instead of linear SVMs used by [36] (more details on Logistic regression Classifier and linear SVM are provided in Appendix B.2), which we found to perform similarly but were faster to train. If this logistic regression classifier is sufficiently accurate, we assume that the word w captures a salient feature. To deal with class imbalance, the performance of the classifier is measured using Cohen's Kappa metric, which can be seen as a modification of classification accuracy which takes into account the expected performance of a random classifier. Cohen's Kappa metric is a way to evaluate the classifier performance, mostly used when dealing with imbalance data, by comparing the observed accuracy (Accuracy) on one hand and the expected accuracy (ExAccuracy) on the other hand (more details on Cohen's Kappa metric is provided in Appendix B.2.2). The class imbalance comes as a result of the fact that many useful features are mentioned only in a small fraction of the documents. For example, in the movies domain, we have 7376 terms (43% of the candidate features) that appeared in less than 600 documents (4% of the documents); some of these features are important such as (*bollywood, spacecraft, anime, championship, pixar, ...*).

The corresponding feature direction is then characterized by the normal vector $\mathbf{d_f}$ of the hyperplane that was learned by the logistic regression classifier. We will use the notation pos_w and neg_w to refer to the set of entities from E which are classified as positive and the set of entities which are classified as negative, respectively. Instead of

[9]It may seem counter-intuitive to use binary classifiers to learn representations of ordinal features. However, the occurrence or non-occurrence of a word in the description is binary, and this is the most important available signal. We experimented with statistics such as pointwise mutual information, which did not lead to better results.

using these feature directions directly, in their method, Derrac and Schockaert clustered these directions and consider each cluster as a *Primary feature*. More details about the primary features will be given in Chapter 4. Table 3.3 shows the top 50 features in the Movies, Place Types, and Buildings datasets. In the movies domain, we can see that most of the top 50 features are related to genres and sentiment associated with these genres, which indicates the genre facet is the dominant facet in this domain. In the place types domain, some of these features are not about places such as (curly, eyes, one, victims) which reflects the fact that the representation was learned from descriptions of photographs (Flickr tags) rather than descriptions of places. In the building domain, most of the top features are related to educational buildings, this may indicate that a large percentage of the entities in the buildings dataset are educational buildings. The organisations dataset seems noisy, as there are many features that may not be looked at as salient features of the organisations domain (e.g. dogmatic, sorrow, tanker, imitation, fugitive, karma). Some of these features may be considered as a feature of a specific type of organisations; for example, (bombers, ballistic) are related to human rights or anti-war organisations. Similarly, the features extracted from the band dataset also contain unrelated features to the musical bands domain, mostly colors such as blue, green, orange, and purple. Finally, from all these examples, we can conclude that the noise in these datasets and the fact that there is a dominant aspect will increase the challenge of learning facet-specific entity embeddings.

The Top 50 Kappa score words				
Movies	**Place Types**	**Buildings**	**Organisations**	**Band**
gore	hollywood	arena	faithfulness	increment
horror	era	faculties	alternates	anchor
suspense	montreal	national	glimpse	drums
laughs	neighborhood	register	repentance	tour
jokes	marquette	street	sorrow	orientation
hilarious	rangefinder	places	tanker	rock
gags	maria	adobe	organization	label
laugh	latte	university	organisation	albums
thriller	basket	colleges	research	width
comedies	pierres	undergraduate	dogmatic	video
funniest	tokyo	hall	program	film
scary	daily	universities	simplified	studio
awesome	queens	county	imitation	lead
emotional	utica	office	programs	live
documentary	chocolate	mansion	bombers	record
dvds	convict	california	ballistic	sound
laughing	warfare	campus	political	show
blood	providence	main	fugitive	debut
web	curly	city	school	auto
powerful	doors	belgrade	scientific	horizontal
dumb	heroes	construction	right-hand	present
email	cell	first	karma	track
creepy	typography	students	fbi	metal
gory	cute	buildings	appendix	number
killing	cream	family	desegregation	watch
slapstick	shade	floor	rights	layer
effects	eyes	home	development	orange
kids	disgusting	education	jackpots	legend
interviews	stork	basketball	community	bar
comedy	netherlands	sciences	prophet	color
rhodes	cityscape	center	health	bottom
audio	price	games	children	guitarist
delightful	mobile	united	religious	bands
government	tiny	museum	global	drummer
transfer	stripes	farm	organisations	century
incredible	null	faculty	scriptures	common
stupid	rose	academic	support	height
fighting	amigo	hockey	membership	punk
win	van	teams	cognate	purple
adults	maya	states	business	value
funnier	events	palace	state	shift
cheesy	gorgeous	research	care	member
superb	calif	century	government	shows
disc	one	iran	sins	black
crime	number	rico	playoff	text
sexual	groups	ponce	fetus	tracks
intense	flax	roof	pale	green
kill	victim	space	military	blue
terrific	textures	side	federal	chart
emotions	refugee	structure	service	singles

Table 3.3: Top 50 words in terms of Kappa score for the Movies, Place Types, Buildings, Organisations, and Band datasets .

3.6 Summary

This chapter has covered information about the datasets used to evaluate our models, including how the datasets were obtained and preprocessed, the attributes used to create the classification problems and whether each was multi-label or single-label, statistics about each dataset. Moreover, in this chapter we explain how we obtained the vector space representations, and the quality dimensions for the considered datasets. The next chapter will explain how we decomposed the given vector space into facets.

Chapter 4

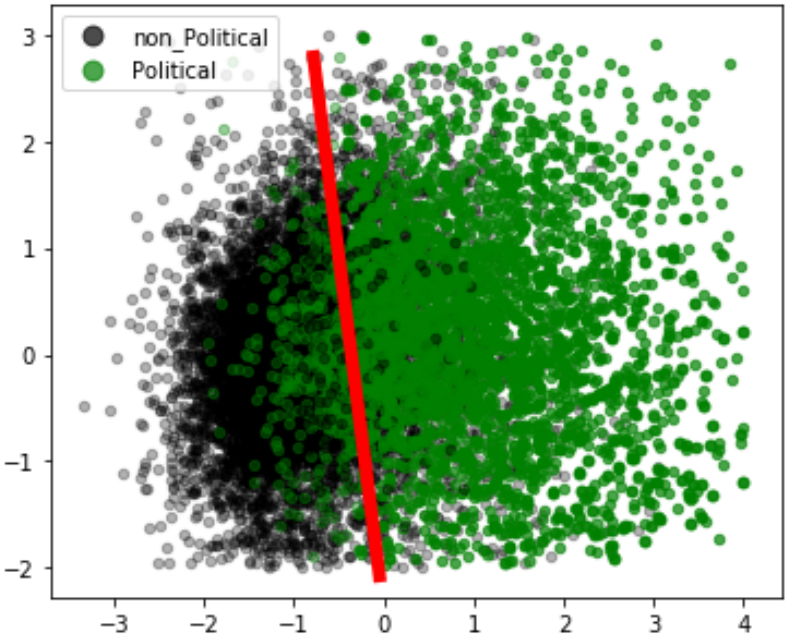
Hierarchical Linear Disentanglement of Data-Driven Conceptual Spaces

4.1 Introduction

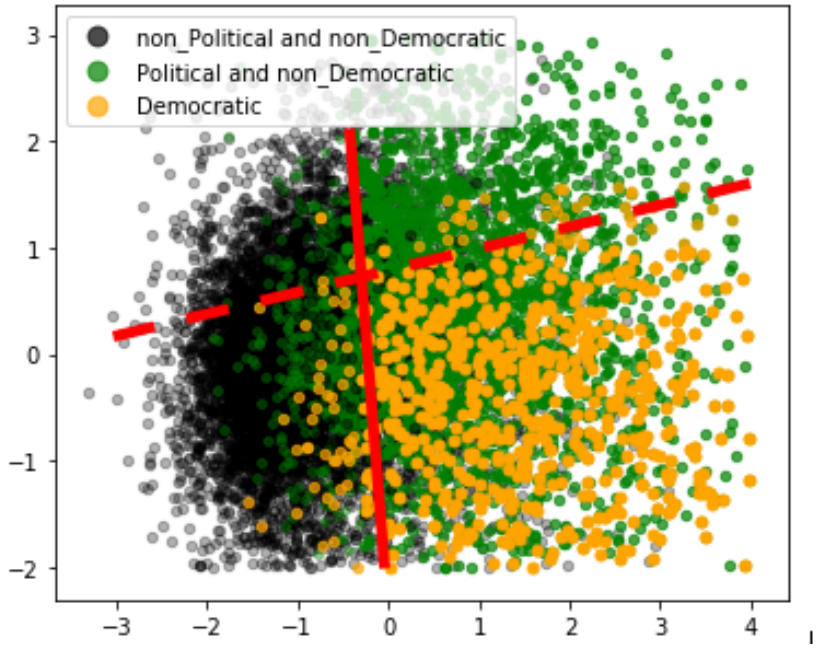
Our aim in this thesis is to learn facet-specific entity embedding by adopting the conceptual spaces framework. In particular, we will try to learn a space similar in spirit to conceptual spaces in two main aspects: (1) the salient semantic features are represented as quality dimensions; (2) these quality dimensions are organized into a set of facets. In the previous chapter, for the first aspect, we adopted the line of work proposed by Derrac and Schockaert in [36] (explained in Section 3.5) to represent semantic features as directions in the space. For the second aspect, we need to find out (i) what are the set of features that describe each facet, (ii) how to model the facets, (iii) how to represent the features in the related facet space, (iv) how to obtain facet-specific representations for the entities. The aim of this chapter is to analyze to what extent directly clustering the directions from [36] can provide us with meaningful facets. The challenging steps are then how to model the facet and how to learn the facet's integral quality dimensions, i.e. how to learn the directions of the facet's semantic features in a meaningful way. The latter is crucial since many semantic features do not make sense for all entities. For instance, in an embedding of Movies, we may consider a feature that captures how closely a movie adheres to the book it is based on.

While meaningful for book adaptations, this feature would be non-sensical for other movies. As an important practical implication, if quality dimensions are learned from the full set of entities, while only being sensical for a subset of these entities, we may expect them to be sub-optimal. This problem is illustrated in Figure 4.1, which displays a projection of an embedding of organisations. In Figure 4.1a, the green dots correspond to those organisations whose associated description contains words such as *political, politic, party, parties, politicians*. Because organisations whose description contains such words are more or less linearly separable from other organisations, the method from Derrac and Schockaert [36] discovered this cluster as a semantic feature. Now consider Figure 4.1b, where the yellow dots correspond to organisations whose descriptions contain words such as *democratic* and *left-wing*. While this cluster describes a feature that is intuitively clear (i.e. organisations associated with left-wing political ideas), this feature is only relevant for a subset of organisations (i.e. political ones). A key, and perhaps surprising, observation is that this is reflected in the vector space. In particular, as can be seen in the figure, this feature cannot be characterized well using a single hyperplane [1].

[1]The fact that the features in 4.1 are not separable in a two-dimensional space does not necessarily mean that they are not separable in a high dimensions space. However, the assumption that our illustration holds in a high dimensions space follows from the numerical results.



(a)



(b)

Figure 4.1: Projection of a 100-dimensional embedding of Organisations (see Section 4.3.3), showing (a) how organisations that are described with words such as *political, politics,party,parties,politicians* (shown in green) are separated from others; and (b) how organisations that are described using words such as *democratic, left-wing* (in yellow) are separated from others.

This problem could be solved by decomposing the entity embedding into different low-dimensional subspaces, each of which focuses on a different facet. However, this approach is very challenging to start with [94]. Therefore, in this chapter instead of trying to find a hard decomposition of the entity embedding into separate facets, we propose a simple but effective method that tries to implicitly decompose the entity embedding into facets. The proposed method is inspired by the fact that the structure of the conceptual spaces can be viewed as a hierarchical one, where the first level of the space are the facets and the second level is the set of quality dimensions corresponding to each facet. Therefore, our method is based on applying the method from [36] in a hierarchical fashion. In the example from Figure 4.1, for instance, we can view the feature *political* as defining a facet. To obtain a suitable characterization of the feature *democratic*, it then suffices to apply the method from [36] to that facet instead of to the full space. In this way, we obtain a set of primary features, and for each of these primary features we obtain a set of sub-features. Our assumptions in this chapter are: 1) that these primary features correspond to facets, 2) that by learning the sub-features using only the related facet's space, we are learning the directions of the facet's semantic features and, at the same time, performing implicit decomposition of the vector space representation into facets. As confirmed by the experimental results, by learning the features in such a hierarchical way, we obtain semantically more meaningful representations than when directly applying the method from [36]. This shows that the intuitive hierarchical relationship that exists between features (e.g. the fact that *democratic* is a sub-feature of *political*) is effectively reflected in the structure of the entity embedding.

Moreover, it is important to note that the relation between the features and the facets (sub/primary) is diverse and not always a hypernym relation, therefore, using work that aims to extract such a relation like [126], [112], [150] to find the features that define a facet will not be successful. For example, in the movies domain, the genre facets and its features can be seen as being in a hypernym relation, e.g. (horror is a genre), while in the organisation domain, for example, the relations between the political facets and

some of its features, such as (agenda, sovereignty, regime, electoral, immigration), are not hypernym relations.

The remainder of this chapter is structured as follows: In Section 4.2, we provide a description of our approach for learning the primary features and the sub-features. In Section 4.3, we provide information about our experiment setup and how we evaluate our proposed model we also provide an analysis of the results from both qualitative and quantitative perspective. We summarise our findings in Section 4.4.

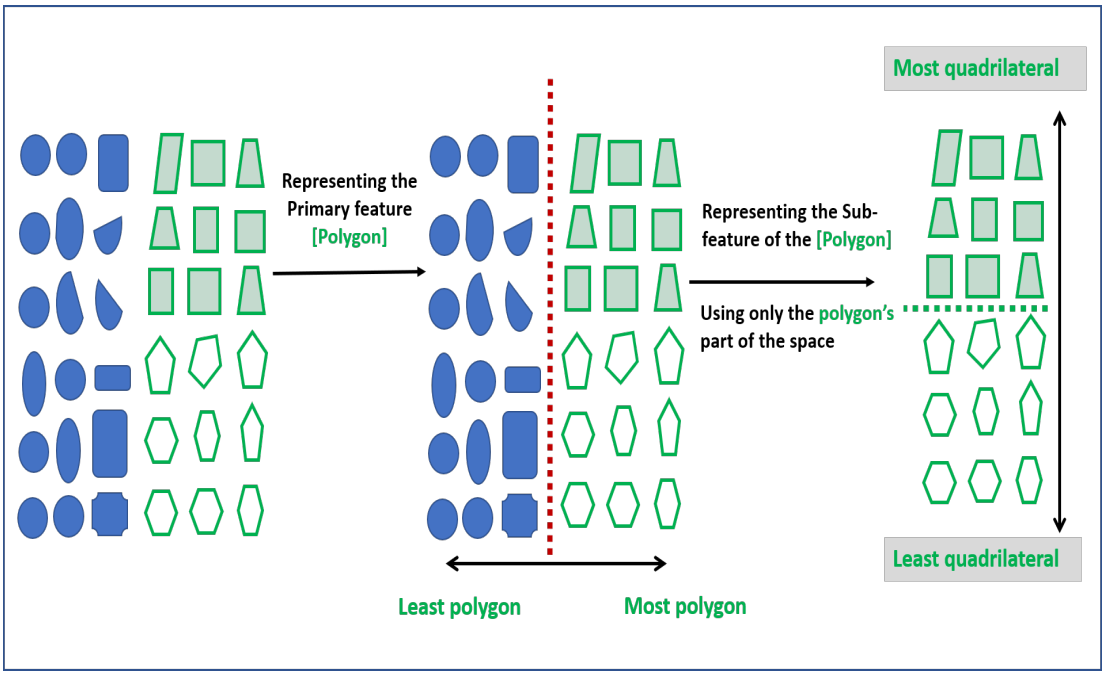


Figure 4.2: A toy example of a 2D shape space. One of the primary features of the space is [polygon]. A linear classifier is used to find the hyper-plane (dotted red line) that separates polygons from other shapes. One sub-feature of [polygon] is [quadrilaterals], i.e. cluster that contains [square, rectangle, rhombus, ...]. A linear classifier is trained only on shapes positively classified as polygons to find the hyper-plane (dotted green line) that separates the quadrilaterals from the rest of the polygons.

4.2 Identifying Feature Directions

As explained in the previous chapter (Section 3.5), we obtained the feature directions $\mathbf{d}_1, ..., \mathbf{d}_t$ that correspond to semantically meaningful features $f_1, ..., f_t$, which were extracted from the text descriptions of the entities. For each entity e, the dot product $\mathbf{d_i} \cdot \mathbf{e}$ represents to what extent e has the feature f_i.

Learning primary features. In [36] to obtain the primary features $p_1, ..., p_k$, The directions $\mathbf{d}_1, ..., \mathbf{d}_t$ are clustered, which serves to reduce the total number of features. This is useful to keep the representation low-dimensional (e.g. to avoid overfitting when training a classifier on the resulting feature-based representation). By associating features with clusters of words, rather than individual words, features are more likely to correspond to salient properties of the domain. Another advantage of clustering the features that we noticed is that irrelevant words tend to be clustered together, resulting in a small number of non-informative clusters, with most of the other clusters corresponding to semantically meaningful features. While Derrac and Schockaert [36] used a variant of k-means, we use affinity propagation [46]. Affinity propagation is a centroid-based algorithm that does not require the number of clusters to be specified, unlike k-means, which is crucial in our hierarchical setting. The affinity propagation, by default, uses the Euclidean distance between points to measure the distance. From our experiments, it was found that using affinity propagation gives better results than k-means; this could be because the number of clusters is induced from the data itself. Another advantage of the affinity propagation over k-means, which could explain the good results, is that all data points in affinity propagation are considered candidate centers, unlike k-means where the centers initialized randomly [46] (see Appendix B.2).

Once the clusters $C_1, ..., C_k$ are obtained, the final step is to associate with each of them a corresponding feature direction. In [36], the direction $\mathbf{d_C}$ for a cluster $\{u_1, .., u_l\}$ is defined as the average of $\mathbf{d_{u_1}}, ..., \mathbf{d_{u_l}}$. We instead learn a new linear classifier, which tries to separate entities whose description contains at least one of the words $u_1, ..., u_l$ from the other entities. We then define $\mathbf{d_C}$ as the normal vector of the corresponding hyperplane. This was found to perform better, and it gives us a natural criterion to select entities that have a given feature to a sufficient extent, i.e. those that are classified as positive by this new classifier. For a cluster C, we write pos_C and neg_C for the set of positively and negatively classified entities. We will also refer to the clusters $C_1, ..., C_k$ as the primary feature clusters, and to the associated directions $\mathbf{d_C}$ as the primary feature directions. Intuitively, we can think of the primary features as facets. The

classifiers corresponding to the primary features thus partition the embedding space into different regions, which group the entities that correspond to these facets. Our aim will now be to identify the internal structure of each of these regions.

Learning sub-features. To find sub-features of a given primary feature p with associated cluster C_p and direction $\mathbf{d_p}$, we essentially re-apply the same method used for learning the primary features, but now only considering the set of entities in pos_{C_p}. In fact, we look for words that linearly separate those entities that are deemed to have the corresponding primary feature[2]. Specifically, for each word w from C_p, we train a linear classifier using only the entities in pos_{C_p}. As before, we use the Kappa score to select those words for which the associated classifier performs sufficiently well. The directions modelling the selected words are then again clustered using affinity propagation. We crucially rely on the fact that this does not require us to specify the number of clusters, as it would not be feasible to tune the number of sub-clusters for each primary feature. Figure 4.2 illustrates our approach.

Let $S_1, ..., S_{k_p}$ be the sub-feature clusters that were found for the primary feature p, and let $\mathbf{d^p_{S_1}}, ..., \mathbf{d^p_{S_{k_p}}}$ be the corresponding sub-feature directions. To define the feature-based representation of a given entity, we now associate one component with each primary feature and one component with each sub-feature. To compute the value corresponding to a given sub-feature for an entity e, we simply take the dot product $\mathbf{e} \cdot \mathbf{d^p_{S_i}}$, i.e. the feature values are computed in exactly the same way as for primary features. Note that while the features are discovered in a hierarchical way, the resulting feature-based representation is thus essentially flat. An alternative would be to combine $\mathbf{e} \cdot \mathbf{d^p_{S_i}}$ with $\mathbf{e} \cdot \mathbf{d_p}$ to define the value of the sub-feature. In initial experiments, however, this was found to perform poorly. One reason is that the classification of entities into pos_{C_p} and neg_{C_p} is not perfect. For instance, an organisation could be a borderline instance

[2]We also experimented with a variant in which the entities were weighted by the probabilities predicted by a logistic regression classifier. We then trained the sub-feature classifier using a weighted logistic loss, but noticed no consistent improvements.

of the category of political organisations, but still be a representative example of a left-wing organisation (e.g. a newspaper with a strong left-wing bias), even if in general we tend to think of *left-wing* as a sub-feature of *political*. Another reason is that by simply using the value $\mathbf{e} \cdot \mathbf{d}^{\mathbf{p}}_{\mathbf{S_i}}$, we minimize the redundancy between the information captured by the primary feature p and the information captured by this sub-feature.

Inspired by this latter view, we also consider a variant in which we require each sub-feature direction $\mathbf{d}^{\mathbf{p}}_{\mathbf{S_i}}$ to be orthogonal to the corresponding primary feature direction $\mathbf{d_p}$, as a way to directly impose the idea that primary features and sub-features should provide complementary information. To this end, we first obtain a sub-feature direction $\mathbf{d}^{\mathbf{p}}_{\mathbf{S_i}}$ as before, and then compute the orthogonal decomposition of this vector w.r.t. the vector $\mathbf{d_p}$. In particular, as the final sub-feature direction, we then use the following vector:

$$\tilde{\mathbf{d}}^{\mathbf{p}}_{\mathbf{S_i}} = \mathbf{d}^{\mathbf{p}}_{\mathbf{S_i}} - \left(\frac{\mathbf{d}^{\mathbf{p}}_{\mathbf{S_i}} \cdot \mathbf{d_p}}{\mathbf{d_p} \cdot \mathbf{d_p}} \right) \mathbf{d_p} \tag{4.1}$$

4.3 Experimental Analysis

4.3.1 Datasets

To test our method, we focus only on the relatively small domains, which are *Movies*, *Place Types*, *Organisations*, *Buildings*, and *Band* datasets. See Chapter 3 for more detail about the datasets.

4.3.2 Methods

We compare two versions of our method: the standard version (*Sub*) and the version where the orthogonal decomposition (4.1) is used (*Ortho*). We also consider three baselines. First, we use a model that only uses primary features (*Primary*). Second, we

use a model in which feature directions are randomly chosen (*Random*), by sampling each coordinate from a standard normal distribution (which after normalization is equivalent to sampling from a uniform distribution on the hypersphere). Note that while using random directions may seem naive, related methods such as using random projections for dimensionality reduction often perform surprisingly well [13]. As the third baseline, we use average-link Agglomerative Hierarchical Clustering (AHC) to cluster word directions instead of affinity propagation. To obtain a two-level clustering from the dendrogram, we tune distance cut-offs d_1 and d_2 to determine the set of primary clusters and their corresponding sub-clusters. Once the clusters are determined, we learn a corresponding cluster direction in the same way as how the cluster directions for primary features are learned with our method. We also experimented with Hierarchical LDA, but found it too slow to be used on our datasets.

Apart from how the feature directions are constructed, the overall number of features also has a strong impact on the result, where increasing the number of features increases the chance that one of them reflects a natural category (even if directions are chosen by chance), but at the risk of overfitting. For the random baseline, we directly tune the number of directions on a held-out development set, considering values from $\{100, 500, 1000, 1500, 2000, 2500\}$. We also verified that no further improvements were possible by choosing more than 2500 or fewer than 100 directions. For the methods which use affinity propagation, we can only influence the number of clusters indirectly, by changing the so-called preference parameter of this clustering algorithm. As is usual, this parameter is chosen relative to the median μ of the affinity scores. For the methods *Sub* and *Ortho*, we considered values from $\{0.7\mu, 0.9\mu, \mu, 1.1\mu, 1.3\mu\}$. For the *Primary* method, we considered a larger set of values to verify that no further improvements were possible: $\{0.5\mu, 0.7\mu, 0.9\mu, \mu, 1.1\mu, 1.3\mu, 1.5\mu\}$. In the case of AHC, to make the results as comparable as possible to those of our model, we tune the cut-offs d_1 and d_2 such that the number of selected clusters is as close as possible to the optimal numbers obtained with affinity propagation. To test whether better results are

possible with AHC when choosing a different number of clusters, we also evaluated a range of other cut-off values, but this did not lead to better results.

Entity embeddings. For each domain we used 100-dimensional vector spaces obtained using multi-dimensional scaling (MDS), following [36]. For the Buildings, Organisations, and Bands domains, we also used 100-dimensional vector spaces generated using Doc2Vec [81]. See Section 3.4 for more details.

4.3.3 Evaluation

At this point, there is no low-dimensional subspace for each facet; therefore, our focus is only to evaluate whether the hierarchical approach finds higher quality features than the method in [36]. In particular, to evaluate whether the discovered features are semantically meaningful, we test how similar they are to natural categories, by training depth-1 decision trees (meaning that only a single feature can be used for prediction) and depth-3 decision trees on our feature-based representations, following [1]. Training an interpretable classifier such as a decision tree is a natural way to test whether the features in the obtained feature-based representation are semantically meaningful. The performance of a shallow decision tree is directly related to the quality of the learned features. Given that a depth-1 decision tree can only use one of these features, the performance of such a decision tree essentially tells us to what extent the classes that are considered in the supervised classification task have been discovered as features. For instance, in the Movie domain, we should expect to see common movie genres among the features. Depth-1 decision trees should thus be able to predict these genres well. The performance of depth-3 decision trees tells us how well natural categories can be characterized using a small set of features (see Figures 4.4, and 4.5).

4.3.4 Methodology

In Section 3.3 we explained how the datasets are divided into training and testing splits. We report the results in terms of F1 score. To obtain the feature directions, we used logistic regression and only considered words for which the corresponding Kappa score is at least 0.3. To reduce the computation time, for datasets where this led to more than 5000 features, only the 5000 top-scoring words are retained. When learning directions for the sub-features, we use a lower Kappa score of 0.1, as the corresponding classification problems are often harder (given that sub-features are often about subtle nuances of the primary feature) and the set of candidate words is smaller.

		Feat.	Random	AHC	Primary	Sub	Ortho
Place Types	Foursquare	D1	0.39±0.01	0.36	0.36	0.43	**0.45**
		D3	0.50±0.01	0.46	0.48	0.54	**0.57**
	Geonames	D1	0.23±0.02	0.22	0.24	0.20	**0.28**
		D3	0.27±0.02	0.29	0.27	0.32	**0.34**
	OpenCYC	D1	0.28±0.01	0.29	0.29	**0.31**	0.30
		D3	0.32±0.01	0.33	0.31	**0.35**	**0.35**
Movies	Keywords	D1	0.24±0.001	**0.26**	**0.26**	0.25	**0.26**
		D3	0.26±0.001	0.27	0.27	**0.28**	**0.28**
	Genres	D1	0.36±0.005	0.38	0.36	**0.43**	0.41
		D3	0.40±0.02	0.43	0.42	0.44	**0.45**
	Ratings	D1	0.44±0.01	0.44	0.45	**0.48**	0.47
		D3	0.46±0.01	0.46	0.47	**0.50**	0.49
Bands	Genres	D1	0.10±0.003	0.16	0.16	**0.17**	0.15
		D3	0.11±0.004	0.14	0.15	**0.16**	0.15
	Country of origin	D1	0.29±0.02	0.33	0.34	**0.40**	0.38
		D3	0.28±0.02	0.33	0.33	**0.43**	0.39
	Location of formation	D1	0.13±0.01	0.15	0.14	**0.17**	**0.17**
		D3	0.14±0.01	0.16	0.14	0.16	**0.19**
Org.	Country	D1	0.40±0.01	0.50	0.67	0.66	0.67
		D3	0.44±0.01	0.57	0.65	**0.71**	0.69
	Headquarter location	D1	0.17±0.01	**0.23**	0.21	**0.23**	0.22
		D3	0.18±0.01	0.26	0.23	**0.27**	0.25
Buildings	Country	D1	0.53±0.03	0.72	0.74	0.74	0.74
		D3	0.60±0.02	0.75	0.80	**0.81**	0.80
	Administrative Location	D1	0.27±0.03	0.33	0.37	**0.49**	0.46
		D3	0.29±0.03	0.30	0.31	**0.45**	0.37

Table 4.1: Performance in terms of F1 score for depth 1 (D1) and 3 (D3) decision trees, considering embeddings learned using MDS .

		Feat.	Random	AHC	Primary	Sub	Ortho
Bands	Genres	D1	0.08±0.004	0.09	0.08	**0.09**	0.09
		D3	0.9±0.00	**0.10**	0.09	0.09	0.09
	Country of origin	D1	0.23±0.01	0.22	0.23	**0.24**	**0.24**
		D3	0.23±0.01	0.21	0.24	0.24	**0.26**
	Location of formation	D1	0.1±0.003	0.10	0.09	**0.11**	0.11
		D3	0.11±0.002	**0.12**	0.10	0.11	0.10
Organisations	Country	D1	0.34±0.01	0.28	0.30	**0.39**	0.37
		D3	0.38±0.02	0.28	0.31	**0.42**	0.42
	Headquarter location	D1	0.16±0.02	**0.22**	0.17	**0.22**	**0.22**
		D3	0.18±0.02	0.19	0.19	**0.23**	0.21
Buildings	Country	D1	0.54 ±0.03	0.55	0.56	**0.57**	**0.57**
		D3	0.55±0.01	0.58	0.51	**0.64**	0.60
	Administrative location	D1	0.14±0.01	0.15	0.11	**0.17**	0.16
		D3	0.15±0.02	0.13	0.12	**0.16**	**0.16**

Table 4.2: Performance in terms of F1 score for depth 1 (D1) and 3 (D3) decision trees, with embeddings learned using Doc2Vec.

Genre	**Primary**	**Sub**	**Diff**	**Genre**	**Primary**	**Sub**	**Diff**
Action	0.47	**0.48**	0.01	Musical	0.17	**0.42**	0.25
Horror	0.60	**0.72**	0.12	War	0.36	**0.49**	0.13
Comedy	0.64	**0.68**	0.04	Western	0.44	**0.46**	0.02
Drama	0.68	**0.74**	0.06	Thriller	0.53	**0.63**	0.10
Animation	0.24	**0.4**	0.16	Mystery	**0.29**	0.28	0.01
Adventure	0.31	**0.36**	0.05	Crime	0.46	**0.53**	0.07
Biography	**0.2**	0.16	0.04	Romance	0.45	**0.46**	0.01
Film-Noir	0.06	**0.09**	0.03	Sci-Fi	0.44	**0.62**	0.18
Family	0.35	**0.37**	0.02	Short	0.10	**0.11**	0.01
Fantasy	0.21	**0.26**	0.05	Sport	**0.37**	**0.37**	0.00
Music	0.36	**0.54**	0.18	History	0.20	**0.30**	0.10
Document.	0.33	**0.47**	0.14				

Table 4.3: Breakdown of performance for movie genres.

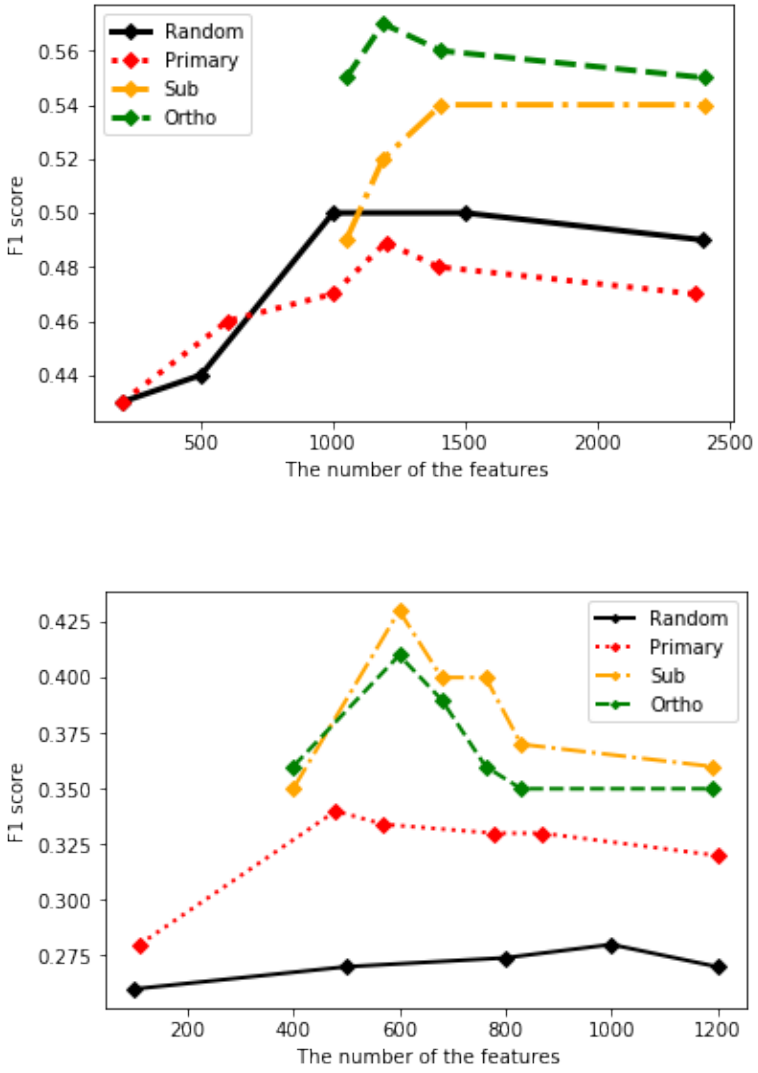


Figure 4.3: Effect of using more features for the depth-3 decision trees on the Foursquare classification task from the Place Types domain (top) and the country of origin task from the Bands domain (bottom).

4.3.5 Results

The results are summarized in Tables 4.1 and 4.2. We can see that our *Sub* method clearly outperforms *Primary*. In fact, there are several cases where the performance of *Primary* is comparable with that of *Random*, yet where our *Sub* method performs substantially better (e.g. the depth-1 results for the genres in the Movies domain). While the sub-features never perform clearly worse than the primary features, there are some cases where both approaches perform similarly (e.g. the genre attribute in the Bands domain). This can be expected when the considered attribute is sufficiently dominant, in which case most or all of the attribute values might correspond to primary features. Interestingly, however, for the Movies domain, many of the considered genres were only modelled well as sub-features. This is illustrated in Table 4.3, which shows the performance of the *primary* and *sub* methods for the genres from the movies dataset. The *Ortho* method generally performs well, but slightly worse than *Sub*. Interestingly, however, for the Place Types, the *Ortho* method performs best overall. This dataset is rather noisy, and taking the orthogonal complement in this case seems to help with preventing overfitting. AHC fails to consistently outperform the other baselines, and is clearly worse than our method. This shows that the improvements obtained by our model are not due to the fact that we impose a hierarchical structure on the features as such, but rather due to specific way in which we learn the directions of the sub-features.

Comparing the MDS and Doc2Vec embeddings, we generally see the same trends, although the results for Doc2Vec are consistently worse. To better understand the impact of the number of features, Figure 4.3 shows the performance of the different methods in function of the total number of features. As can be seen, while the number of features clearly matters, the improvement that we observed for the sub-features is remarkably robust.

Movies	
Primary	**Sub-features**
[delightful, cute]	[romantic, chemistry, romance], [fantasy, charisma], [disneys],[childish, adventures, fare], [musicals, dance], [magical, magic, enchanting], [heartwarming, sentimental, warm]
[gore, gory]	[bloody, zombie, massacre], [killings, serial, maniac], [hardcore, carnage, gratuitous], [supernatural]
[sci, science]	[creatures, humans, eaten], [franchise, sequel, sequels], [cgi, technology, computer, graphics], [mythology, kingdom, ancient], [doctor, blast, mindless, brain], [scifi, futuristic, outer, aliens], [animation, animators, pixar], [scientist, science, scientific]
[documentary, interviews]	[athletes], [musicians, concert, concerts, albums], [biography], [individuals, perspective, focuses, individual], [educational], [inspiring, awe, captured, appreciation], [artists, artist], [facts, research, account, thousands], [recording, recordings]
Place Types	
Primary	**Sub-features**
[streetlights, lamppost]	[scaffold, railings, embankment], [mailbox, hydrant] [lampposts, streetlamps], [streetscape,crosswalk] [paving, cobblestone, cobblestones]
[neon, advertisement]	[advertisements, marketing, ads], [stickers, sticker] [lettering, typography, logo]
[derelict, ruined]	[exploring, exploration], [relic, ruin, weathered] [desolate, bleak, deserted],[decay]
[xmas, crafts]	[hearts, ornaments, beads], [vase, bouquet, candles]

Table 4.4: Examples of primary and sub features for Movies and Place Types domains. Clusters of words that define a feature are grouped using [...].

Buildings	
Primary	**Sub-features**
[site, hotel]	[office, house, facility, headquarters, post, tenant, administrative, plaza], [center, block, corner, district, square, parking, entrances]
[company, complex]	[facilities, conference, staff, operations, employees, electronic, spa, auditorium, emphasis, network, gymnasium, plant], [industrial, production, subsidiary, chain], [nationwide] [lobby, halls, sqft, luxury, quad], [flagship]
[side, west]	[central, northern, western, southern, secondary], [side, west, north, south, east], [island], [terrain] [borders], [wilderness], [junction, roads, rural]
[style, architecture]	[example, designs, elements, motifs, neo-classical, sense, geometric], [viewer]
Organisations	
Primary	**Sub-features**
[political, peace]	[party, foreign, democracy, reform, terrorism, liberal democratic, constitutional, parliamentary, politicians, debate, left-wing, agenda, sovereignty, regime, electoral, immigration, capitalism, non-partisan], [peace, peaceful, disarmament], [unionists],[freedoms]
[movement, activists]	[protest, activist, nonviolent, activism, demonstrations] [justice, equality, movement, radical, suffrage, discrimination, union, movements, anti-war, revolution, racism, unity, resistance, ideology, socialism, progressive, march, struggle, stance, factions, solidarity], [anarchists] [feminist, feminism],[pacifist, pacifism], [pacifists]

Table 4.5: Examples of primary and sub features for Buildings and Organisations domains. Clusters of words that define a feature are grouped using [...].

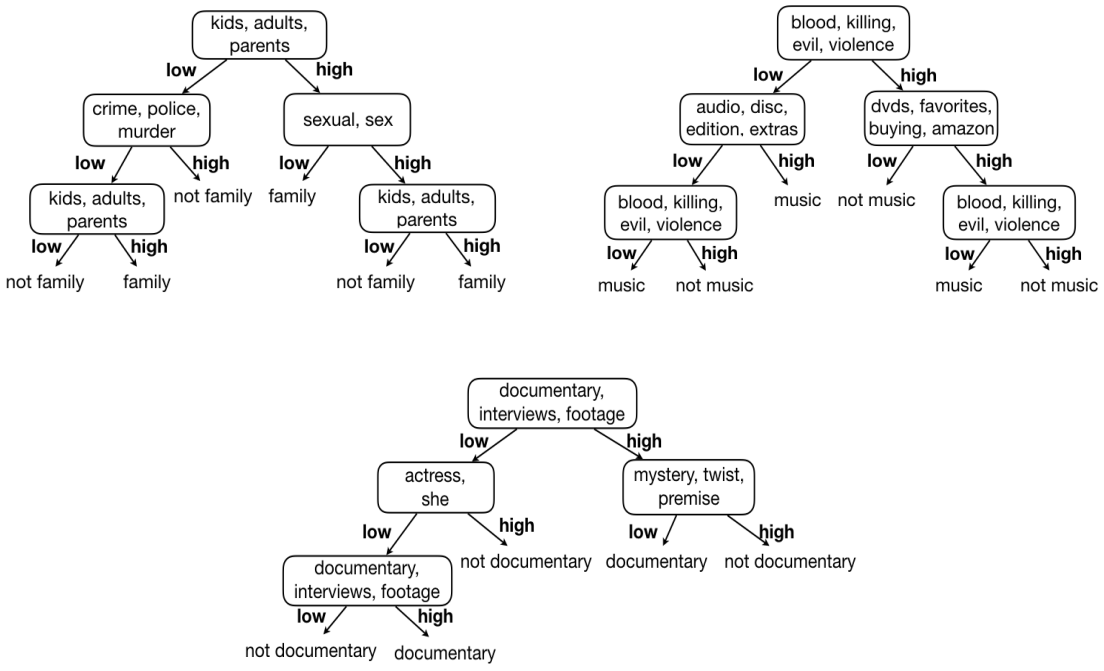


Figure 4.4: Example of depth-3 decision trees for the *family*, *music*, and *documentary* genres in the Movies domain, using Primary features only .

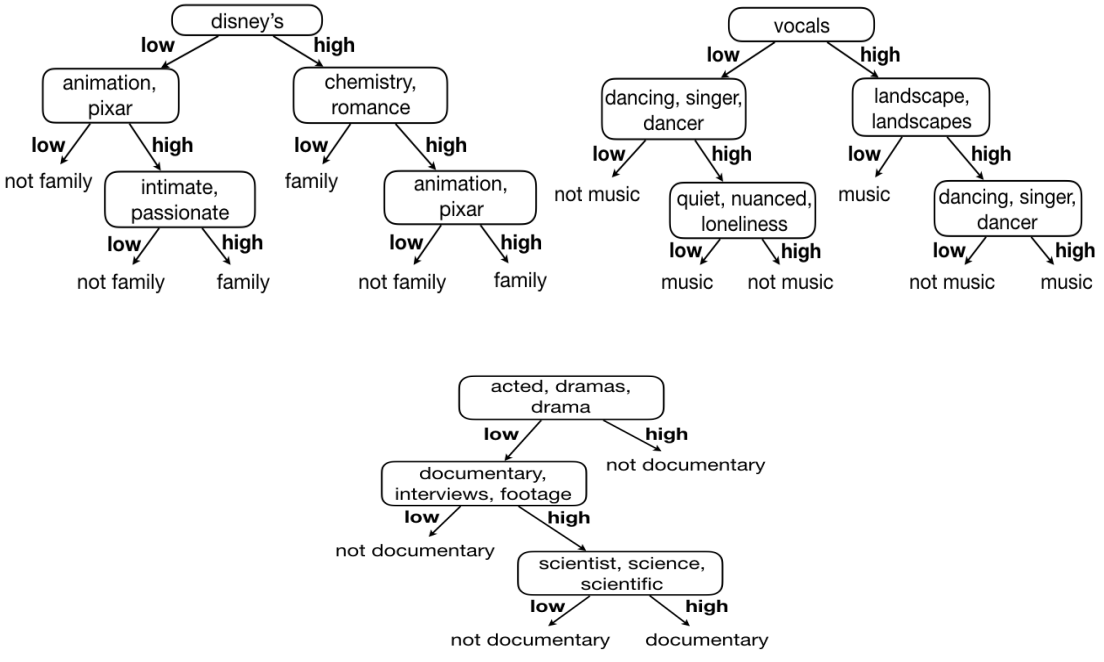


Figure 4.5: Example of depth-3 decision trees for the *family*, *music*, and *documentary* genres in the Movies domain when Sub-features are used.

4.3.6 Qualitative analysis.

One immediate advantage of learning quality dimensions is that we can use them to learn interpretable classifiers. Figures 4.4 and 4.5 illustrate this, respectively for the *Primary* baseline and for our approach with sub-features. They show decision trees for predicting the movie genres *family*, *music* and *documentary*. For most features, only a subset of the words from the corresponding cluster is shown. The exact thresholds that were chosen by the decision tree learner are also omitted. When looking at Figure 4.4, while it is easy to interpret which patterns the decision tree learner has uncovered, we can also see that the features that have been learned are not sufficiently fine-grained. This leads to decision trees in which the most informative feature (e.g. [kids, adults, parents] for the family genre) is repeated several times, albeit with different thresholds. In contrast, in Figure 4.5 more fine-grained features are included. Note that these fine-grained features cannot be found by simply learning more primary features (cf. Figure 4.3). For example, for the family genre, we see two features that specifically relate to children's movies ([disney's] and [animation, pixar]). Note that these features induce rankings which will be meaningful in identifying children's movies beyond those that were developed by Disney and Pixar (and even beyond animated movies). For the music genre, the classifier based on primary features does not include any highly relevant features. In contrast, the corresponding decision tree in Figure 4.5 can take advantage of the highly informative features [vocals] and [dancing,singer,dancer]. Finally, for the documentary genre, the decision tree with sub-features captures the fact that such movies do not involve acting, which turned out to be a highly relevant sub-feature. As far as the primary features are concerned, the only more or less similar feature is [actress,shes], which intuitively corresponds to movies with a female lead actress. Note how these decision trees can give us some clear insights into the quality of the learned models. For instance, the tree for documentary in Figure 4.5 suggests that documentaries are about science, or related subjects, which points to a kind of overfitting.

It is clear what kind of properties about the considered genres the learned models have uncovered. Such examples show that data-driven conceptual spaces, while clearly being messier than the idealised representations considered by Gärdenfors [48], can indeed bridge between vector space and symbolic representations in a useful way.

Table 4.4 shows some illustrative examples of features that were found for the Movies and Place-Types domains. As can be seen, sub-features can play a number of different roles. For the primary feature [delightful, cute], the sub-features correspond to different kinds of movies that are often described as delightful or cute. These comprise a rather diverse set of movies, encompassing romantic, musicals and children's movies, among others. Without considering sub-features, movies from these various genres would have a very similar representation, which is clearly undesirable. By identifying sub-features, we are able to differentiate between these genres. The primary feature [gore, gory], on the other hand, corresponds to a more coherent movie genre. The sub-features in this case essentially correspond to sub-genres, such as zombie and serial killers movies. For [sci, science] as primary feature, we see the sub-feature [cgi, technology, computer, graphics] as an example of a quality dimension that mostly makes sense within the scope of sci-fi movies. A similar example is [educational] which is found as a sub-feature of [documentary, interviews]. For the space of Place Types, the features and sub-features reflect how people describe photographs of a given place. For example, the sub-features of [neon, advertisement] reflect that advertisements could be remarkable because of their content, as captured by [advertisements, marketing, ads], or by their unique style, as captured by [lettering, typography, logo]. Furthermore, Table 4.5 shows some examples from the Buildings and Organisations domains, which are noisier than the Movies and Place types domains. The sub-features for the primary feature [site, hotel] are types of buildings [office, house, facility, ...] and quality dimensions that describe the location [center, block, corner, district, square, ...]. For the primary feature [complex, company], the sub-features describe company buildings in terms of the people, facilities, and the type of the business. For Organisations, the sub-features of [political, peace], for example, are mostly features strongly related to

the political facet. Similarly for [movement, activists] where the sub-features describe the purposes of movements, e.g. anti-war, feminist, racism and how these movements express their beliefs, e.g. protest, nonviolent.

Regarding facets, the examples in Table 4.4 and 4.5 also illustrate the hybrid nature of the found features, where some of them intuitively correspond to more or less well-defined facets while others intuitively correspond to quality dimensions. As far as the primary features are concerned, we would expect to see mostly features of the former kind, but as the example [delightful, cute] illustrates, that is not always the case. In fact, most of the primary features describe part of a facet rather than a complete facet; for example, we would expect to find a cluster that has different types of genre but we find primary features that either describe sub-genres, such as [documentary, interviews], or quality dimensions related to certain types of genre, e.g. [delightful, cute]. Another example is [style, architecture] in Table 4.5, which we would expect to have features describing different types of style (e.g. historical, classical, modern) rather than having more or less thematic cluster. Similarly, the primary feature [side, west] in Table 4.5 the features describe a thematic, yet noisy, facet as we would expect to find the features [rural, island] in a separate facet with terms like [city, town, village, ...]. For Organisation, very few examples are related to well-defined non-thematic facets such as [movement, activists]. For the Bands datasets, we could not find good qualitative examples. This because it is a very noisy dataset. Therefore, we will exclude it from the next experiments.

These examples indicate that clustering the directions will mostly lead to either thematic facets or incomplete facets. This strongly suggests that to find non-thematic facets, e.g. different types of architectural styles, drastic changes are needed in terms of how we use the directions and what type of clustering algorithm we use, and more importantly how to model the facets. For non-thematic facets, mainly, separating the space (as we proposed in this chapter) will not always work, especially for inclusive facets, i.e. the facets that are relevant for most of the entities. To illustrate, if we have a genre

facet, which should group features like [drama, comedy, horror, documentary, ...] it will be hard to find a hyperplane that separates the movies that have at least one of these properties from the ones that do not, as each movie corresponds to at least one genre.

4.4 Summary

One of the most appealing properties of the conceptual spaces framework is the fact that entities are represented in terms of their salient semantic features. Such feature-based representations can to some extent be derived from learned entity embeddings as well, by finding semantically meaningful directions. However, this is complicated by the fact that many features only make sense for particular subsets of the entities, making existing methods sub-optimal. In this chapter, we showed that this issue can be mitigated by learning semantically meaningful directions in a hierarchical way. Essentially, some of the directions that are identified serve the purpose of splitting up the vector space into different facets, while other directions are more similar in spirit to the quality dimensions from the framework of conceptual spaces.

However, despite the improvement in classification results, we found this approach to be more suitable for finding thematic facets only. In principle, it will be hard to split the vector space implicitly into non-thematic, inclusive, facets such as the genre facet in the Movies domain. In many cases, we found that some of the obtained primary features are actually sub-facets of a non-thematic facet, i.e. incomplete facets. In the next chapters we will try to find facets that could be thematic or non-thematic; this will be very challenging as the noisy nature of data-driven representations means that the difference between the directions that define facets and directions that correspond to quality dimensions is not always clear-cut (i.e. most facets are vague, and graded membership degrees can often equivalently be seen as semantic features). Therefore, in the next chapter we will use some sort of external knowledge to find the facets, then

explicitly decompose the vector space into low-dimensional facets.

Chapter 5

Learning Conceptual Spaces with Disentangled Facets

5.1 Introduction

In the previous chapter, we attempted to implicitly divide an entity embedding into facets. Despite the encouraging quantitative results, most of the facets produced are not sufficiently inclusive, i.e. they are incomplete facets. For example, in the Movies dataset, we have multiple primary features that describe the genre facet; this is because the directions of the features that describe different genres are not similar, and it is therefore hard to group them together in one facet. Moreover, the way the previous method learned the features of each facet (sub-features) using only the facet part of the space is not suitable for non-thematic and inclusive facets (such as the genre facet), where most movies have at least one type of genre. Another limitation is that these facets are not associated with low-dimensional subspaces. Having low-dimensional facets is a crucial part of learning a conceptual space, as these subspaces provide the flexibility to model the different facets of similarity. Another advantage of low-dimensional facet-specific spaces is that learning from few examples should be easier than in high-dimensional spaces.

Therefore, in this chapter, we will try to find comprehensive and varied collections of facets and represent them as low-dimensional subspaces. To do this we explore a

strategy for decomposing a given vector space embedding into separate facet spaces. In particular, we first determine which interpretable features are modeled by the vector space. Then we cluster the word vectors corresponding to those features. Despite being intuitive, given that word embeddings are known to group together functionally similar words, we found this strategy to perform poorly in its basic form. First, simply looking for clusters in word embeddings often leads to thematic clusters, e.g. grouping *horror* together with words such as *scary* and *zombie* rather than other genres such as *western* and *drama*. To address this, we explicitly prevent two words from ending up in the same cluster if the features they are modeling (i.e. the feature directions) are too similar. In particular, in order to obtain non-thematic clusters, we need to encourage the clustering algorithm to group features that have dissimilar directions but whose corresponding labels have similar word embeddings. Second, in most domains, there are one or two central facets that tend to be highly correlated with most of the other facets (e.g. genre in the movie domain). To ensure that the resulting facet spaces are sufficiently different (rather than capturing minor variations of the most central facets), we found it useful to use an iterative approach, where previously found facets are "removed" from the vector space embedding before proceeding to find further facets. With these two modifications, we find that useful facets can indeed be found, which consistently lead to better classification performance compared to the original vector space embedding.

The rest of this chapter is organized as follows: The description of the proposed model, including how we group the directions into facets, and how we model these facets as subspaces is in Section 5.2. Section 5.3 gives details about the datasets, experiments, evaluation, results, and qualitative analysis. Finally, our findings are summarized in Section 5.4.

5.2 Decomposing Conceptual Spaces

Similar to chapter 4, we used the method proposed in Derrac and Schockaert [36] to finds a set of features F which can be modelled as directions in the given vector space. The directions are obtained by training a logistic regression classifier to predict whether a candidate feature appears in the entity description or not (see Section 3.5). Our hypothesis is that we can group these features into meaningful facets and that we can represent these facets as subspaces of the given vector space embedding. In other words, the desire is to decompose a high-dimensional vector space embedding into several low-dimensional subspaces each of which capture a certain facet of similarity. Section 5.2.1 discusses our approach for finding these subspaces.

5.2.1 Finding Facets

Our aim is to group the features from F into meaningful facets. For instance, in the movies domain, we might expect to see facets corresponding to e.g. genre, language and release date. It does not seem possible (nor desirable) to formally define what constitutes a good facet, which is similar to the difficulty of evaluating unsupervised learning methods more generally, e.g. defining what makes a good cluster. Intuitively, however, a facet should group features which are of the same kind (e.g. genres) and should in some sense be exhaustive (i.e. all genres, rather than a set of features that refer to one or a few particular genres). In Chapter 4, we only were able to obtain thematic facets, and we found that finding exhaustive facets is challenging; therefore, our focus in this chapter is to find such facets.

Using subspace clustering. The aim of subspace clustering is to decompose a high-dimensional space into the union of lower-dimensional spaces. One may wonder whether we can learn useful facets by applying subspace clustering to the feature directions $\mathbf{d_f}$. Unfortunately, in our initial experiments, this approach did not prove successful. This is illustrated for the movies domain in Table 5.1, where we used the state-of-the-art SSC-OMP subspace clustering method [148]. For this comparison, we first manually grouped the features from F to obtain a gold standard. We only used the top 500 features, based on the Kappa score, we obtained with the method from Derrac and Schockaert [36] to create the gold standard and focused mostly on non-thematic facets. From these 500 features, we manually grouped all the features that are related to genre, such as *horror*, *comedy* and *drama* in one facet and all the features that are related to the media such as *cgi*, *dvds* and *vhs* in another facet. Moreover, when creating the gold standard genre facet, sentiments that are usually associated with certain genre, such as (*horror*, *scary*) were not included, and as a result a well defined gold standard genre facet was obtained. The first column of the table shows two of the resulting facets: one corresponding to genres and one corresponding to different media types (which indirectly captures the time period during which a movie was released). The right-most column shows the closest facets that were found with SSC-OMP. As can be seen, these facets are largely non-sensical. For instance, in the first case, words such as *blu* and *disc* are clustered together with semantically unrelated words such as *fighting*, *england* and *accurate*. In the second example, genres such as *horror* and *thriller* are grouped together with unrelated words such as *cheesy*, *widescreen* and *award*. This negative result seems in accordance to the findings from Locatello et al. [94] that unsupervised disentangled representation learning (DRL), i.e. learning a latent vector representation where groups of the dimensions of the latent codes correspond to meaningful interpretable features, seems impossible without a strong inductive bias or supervision signal. Moreover, Locatello et al. [94] conclude, based on empirical investigations on six unsupervised DRL methods, that these methods are not reliable as they are highly sensitive to initializations and hyperparameters. This means that we

need a strong inductive bias to find non-thematic facets and learn disentangled representation for each one. We also tried several other subspace clustering methods, for a wide range of different configurations, without obtaining better results. Similarly, we experimented with neural approaches for learning disentangled representations directly from the bag-of-words representations of the entities, but again unsuccessfully.

Using word embeddings. These negative results strongly suggest that some kind of external knowledge is needed to find meaningful facets. To this end, we focus on the use of word embeddings, which seems natural given the fact that words of the same kind (e.g. different names of genres) tend to be used in similar contexts, and can thus be expected to have similar word vectors. In particular, our basic approach for identifying facets consists in clustering the word vectors, from some standard pre-trained word embedding model, corresponding to the features in F. One important drawback of this basic strategy, however, is that it often leads to thematic clusters. For instance, while we would want *horror* to be clustered together with other names of genres, when simply clustering word vectors without any further guidance, *horror* may be clustered together with thematically similar words such as *scary* and *zombie*. To allow us to discover non-thematic facets, we rely on the insight that if a and b are thematically similar words (e.g. *horror* and *zombie*) then the corresponding feature directions $\mathbf{d_a}$ and $\mathbf{d_b}$ will also be similar. However, for paradigmatically similar words, such as *horror* and *comedy*, this should not be the case. In other words, two words should intuitively end up in the same clusters if they have similar word vectors but dissimilar feature directions.

While there are many ways to implement this intuition, we found that using the cosine similarity between $\mathbf{d_a}$ and $\mathbf{d_b}$ was not always reliable. In our earlier experiments, we could not obtain non-thematic facets by directly grouping words that have high cosine similarity in word embedding and their corresponding directions have low cosine similarities. Instead we rely on the following measure of overlap between the sets pos_a

and pos_b:

$$o(a,b) = \min\left(\frac{|pos_a \cap pos_b|}{|pos_a|}, \frac{|pos_a \cap pos_b|}{|pos_b|}\right) \tag{5.1}$$

where pos_a and pos_b are the sets of entities which are classified as positive by the linear classifier corresponding to word *a* and word *b*, respectively.

The dissimilarity between features a and b from F is then defined as follows:

$$d(a,b) = \begin{cases} 1 - \cos(\mathbf{w}_a, \mathbf{w}_b) & \text{if } o(a,b) \leq q \\ 1 & \text{otherwise} \end{cases} \tag{5.2}$$

where the overlap threshold q is a hyperparameter and $\mathbf{w}_f$ denotes the word vector for feature f. This means that words such as *horror* and *comedy* are supposed to have a low dissimilarity score, because the overlap between the movies that have these features will be low, i.e. $o(horror, comedy) \leq q$, while these words have similar word vectors, meaning that $\cos(\mathbf{w}_{horror}, \mathbf{w}_{comedy})$ is very high. As a result, the dissimilarity score $d(horror, comedy)$ will be low, which will encourage the clustering algorithm to group these two words in one cluster.

The aim of the clustering step is to find a number of disjoint subsets of F, each of which intuitively corresponds to a facet. We will denote these subsets, candidate facets, by $X_1, ..., X_k$. To avoid finding redundant facets, we identify them in an incremental fashion. In particular, from the clusters obtained by the clustering algorithm, we only select the single most important one, i.e. the one which is most likely to describe a salient facet. For this purpose, we rank clusters according to the following score:

$$score(X_i) = |\bigcup_{f \in X_i} pos_f| \tag{5.3}$$

This score reflects the intuition that we prefer clusters with features that are general and diverse, i.e. such that most of the entities would have at least one of the features from the cluster. As will be explained below, after the subspace corresponding to this facet has been determined, we iteratively apply the same method on a reduced vector space to find the next most important facet, until the desired number of facets k has been found.

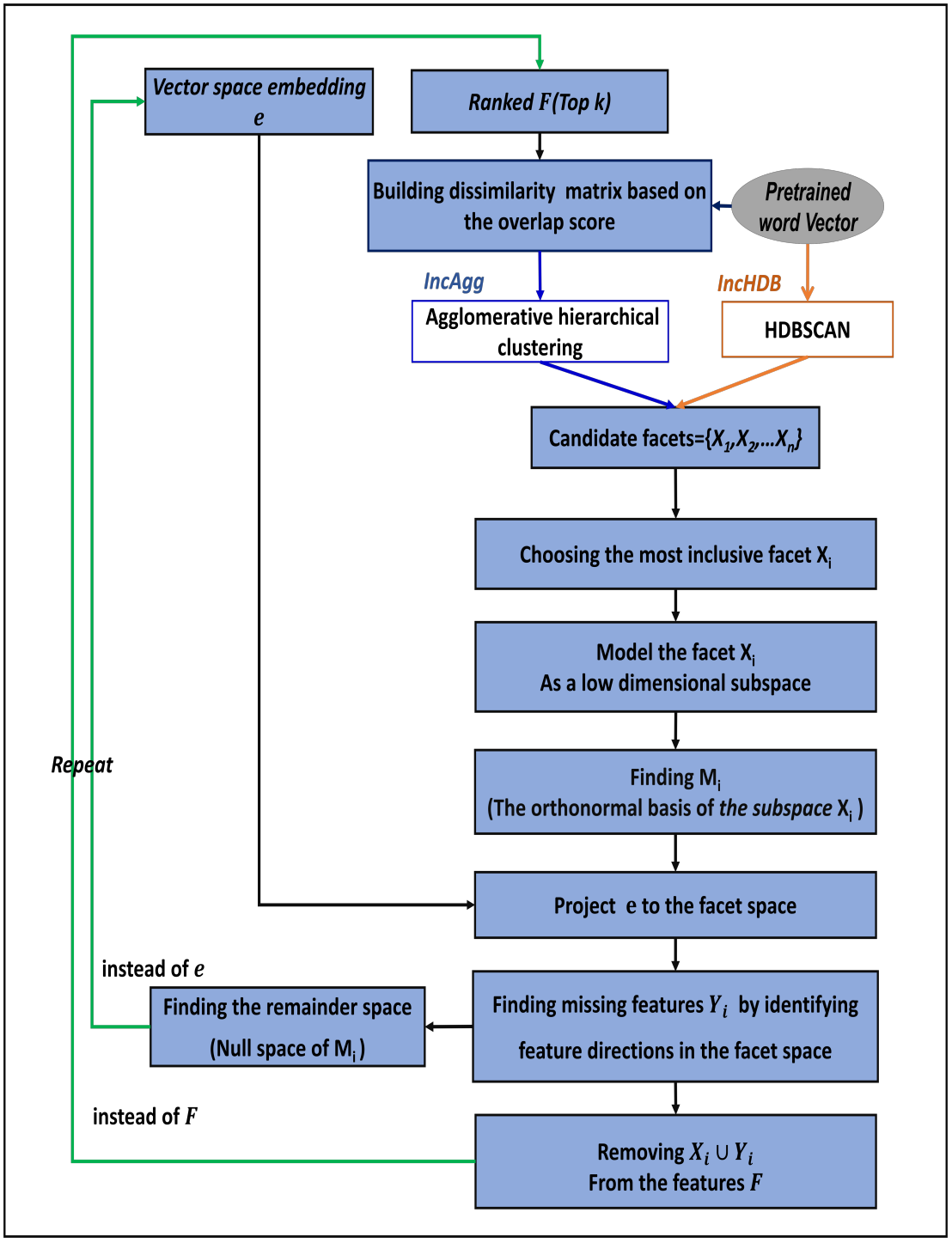


Figure 5.1: Decomposing vector space embedding into facet-specific subspaces

5.2.2 Modelling Facets as Subspaces

We model each facet X_i as a linear subspace of the given vector space embedding. To find this subspace, we learn new feature directions $\mathbf{c_f}$ for each $f \in X_i$, which still capture these features but lie in a low-dimensional subspace. In particular, we minimize the following objective:

$$\sum_{e \in E} \sum_{f \in X_i} \log \sigma(\mathbf{c_f e} + b_f) \tag{5.4}$$

where

$$\mathbf{c_f} = \lambda_1^f \mathbf{a_1^i} + \lambda_2^f \mathbf{a_2^i} + ... + \lambda_r^f \mathbf{a_r^i} \tag{5.5}$$

with r the desired number of dimensions of the subspace. Note that (5.4) essentially expresses that for each $f \in X_i$, we want to train a logistic regression classifier with coefficient vector $\mathbf{c_f}$. However, as expressed in (5.5), rather than learning these coefficient vectors independently, they are constrained such that they can be written as a linear combination of the vectors $\mathbf{a_1^i}, ..., \mathbf{a_r^i}$, which will be learned by the logistic regression classifiers. The logistic regression classifiers for all the features in X_i are trained simultaneously to force these features to lie in the same subspace. As a result, for the features $\{f_1, f_2..., f_s\}$ that belong to X_i there are corresponding directions $\{\mathbf{c_{f_1}}, \mathbf{c_{f_2}}..., \mathbf{c_{f_s}}\}$. Each one of the these directions can be written as a linear combination of $\mathbf{a_1^i}, ..., \mathbf{a_r^i}$. The resulting feature directions thus span a subspace of (at most) r dimensions. The vectors $\mathbf{a_1^i}, ..., \mathbf{a_r^i}$, which are not orthogonal, are then used to obtain the orthonormal basis for this subspace X_i. The orthonormal basis will then allow us to project the entities to the subspace X_i. Let $\mathbf{M}_i \in \mathbb{R}^{r \times n}$ be an orthonormal basis for this subspace. Then the r-dimensional facet-specific embedding of entity e will be obtained as follows:

$$\mathbf{e}^i = \mathbf{M}_i \mathbf{e} \tag{5.6}$$

There may be some features from F which are not contained in X_i but can nonetheless be modelled well in the resulting subspace (i.e. if they are semantically related to the

features in X_i). To identify these features, we apply the method explained in Section 3.5 to the facet-specific embeddings. For each of the features, a logistic regression classifier is trained to predict whether the features occurred in the BoW of the entities or not using the r-dimensional facet-specific embedding of entities. If a feature is strongly correlated to this facet, it is expected that the classifier of this feature will have a high Kappa score. We write Y_i to denote the features that were thus identified, beyond the ones from X_i (i.e. the features with the highest Kappa score in the i-th subspace).

Next we determine the null space of the basis $\mathbf{M}_i$, i.e. an $(n - r) \times n$ dimensional matrix $\mathbf{R}_i$ satisfying

$$\mathbf{M}_i \mathbf{R}_i^T = \mathbf{0}$$

This matrix $\mathbf{R}_i$ is a basis for the orthogonal complement of the subspace spanned by $\mathbf{M}_i$. Intuitively, it defines what remains of the vector space embedding after we remove (i.e. project away) the subspace modelling the facet X_i. This step is crucial to obtain a different facet each time rather than finding different variants of the dominant facet.

To find the remaining facets, we repeat the same procedure, but with two changes. First, the $n - r$ dimensional remainder space is used instead of the original embedding space, i.e. we use $\mathbf{R}_i\mathbf{e}$ as the vector representation of e. Second, the features in $X_i \cup Y_i$ are no longer considered by the clustering algorithm. This process is repeated until the desired number of facets has been found, each time considering an increasingly lower-dimensional remainder space and clustering only those features that are not already modelled in a previously identified facet. To illustrate, in order to decompose a 100-dimensional space into 10-dimensional facets, the first facet will be learned using the 100-dimensional space, and after that facet is obtained, the dimensions of that facet will be removed from the full space. Next, the second facets will be learned using the remaining 90-dimensional space. Intuitively, by learning the facets in this incremental way, we should be able to avoid finding multiple variants of the same facets.

The middle column of Table 5.1 shows two of the facets that were found with this

approach. Intuitively, these facets are clearly more meaningful than those that were found with SSC-OMP. Note that the top additional features Y_i allow us to find near-synonyms of features in the clusters, but most importantly helped in finding missing features. For example, in the media facet in the first row, features such as (amazon, ray, cgi) are missing in the initial cluster but are found in Y_i.

MOVIES		
Gold standard	**IncHDB**	**SSC-OMP**
blu, ray, cgi, dolby, surround, computer, technology, theaters, theatre, ordered, purchase, dvds, amazon, copy, audio, disc, edition, widescreen, transfer, digital, print, vhs, discs	**Initial cluster X_i:** audio, disc, dvds, digital, vhs, dolby, technology, discs, computer, version **Top additional features Y_i:** transfer, edition, blu, cgi, ray, widescreen, amazon, extras, awesome, computer, purchase, buying, surround, price, trailer, included, favorites, theaters, alot, previews, extra, player	blu, arts, disc, edition, ordered, crime, british, creepy, disturbing, fighting, charm, rent, reviewers, oscar, excited, questions, england, education, victims, packed, marriage, tense, detail, hell, culture, situation, accurate, trailers
thriller, comedic, comedy, documentary, comic, satire, documentaries, drama, melodrama, horror,action, adults, animation, crime, fantasy,family, musical, mystery, romance, war, western	**Initial cluster X_i:** thriller, comedic, comedy, documentary, comic, satire, documentaries, humor, humour, cheesy, adaptation, wit, melodrama, campy, parody **Top additional features Y_i:** hilarious, gags, laughs, jokes, slapstick, funniest, thrillers, funnier, suspense, witty, unfunny, amusing, suspenseful, historical, horror, romance, interviews, psychological	horror, thriller, political, charming, funnier, slapstick, documentaries, hilarious, killed, seat, issue, cheesy, gory, mystery, effects, amazon, widescreen, transfer, realistic, relationship, monster, epic, portrayed, glad, premise, hearing, evil, car, formula, decision, violent, villain, gun, goofy, game, teens, garbage, humor, ruin, product, amount, dad, loving, personality, award, folks

Table 5.1: Comparison of learned facets with gold standard for the movies domain.

5.3 Experimental Analysis

5.3.1 Datasets

We have carried out experiments with vector space embeddings for four different domains: *Movies*, *Place Type*, *Buildings*, and *Organisations* domains. We excluded the *Band* domain as we found this dataset to be very noisy; neither classification results nor features obtained from this domain in Chapter 4 are as good as those for the other domains. Information about the datasets and how we divided them into training and testing splits has been given in Section 3.3.

5.3.2 Methods

We have experimented with two clustering algorithms: agglomerative hierarchical average link clustering and HDBSCAN [23]. However, in the case of HDBSCAN we noticed that when using overlap-based dissimilarity, we typically ended up without any clusters[1]. For HDBSCAN we therefore simply used the cosine similarity between the word vectors. We refer to our method with agglomerative clustering as *IncAgg* and to the variant with HDBSCAN as *IncHDB*. Figure 5.1 summarizes the steps involved in finding facets and modeling them as low-dimensional subspaces using these two clustering algorithms. In addition, we considered a variant of the method with agglomerative clustering which relies on cosine similarity instead of the overlap-based dissimilarity (*CosIncAgg*). Finally, we also report results for variants of our methods in which we did not obtain the facets incrementally (*NonIncAgg* and *NonIncHDB*). In these cases, we simply extract r clusters from the initial set of features F and determine the corresponding facets directly. In all cases, we use 50-dimensional pre-trained GloVe word vectors [117] for clustering the features.

[1]Note that HDBSCAN does not cluster all the data points, as it removes data points which are considered as noise.

To generate the initial vector space embedding, we follow the approach proposed in [36] based on multi-dimensional scaling. In all cases, we used 100-dimensional vector spaces and learned 10 facets, each being modelled as a 10-dimensional subspace. To select the set of features F, we initially consider the 500 highest scoring words according to the Kappa metric. However, if we end up without any clusters (in the case of HDBSCAN), we expand the set of features to the 1000 top words. The overlap threshold q is selected based on held-out tuning data, considering values from $\{0.3, 0.5, 0.7\}$. To flatten the agglomerative clustering, we tune the number of clusters from $\{50, 100, 200\}$[2].

5.3.3 Baselines

We compare the model with the full 100-dimensional *MDS* (i.e. the entity embeddings without considering the facets) to evaluate the impact of having a facet-specific representation. We also compare the quality of the feature directions of this model with the two strategies for learning sub-features (*Sub* and *Orth*) that we proposed in Chapter 4.

5.3.4 Evaluation Tasks

Intrinsic evaluation of the learned facets is difficult, among others because what we might consider to be a natural facet is highly subjective. The main hypothesis in this thesis argues that having facet-specific entity embeddings will improve the quality of the learned feature and allow us to predict the properties and the natural categories in a more reliable way (see Section 1.2). Moreover, it is argued that facet-specific entity embeddings are useful in cases where there are few positive examples. Therefore, in our quantitative evaluation, we will focus on the impact of the learned facets in a number of classification tasks. This is also motivated by the view that some types of

[2]The source code is available online at
`https://github.com/rana-alshaikh/Disentangled-Facets`.

classifiers need semantically meaningful features to perform well. Following [36] and [1], for each attribute (e.g. the genre in the movie domain) a binary classifier is trained for each class of this attribute (e.g. the horror class) and then the mean F1 score is reported for all the classes of that attribute. This is will allow us to evaluate to what extent the resulting facet-specific entity embeddings are able to predict these classes.

As in Chapter 4, we will again use depth-1 and depth-3 decision trees to evaluate the quality of the learned features. For the *MDS*, we will use the top-2000 features that we obtained with the method explained in Section 3.5. For a fair comparison, we re-obtained the *Sub* and *Orth* using only the top 2000 features. To evaluate the facets, we instead applied this method to find the top-200 features for each of the facet subspaces.

The performance of the decision trees will allow us to evaluate whether we are able to learn higher-quality feature directions thanks to the decomposition of the vector space into facet subspaces. To evaluate the quality of the facets independently of the quality of the feature directions, we also consider classifiers which use as input the facet-specific vector representations $\mathbf{e}^i$ of the entities. Specifically, we train a support vector machine (SVM) for each of the facets, leading to the predictions $p_1, ...p_k$. These predictions are then aggregated to a final prediction using a logistic regression meta-classifier. As baseline, we simply train a single SVM classifier in the full vector space. Note that this approach is motivated by the theory of conceptual spaces, which suggests that entities have to be compared using Euclidan distance within domain-specific spaces, with overall similarity then determined as a weighted average of the domain-specific similarities. As our final classifier, we used the approach proposed in [18] which is also loosely inspired by conceptual spaces where a Gaussian model is estimated from the positive training (i.e. concepts represented as Gaussians). To estimate the Gaussian, as the number of positive examples is usually smaller than the number of dimensions, they assume that the covariance matrix is a diagonal matrix (i.e. the dimensions are independent). In particular, they estimate a univariate Gaussian for each dimension using Bayesian estimation. For the baseline, we use the dimensions of the

full vector space. For the facet-based representations, we use the dimensions of the facet subspaces. To classify a test example, we then add up the log-probabilities obtained from the facet-specific Gaussians. The example is predicted as positive if the result is above a given threshold. The advantage of this method is that we do not need to train a separate meta-classifier. Intuitively, if a given facet is not relevant for the category which we are trying to predict, we can expect the corresponding Gaussian to have a high variance, which means that it will have a low impact on the final result.

5.3.5 Results

The results are summarized in Table 5.2. Our main method *IncAgg* outperforms the *MDS* baseline for almost all classification tasks and types of classifiers. In several cases the improvements over *MDS* are substantial, particularly in the *Place Types*, *Buildings* and *Organisations* datasets. For the HDBSCAN based variant, the results are more mixed, which seems related to the fact that the overlap based dissimilarity could not be used in that case. Indeed, the cosine based variant of *IncAgg*, i.e. *CosIncAgg*, also performs consistently worse than *IncAgg*. Looking at the performance of *NonIncAgg* and *NonIncHDB* reveals that learning facets in an iterative fashion is critical, given that these two variants perform worse than the baseline in many cases. The decline in the performance when using the non-incremental models instead of incremental models varies from severe (such as in the *Place Types* dataset and the Genre classification problem in the *Movies* dataset) to moderate (such as in the *Buildings* dataset).

Looking more closely at the results of our main method *IncAgg*, it is interesting to note that large improvements are obtained for depth-1 decision trees, which shows that our facet subspaces make it easier to identify features that correspond to the categories from the corresponding classification problems. However, large improvements can also be seen for SVMs and the Gaussians, which shows that the actual decomposition of the space is also helpful.

Comparing the proposed models with the methods from Chapter 4, *Sub* and *Orth*, we can see from the results of depth-1 and depth-3 decision trees that the quality of the feature directions obtained by *IncAgg* outperform those obtained by the implicit approaches in most cases, for all domains except the *Movies* domain. In that domain, the *Sub* and the *IncAgg* results are very competitive, although in general the overall improvements in this domain over *MDS* are not substantial.

		Place Types			Movies			Organisations		Buildings	
		Fours.	Geo.	OpenC.	KeyW.	Genre	Rating	Country	HL.	Country	AL.
DT-D1	MDS	0.34	0.26	0.26	0.26	0.38	0.43	0.67	0.24	0.47	0.47
	Sub	0.38	0.27	0.30	**0.27**	**0.43**	0.46	0.66	**0.27**	0.46	0.49
	Ortho	0.40	**0.30**	0.29	0.26	0.38	0.46	0.67	0.26	0.46	**0.50**
	IncAgg	**0.45**	**0.30**	0.30	0.25	0.40	**0.47**	**0.76**	0.26	**0.50**	0.**50**
	CosIncAgg	**0.45**	0.26	0.30	0.24	0.38	0.43	0.75	0.23	0.43	0.42
	IncHDB	0.43	0.26	0.28	0.25	0.38	0.40	0.50	0.22	0.46	0.46
	NonIncHDB	0.30	0.20	0.27	0.23	0.34	0.40	0.50	0.20	0.46	0.47
	NonIncAgg	0.33	0.24	0.27	0.23	0.33	0.42	0.40	0.21	0.48	0.47
DT-D3	MDS	0.52	0.27	0.32	0.27	0.43	0.47	0.70	0.27	0.47	0.46
	Sub	0.50	0.29	0.31	**0.29**	**0.47**	**0.49**	0.72	0.30	0.51	0.51
	Ortho	0.57	0.32	0.32	0.28	0.45	0.47	0.69	0.30	0.52	**0.53**
	IncAgg	**0.58**	**0.34**	**0.34**	0.27	0.41	0.47	0.77	**0.30**	**0.54**	0.52
	CosIncAgg	0.54	0.28	**0.34**	0.25	0.40	0.45	**0.78**	0.26	0.47	0.45
	IncHDB	0.57	0.26	0.31	0.27	0.41	0.45	0.70	0.27	0.49	0.50
	NonIncHDB	0.43	0.24	0.27	0.26	0.38	0.44	0.60	0.21	0.48	0.49
	NonIncAgg	0.36	0.30	0.29	0.24	0.38	0.45	0.65	0.22	0.51	0.50
SVM	MDS	0.65	0.31	0.35	0.25	**0.54**	0.54	0.71	**0.26**	0.38	0.39
	IncAgg	**0.73**	0.33	**0.37**	**0.26**	**0.54**	**0.55**	0.76	**0.26**	**0.52**	**0.51**
	CosIncAgg	0.62	0.33	0.34	0.25	0.52	0.53	**0.80**	0.12	0.50	0.50
	IncHDB	0.65	0.30	0.36	0.23	0.50	0.51	0.70	0.20	0.51	**0.51**
	NonIncHDB	0.60	**0.35**	**0.37**	0.24	0.46	0.52	0.68	0.24	**0.52**	**0.51**
	NonIncAgg	0.58	**0.35**	0.35	0.24	0.48	0.51	0.72	0.26	0.50	**0.51**
Gaussian	MDS	0.81	0.45	**0.46**	0.26	0.58	0.48	0.74	0.27	0.53	0.51
	IncAgg	**0.87**	**0.48**	0.45	**0.28**	**0.60**	**0.51**	**0.81**	0.27	0.54	**0.55**
	CosIncAgg	0.81	0.45	**0.46**	**0.28**	**0.60**	**0.51**	**0.81**	**0.28**	0.53	0.53
	IncHDB	0.84	0.43	0.43	0.27	**0.60**	**0.51**	0.80	**0.28**	0.54	0.53
	NonIncHDB	0.75	0.41	0.40	0.23	0.51	0.47	0.75	0.27	**0.59**	0.53
	NonIncAgg	0.71	0.46	0.45	0.22	0.52	0.46	0.77	0.27	0.58	0.53

Table 5.2: Classification tasks performance (in terms of F1 score) when using the MDS space and four variation of the facet-based representations.

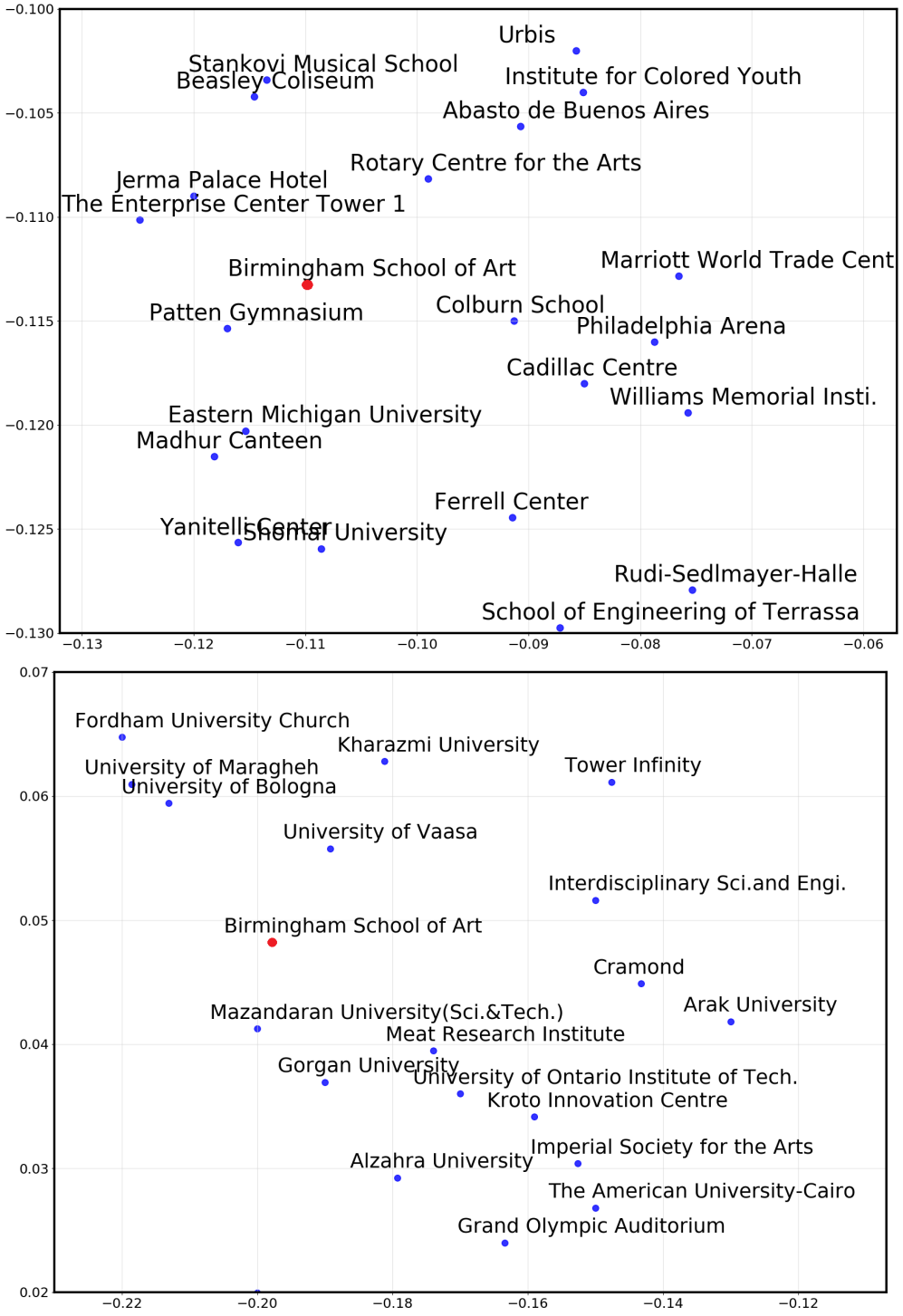


Figure 5.2: Projection of a 100-dimensional semantic space and 10-dimensional facets of buildings. Top: showing the full space. Bottom: showing the 10-dimensional representations for the facet $X_i = \{$campuses, students, offices, centers, facilities, area, hotels, homes, bridges, hospitals, cities, shops, stations$\}$.

ORGANISATIONS	
Cosine similarity	**Overlap-based dissimilarity**
union, europe, protection, aid, spain, right, cross, war, players, black, defenders, german, european	canadian, australian, australia, africa, nations, african, canada, states, countries, asia, united, british, european, competition, world, europe, asian, britain, country, german
PLACE TYPES	
landscapes, serene, tranquil, closeup, surreal, greenery, scenery, scenic, breathtaking, picturesque	sculptures, decoration, churches, fauna, historical, landscapes, st, archeology, small, sculpture, basilica, monuments, convent, heritage, artistic, monument, sacred, forgotten, cemetery, baroque, festival, promenade, renaissance, hall, flora, pavilion, memorial

Table 5.3: Examples of clusters when standard cosine similarity is used (left) and with the proposed overlap based dissimilarity score (right).

5.3.6 Qualitative Analysis

Figure 5.2 illustrates how our subspaces capture similarity in a facet-specific way, showing the two first principal components of the embedding of *Birmingham School of Art* in the full space and in the subspace of a facet that intuitively captures building type. While the neighbors in the full space are a mixture of different building types (hotels, commercial buildings, museum, and educational buildings), in the facet subspace all nearest neighbors are universities.

Table 5.3 illustrates the impact of using overlap-based dissimilarity, where the clusters obtained with cosine similarity are clearly more thematic, while the ones obtained with the overlap-based metric include non-thematic facets (i.e. geographic location and the natural–cultural opposition). Finally, Table 5.4 shows some of the facets obtained for the buildings domain. The first example shows a facet which intuitively captures the historical–contemporary opposition, while the second example shows a facet that captures the rural–city opposition.

BUILDINGS	
Initial cluster X_i:	**Top additional features Y_i:**
architecture, art, history, literature, architectural, society, culture, ancient, scholars, vernacular, classical, historical, contemporary, cultural, medieval	structure, floors, county, architect, graduate, palace, revival, property, hall, united, floor, farm, design, art, space, style, states, downtown, interior, mansion, arena, architectural, architecture, chemistry, entrance.
city, district, town, rural, central, cities, neighborhood, areas, part, section, province, village, north, western, portion, middle, residents, branch	company, wife, headquarters, firm, area, events, estate, people, facility, streets, avenue, schools, hotel, commercial, former, states, architects, original, farm, example, residence,

Table 5.4: Examples of the facets from the Buildings dataset (using *IncAgg* method) .

5.4 Summary

We considered the problem of decomposing a vector space embedding into facets, which are characterized by a set of semantically related features and a corresponding subspace of the embedding. In particular, we focused on unsupervised methods, considering both approaches that rely on the vector space itself (i.e. using subspace clustering) and approaches that additionally take into account the information about word meaning that is captured by pre-trained word vectors. Overall, we found this problem to be highly challenging, in accordance with the findings from Locatello et al. [94] regarding unsupervised disentangled representation learning. However, we were still able to obtain useful facets based on two crucial modifications to a standard clustering based strategy. First, we measure the similarity between features based on two factors: the similarity between their word vectors and the dissimilarity between their meaning in the vector space embedding (measured in terms of overlap). Second, we found it essential to learn facets in an iterative fashion, to avoid too much redundancy

between the different facets. In this chapter, we have heavily depended on clustering algorithms to find facets and have only considered decomposing a given MDS space. In the next chapter, we will discuss the disadvantages of these choices and propose a way to learn facet-specific entity embeddings from scratch.

Chapter 6

A Mixture-of-Experts Model for Learning Multi-Facet Entity Embeddings

6.1 Introduction

In Chapter 5, we proposed an incremental approach to decompose a given entity embedding into facet-specific vector spaces. To provide the required inductive bias, we first determined which properties are captured by the given entity embedding. These properties correspond to words from text descriptions of the entities, whose occurrence can be predicted from the entity vectors. To identify words that are likely to describe properties from the same facet, we relied on the intuition that such properties should have similar word vectors, in a given pre-trained word embedding.

The experimental results from chapter 5 show that learning facet-specific embeddings is indeed helpful for concept induction. However, the incremental approach is applied to entity embeddings that have been learned from text descriptions using multi-dimensional scaling (MDS), which has two important limitations. First, MDS has a quadratic space complexity, which makes it unsuitable for large domains. Second, and most fundamentally, in that method we crucially rely on the assumption that facets of interest correspond to linear sub-spaces of the initial entity embedding. As another

limitation of the incremental approach, the assumption that facets can be identified with clusters in a word embedding space seems too strong. While words that describe properties of the same kind (e.g. different names of movie genres) are indeed often clustered together in a word embedding, the range of words that are relevant to a given facet is usually more varied (e.g. adjectives such as *scary* are relevant when modelling genre, but this word may not be clustered together with genre names). In chapter 4, we proposed an implicit approach to decompose the embedding into different domains which had similar limitations. In particular, the implicit approach still depends on a clustering algorithm, relies on the same assumption that facets can be linearly separable, and it is not scalable to large domains. To address these issues, in this chapter we propose a method that directly learns multi-facet entity embeddings from text descriptions. To this end, we use a mixture-of-experts formulation [62], in which the experts essentially correspond to GloVe models [117], each focusing on a subset of the vocabulary. The decision on which words are modelled by which experts is made by a so-called gating network, which uses pre-trained word vectors as input. In this way, we can capture the intuition that words which are relevant to the same facet typically have similar word embedding representations, without having to assume that all such word appear in a single cluster.

The rest of this chapter is structured as follows. In Section 6.2, we describe the model and explain how we estimate the model parameters. Section 6.3 covers details of the experiments, including the quantitative results, the qualitative results, and a discussion. Finally, Section 6.4 summarises our findings.

6.2 Model Description

The main idea underpinning the mixtures-of-experts (MoE) model [62] is to train a neural network by (i) learning a soft partition of the feature space and (ii) training a separate neural network for each partition class. The individual neural networks, referred to as *experts*, are thus specialized towards the examples from the corresponding partition class. These experts are jointly trained with a so-called *gating network*, which is used to determine the (soft) partition. To apply this model to our setting, we thus need to determine the structure of the gating network and the nature of the experts.

Our aim is to learn facet-specific entity embeddings from the bag-of-words representations (BoW) of a given set of entities. To apply the MoE model to this problem setting, we need an embedding method that can be formulated as a classification or regression problem. Moreover, to allow for an effective gating network, we need the ability to efficiently determine how well different properties are captured by the different entity embeddings. To address both issues, we build on the GloVe word embedding model [117], which is a common choice for learning entity embeddings from BoW representations [64]. Using the notations and terminology of entity embeddings, the GloVe model can be formulated as follows:

$$G = \sum_{j=1}^{m} G_j \quad G_j = \sum_{i:x_{ij}>0} f(x_{ij}) \left(\mathbf{e_i} \cdot \tilde{\mathbf{w}}_\mathbf{j} + b_i + \tilde{b}_j - \log x_{ij} \right)^2 \quad f(x_{ij}) = \left(\frac{x_{ij}}{100} \right)^{0.75} \tag{6.1}$$

Here $\mathbf{e_i}$ represents the embedding of entity e_i, $\tilde{\mathbf{w}}_\mathbf{j}$ is a representation of the word w_j, b_i and $\tilde{b}_j$ are bias terms, x_{ij} is the number of occurrences of w_j in the BoW representation of e_i, and the weight $f(x_{ij})$ is aimed at reducing the impact of rare words. The term G_j captures how well the entity embedding is modelling the word w_j. Similar to Derrac and Schockaert [36], we found that words which are modelled well, i.e. for which the loss term G_j is low, tend to correspond to semantically meaningful properties. The main idea of our method is to learn multiple GloVe embeddings (i.e. experts), where

each embedding will be specialized towards a subset of all words. The key challenge is to train these embeddings such that the properties captured by a given embedding form a semantically meaningful facet or domain. For example, when learning a representation of movies, we would expect to see one GloVe expert that focuses on genre (e.g. capturing words such as *horror*, *zombie*, or *funny*).

What makes this problem particularly challenging is that properties from different facets are often correlated (e.g. particular actors may be strongly associated with a particular movie genre). This is in accordance with the theory of conceptual spaces, but it means that a strong inductive bias is needed to learn these representations. Similar to the incremental approach, we rely on pre-trained word vectors to provide this bias. In particular, we rely on the assumption that whenever a word is related to a given facet, words with similar embeddings tend to be related as well. This assumption is less strong than the assumption in the incremental approach, where each facet was assumed to correspond to a single cluster.

The decision to use MoE to learn facet-specific entity embeddings came as a result of our unsuccessful initial experiments using the Glove model to learn the full space and, at the same time, learn a low-rank projection matrix for each facet. We tried to learn a vector space with k facets. Intuitively, each of those facets should correspond to a low-dimensional subspace. We found that this model tends to model almost the same features across all facets, i.e. the model failed to make clear decisions on which features belonged to which facets. More detail on the initial model is provided in Appendix A.

6.2.1 Model Formulation

If we ignore the weight $f(x_{ij})$, the relationship between least squares regression and the Gaussian distribution makes it easy to see that the GloVe model maximizes the likelihood of the data X (i.e. the matrix of co-occurrence counts x_{ij}) in accordance with the following probabilistic model:

$$\mathcal{L}_{GloVe}(X;\sigma,\mathbf{e},\tilde{\mathbf{w}},b,\tilde{b}) = \prod_{i,j} \mathcal{G}(\log x_{ij} \mid \mathbf{e_i}\cdot\tilde{\mathbf{w}}_\mathbf{j} + b_i + \tilde{b}_j, \sigma^2) \tag{6.2}$$

where $\mathbf{e}$ is the set of the entity representations, $\tilde{\mathbf{w}}$ is the set of the word representations, b and $\tilde{b}$ are the sets of the bias terms for the entities and the words, respectively, and $\mathcal{G}$ is the Gaussian distribution and the variance σ^2 is an arbitrary strictly positive constant. In our MoE model, each expert makes a different prediction for the mean $\mathbf{e_i}\cdot\tilde{\mathbf{w}}_\mathbf{j} + b_i + \tilde{b}_j$. Let us write $\mathbf{e_i}^k$ for the embedding of entity e_i by the k$^{\text{th}}$ expert. Similarly, $\tilde{\mathbf{w}}_\mathbf{j}^\mathbf{k}$ corresponds to the embedding of word w_j, according to this expert, while b_i^k and $\tilde{b}_j^k$ are the associated bias terms. We write K for the total number of experts. Furthermore, let us write $g(k,j)$ for the probability that word w_j should be assigned to the k$^{\text{th}}$ expert. The aim of our model is then to maximize the following likelihood:

$$\mathcal{L}_{MoE}(X;\sigma,\phi,\theta,\xi,\beta_i,\beta_j) = \prod_{i,j}\sum_k g(k,j)\mathcal{G}(\log x_{ij} \mid \mathbf{e_i^k}\cdot\tilde{\mathbf{w}}_\mathbf{j}^\mathbf{k} + b_i^k + \tilde{b}_j^k, \sigma^2) \tag{6.3}$$

where θ is the set of the entity representations produced by each expert for each entity (i.e. $(e_i^1,...,e_i^k)$), ξ is the set of the word representations produced by each expert for each word (i.e. $(w_j^1,...,w_j^k)$), while β_i, and β_j are the sets of the bias terms, learned by each expert, for the entities and the words, respectively. The probability $g(k,j)$ will be parameterized by a neural network, called the gating network, where ϕ is the set of the parameters of this neural network. In particular, let $(y_1^j,...,y_K^j) = \phi(\mathbf{x_j})$ be the output of a multi-layer perceptron, where the input $\mathbf{x_j}$ is the pre-trained word vector for w_j. The probabilities $g(k,j)$ are then obtained using softmax:

$$g(k,j) = \frac{\exp(y_k^j)}{\sum_{i=1}^K \exp(y_i^j)} \tag{6.4}$$

Note that the decision on which expert should be used for the prediction of x_{ij} only depends on the word w_j in our model. The aim of the gating network is thus to find a

meaningful grouping of the words from the BoW representations. Another possibility would be to design the gating network such that the entity e_i is taken into account as well. In principle, this would be useful to determine for each of the learned facets, which entities can have a meaningful representation in that facet. However, in preliminary experiments we were not able to achieve better results which such an approach.

6.2.2 Parameter Estimation

Our aim is now to train the parameters of the gating network and those of the different GloVe experts. We rely on Expectation Maximization (EM) for this purpose.

E-step: For each context word w_j, we estimate a probability distribution over experts, which is based on how well these experts are currently modelling this word. In particular, let us write $\epsilon_{(k,j)}$ for the error term associated with w_j and the k$^{\text{th}}$ expert, i.e.:

$$\epsilon_{(k,j)} = \sum_{i:x_{ij}>0} \left(\mathbf{e_i^k} \cdot \tilde{\mathbf{w}}_\mathbf{j}^\mathbf{k} + b_i^k + \tilde{b}_j^k - \log x_{ij} \right)^2 \tag{6.5}$$

Note that in contrast to the standard GloVe formulation, we do not use the weight $f(x_{ij})$, as we found this weighting strategy not to be helpful in our setting. In particular, we noticed that when using $f(x_{ij})$ with $g(k,j)$ the model converge was slower than when it was removed, and omitting it simplifies the formulation of the model without affecting the performance of the model. The probability $S_{(k,j)}$ that w_j should be assigned to the k$^{\text{th}}$ expert is then estimated as follows:

$$S_{(k,j)} = \frac{\exp(-\epsilon_{(k,j)})}{\sum_{i \in K} \exp(-\epsilon_{(i,j)})} \tag{6.6}$$

These probabilities $S_{(k,j)}$ will be used as the supervision signal for training the gating network.

M-step: We train the gating network by minimizing the cross-entropy between the probabilities $S_{(k,j)}$ obtained from the E-step and the probabilities $g(k,j)$ predicted by the gating network:

$$E_{gate} = -\sum_{j=1}^{m}\sum_{k=1}^{K} S_{(k,j)} \ln g(k,j) \tag{6.7}$$

with $g(k,j)$ defined as in (6.4). For each expert, the corresponding parameters are learned by using the following weighted version of the standard GloVe loss (without the weights $f(x_{ij})$:

$$G_{(k)} = \sum_{j=1}^{m}\sum_{i:x_{ij}>0} g(k,j)\left(\mathbf{e_i^k}\cdot\tilde{\mathbf{w}}_\mathbf{j}^\mathbf{k} + b_i^k + \tilde{b}_j^k - \log x_{ij}\right)^2 \tag{6.8}$$

In the first iteration of the EM method, the parameters are initialized by training a standard GloVe embedding. In particular, instead of randomly initializing the parameters in (6.6) and (6.8), a standard GloVe (i.e, without $g(k,j)$) was trained for 10 iterations for each facet, the learned parameters were then used to initialize (6.6) and (6.8) in the first iteration of the EM method. In subsequent iterations, we use the parameters from the previous iteration for initialization.

6.3 Experiments

We experimentally analyze the performance of the proposed mixture-of-experts (MoE) model. Our main focus is on showing that learning facet-specific embeddings is useful compared to learning standard embeddings. We also compare this method with the incremental approach from Chapter 5.

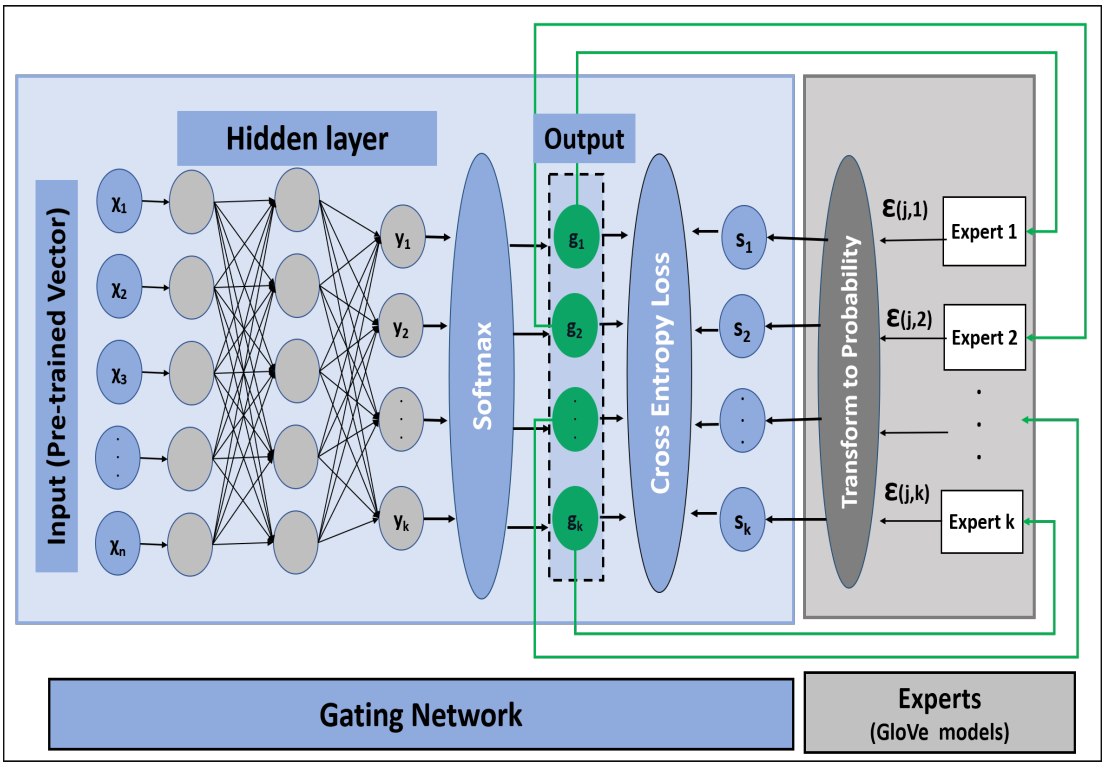


Figure 6.1: MoEGloVe Architecture

6.3.1 Datasets

To allow for a comparison with the method from Chapter 5, we first consider three of the smaller datasets: the *Movies*, *Place Types*, and the *Buildings*. Then, we also evaluate our method on the two larger datasets *Locations* and *Wikipedia*. In this chapter, we did not include the *Organisations* dataset, since its BoW representations heavily relies on words for which we did not have a pre-trained word vector. Since our model cannot take into account such out-of-vocabulary words when learning the embeddings, no suitable representations could be learned for this domain. Note that this can be easily addressed by training the word vectors on a domain-specific corpus.

6.3.2 Methodology

To learn the embeddings with our proposed model, we train k experts, choosing k from $\{4, 5, 10\}$ based on the tuning data. In all cases, we fix the total number of dimensions of all embeddings to 100 (e.g. if $k = 4$ then each expert learns a 25-dimensional embedding), this only to simplify the evaluation. In this method, the number of the experts and dimensions are hyperparameters that only depend on the dataset, and are independent of each other. For example, for some domains, only two facets are needed with 10-dimensions each. As input to the gating network, we use a 50-dimensional GloVe word embedding. We run the EM algorithm for five iterations, which we found sufficient for the experts to converge. In each iteration, we train the gating network for 20 epochs and the experts for 10 epochs, using AdagradOptimizer with a learning rate of 0.05 and mini-batch size of 1000.

6.3.3 Baselines

Our main baseline is the standard GloVe model, as in (6.1), which was found by Jameel and Schockaert [65] to produce highly competitive entity embeddings, compared to a wide range of other methods. We also experimented with methods based on variational autoencoders, including the Neural Variational Document Model [103], but we were not able to obtain competitive results in this way. For a fair comparison, we fix the number of dimensions in all entity embeddings to 100, as this is the total number of dimensions of all the facet embeddings (see Section 6.3.2). We also compare our MoE method against the incremental approaches from Chapter 5, which are referred to as *IncAggGloVe* and *IncHDBGloVe*. These methods differ in the clustering algorithms that are used for identifying facets, which are Agglomerative Hierarchical Clustering and HDBSCAN, respectively. Note that in addition to the incremental approaches, where MDS was used, we apply these methods to a 100-dimensional GloVe embedding, to allow for a more direct comparison. However, we found that these methods were not

able to scale to the new *Locations* and *Wikipedia* datasets, even when using GloVe for the base embedding, hence we can only consider them for *Movies*, *Place types* and *Buildings*.

6.3.4 Evaluation tasks

We evaluate the quality of the learned embeddings based on the performance of a number of different classifiers which use these embeddings as input. In particular, we followed the same approach as the previous chapter, which uses four types of classification methods. The first method is again to train a (linear) SVM classifier on each of the different facet-specific spaces and then combine these predictions using a logistic regression meta-classifier. For the GloVe baseline, we simply train an SVM classifier on the full space. The second method is based on the same view, but instead of using SVMs we use k-nearest neighbors (kNN). The value of k was chosen from $\{1, 3, 5\}$ based on the tuning data. The third method, is to estimate a Gaussian distribution, in each of the facet-specific spaces, in the same way as in Chapter 5. The fourth classification method is based on low-depth decision trees, we again use dept-1 and dept-3 decision trees to evaluate the quality of the feature directions themselves. The aim is to evaluate to what extent important semantic features can be modelled as vectors. In particular, we first select the N words which are best modelled in the vector space (for each expert), i.e. we choose the words j for which the error term $\epsilon_{(k,j)}$ is minimal. For the GloVe baseline, we set $N = 2000$. For the other methods, we select $N = 200$ words from each of the facet-specific spaces.

6.3.5 Results

The results are summarized in Table 6.1 for the three smaller datasets and in Table 6.2 for the two larger datasets. As can be seen from the tables, our model outperforms each of the baselines. Moreover, the improvement over GloVe is substantial in many cases, which clearly shows the usefulness of learning multiple facet-specific vector spaces, rather than a single higher-dimensional space. Our model also outperforms IncAgg and IncHDB, in addition to being much more scalable. In fact, surprisingly, the IncAgg and IncHDB methods perform worse than the GloVe baseline in several cases. In contrast, as was reported in Chapter 5, when MDS is used as the base embedding, these methods consistently improve on this base embedding, although they still do not reach the performance of the MoEGloVe model. Table 6.1 shows a detailed comparison between the MoEGloVe methods and MDS based representation used in the incremental approach. For a fair comparison, we relearned the MDS for the movies dataset using only the words that have pre-trained word embedding as they are far less than the number of the total vocabulary.

While we are not primarily concerned with the overall performance of the classifiers, it is interesting to note that the performance of the SVN, kNN and Gaussian classifiers are broadly comparable. The decision trees perform worse overall, as could be expected. However, the relative performance of the decision trees, compared to the other classifiers, can reveal which categories can be modelled in terms of the most dominant linear features, i.e. the vectors $\mathbf{w_i^k}$ with the lowest associated error term $\varepsilon_{(k,j)}$. Such features can intuitively play the role of quality dimensions in applications [36]. The results suggest that the land cover categories in the *Locations* domain correspond to such dominant linear features. In contrast, for the Foursquare categories, the performance of the decision trees is much worse than that of the other classifiers, showing that methods that rely on learned quality dimensions would not model these categories well.

		Place Types			Movies			Buildings	
		Fours.	Geo.	OpenC.	KeyW.	Genre	AR	Country	AL.
DT1	MDS	0.34	0.26	0.26	**0.26**	0.38	0.43	0.47	0.47
	GloVe	0.34	0.23	0.26	0.22	0.32	0.39	0.46	0.30
	IncAggMDS	**0.45***	**0.30***	**0.30***	0.25	**0.40**	**0.47***	0.50*	**0.50***
	IncHDBMDS	0.43	0.26	0.28	0.25	0.38	0.40	0.46	0.46
	IncHDBGloVe	0.35	0.26	0.29	0.24	0.30	0.39	0.48	0.44
	IncAggGloVe	0.35	0.26	0.29	0.23	0.37	0.40	**0.52**	0.49
	MoEGloVe	**0.45***	0.27*	**0.30***	**0.26***	**0.40***	0.44*	0.48*	0.45*
DT3	MDS	0.52	0.27	0.32	**0.27**	**0.43**	**0.47**	0.47	0.46
	GloVe	0.47	0.29	0.30	0.24	0.37	0.40	0.50	0.41
	IncAggMDS	0.58	**0.34***	**0.34***	**0.27**	0.41	**0.47**	**0.54***	**0.52***
	IncHDBMDS	0.57	0.26	0.31	0.26	0.39	0.44	0.49	0.50
	IncHDBGloVe	0.50	0.24	0.31	0.24	0.35	0.40	0.52	0.47
	IncAggGloVe	0.44	0.26	0.29	0.23	0.37	0.39	0.53	0.49
	MoEGloVe	**0.59***	0.31*	**0.34***	0.26*	0.40*	0.45*	0.52*	0.48*
SVM	MDS	0.65	0.31	0.35	0.25*	0.43	0.45	0.38	0.39
	GloVe	0.75	0.39	0.37	0.17	**0.47**	0.33	0.64	0.45
	IncAggMDS	0.73*	0.33*	0.37*	0.23	**0.47**	0.45	0.52*	0.51*
	IncHDBMDS	0.65	0.30	0.36	0.23	**0.47**	**0.47**	0.51	0.51
	IncHDBGloVe	0.58	0.27	0.33	0.20	0.44	0.37	0.55	0.51
	IncAggGloVe	0.62	0.26	0.31	0.22	0.46	0.40	0.60	0.50
	MoEGloVe	**0.76***	**0.45***	**0.40***	**0.26***	**0.47**	**0.47***	**0.68***	**0.55***
KNN	MDS	0.65	0.31	0.35	0.20	0.50	0.42	0.47	0.49
	GloVe	0.70	0.40	0.38	0.13	0.23	0.30	0.56	0.54
	IncAggMDS	**0.73***	0.40*	0.40*	0.25*	**0.52***	0.47*	0.51*	0.50*
	IncHDBMDS	0.65	0.33	0.37	0.25	**0.52**	0.25	0.47	0.49
	IncHDBGloVe	0.58	0.38	0.38	0.16	0.33	0.30	0.57	0.52
	IncAggGloVe	0.68	0.32	0.40	0.17	0.32	0.35	0.57	0.50
	MoEGloVe	**0.73***	**0.42***	**0.42***	**0.27***	0.47*	**0.48***	**0.64***	**0.57***
Gaussian	MDS	0.81	0.45	0.46	0.26*	0.58*	0.48*	0.53	0.51
	GloVe	0.71	0.48	0.43	0.19	0.50	0.42	0.62	**0.56**
	IncAggMDS	**0.87**	0.48	0.45	0.23	0.55	0.45	0.59	0.55
	IncHDBMDS	0.84	0.43	0.43	0.27	**0.60**	**0.51**	0.54	0.53
	IncHDBGloVe	0.62	0.38	**0.50**	0.22	0.44	0.42	0.62	**0.56**
	IncAggGloVe	0.64	0.43	0.48	0.22	0.48	0.41	0.62	**0.56**
	MoEGloVe	0.74	**0.50***	0.49*	**0.24***	0.54*	0.47*	**0.66**	**0.56**

Table 6.1: Classification tasks performance (in terms of F1 score) when using the MDS space and GloVe Space. * indicates statistically significant scores (MDS Vs IncAggMDS) and (GloVe Vs. MoEGloVe) based on the McNemar's test ($p < 0.05$).

		Wikipedia										Locations		
		SM.	MoL.	Mocl.	MoC.	MuC	MuG	BLF	BC	HG	HC	CL1	CL2	CL3
DT1	GloVe	0.19	0.37	0.33	**0.18**	0.15	**0.09**	**0.16**	0.30	0.32	0.35	0.24	0.10	0.05
	MoEGloVe	**0.23**	**0.42**	**0.42**	**0.18**	**0.18**	0.08	0.15	**0.34**	**0.43**	**0.49**	**0.31**	**0.13**	**0.06**
DT3	GloVe	0.23	0.38	0.31	0.19	0.16	**0.10**	**0.17**	0.31	0.33	0.36	0.30	0.11	0.05
	MoEGloVe	**0.28**	**0.46**	**0.43**	**0.21**	**0.19**	**0.10**	**0.17**	**0.33**	**0.36**	**0.43**	**0.31**	**0.13**	**0.07**
SVM	GloVe	0.49	0.53	0.48	0.20	0.17	0.08	0.11	0.32	0.44	**0.66**	0.15	0.07	0.01
	MoEGloVe	**0.61**	**0.62**	**0.51**	**0.25**	**0.27**	**0.12**	**0.18**	**0.47**	**0.47**	0.64	**0.30**	**0.10**	**0.06**
kNN	GloVe	0.49	0.33	**0.56**	0.17	0.14	0.07	0.12	0.29	0.50	0.52	0.29	0.21	0.17
	MoEGloVe	**0.60**	**0.50**	0.55	**0.25**	**0.25**	**0.11**	**0.16**	**0.39**	**0.52**	**0.58**	**0.39**	**0.24**	**0.21**
Gauss	GloVe	0.51	0.52	0.55	0.25	0.24	0.11	0.16	**0.19**	0.49	0.13	0.30	0.11	0.10
	MoEGloVe	**0.57**	**0.59**	**0.57**	**0.27**	**0.26**	**0.12**	**0.20**	0.18	**0.51**	**0.16**	**0.33**	**0.17**	**0.14**

Table 6.2: Classification performance in terms of F1 score For *Wikipedia* and *Locations*.

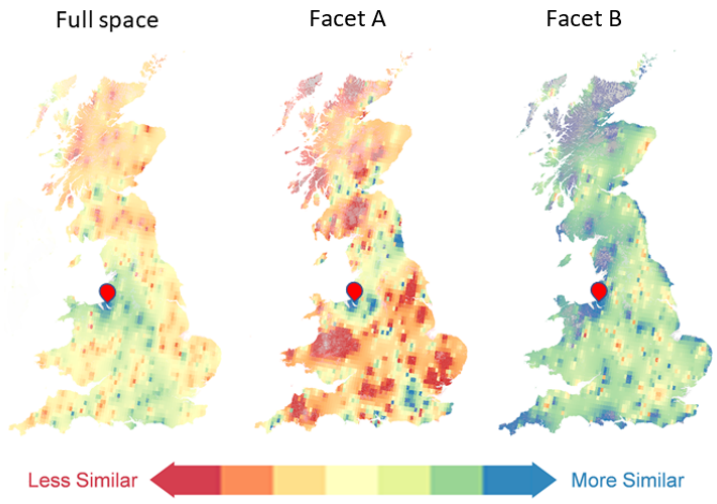


Figure 6.2: Projection of the full space and two 10-dimensional facets of the *Locations* dataset [2].

6.3.6 Qualitative Analysis

To illustrate the usefulness of facet-specific vector spaces, Table 6.3 shows the nearest neighbors of some movies (i) in the full space and (ii) in one of the facet-specific spaces, which is intuitively specialized towards genre. While several of the nearest neighbors in the full space have a similar genre, we can also see many other neighbors (shown in red). In contrast, the neighbors in the genre-specific space all have a similar genre. In Table 6.4 we show, for a number of experts, which words are assigned to them by the gating network, i.e. for which words the probability $g(k, j)$ is highest. For the *Movies* domain, for instance, we can see that one expert focused on technical aspects of the movies (e.g. *soundtrack*, *graphics*, *cinematography*), while the second expert focused on genre, and the third expert focused on the particular genre of historical movies. For the *Wikipedia* and *Place Types* datasets, which cover a wider range of entities, the discovered facets are mostly thematic. For instance, for the *Wikipedia*

[2]The map was generated by Shelan Jeawak, one of the co-authors of the paper that interoduced this approach.

dataset, we found facets related to companies, music and politics. Finally, Figure 6.2 visually shows the different aspects of similarity that are captured by two experts for the *Locations* dataset. Specifically, the figure visualizes how similar different parts of the UK are to the target location, in Liverpool. For one expert (Facet A), the most similar regions correspond to other urban areas (including London, Southampton, and Newcastle). On the other hand, the second expert (Facet B) has identified coastal areas across the UK as the most similar regions. While this latter facet may be important in some application contexts, it is clearly not well-captured in the full space (i.e. the standard GloVe embedding).

MOVIES		
Movie	**5NN in full space**	**5NN in the Genre facet**
Troy 2004 [Drama, Adventure]	**Alexander 2004** [Action, Drama, Adventure, Biography, War, Romance, History], **Hum 1991** [Action, Drama, Crime, Family], Pig 2010 [Horror], **Kingdom of Heaven 2005** [Action, Drama, Adventure, War, History], **Mac 1992** [Drama]	**Alexander 2004** [Action, Drama, Adventure, Biography, War, Romance, History], **King Arthur 2004** [Action, Drama, Adventure, War, History], **Kingdom of Heaven 2005** [Action, Drama, Adventure, War, History], **Lawrence of Arabia 1962** [Drama, Adventure, Biography, War, History], **Master and Commander, The Far Side of the World 2003** [Action, Drama, Adventure, War]
Iron Man 2008 [Action, Adventure, Sci-Fi]	**Fantastic Four 2005** [Action, Adventure, Fantasy, Sci-Fi, Short], **Hulk 2003** [Action, Sci-Fi], U 2006 [Animation, Family, Music, Musical], **Seven 1979** [Action, Drama],Ali 2001 [Drama, Biography, Sport]	**Hulk 2003** [Action, Sci-Fi], **Aliens 1986** [Action, Adventure, Thriller, Sci-Fi], **Terminator 3-Rise of the Machines 2003** [Action, Thriller, Sci-Fi], **X2 2003** [Action, Adventure, Thriller, Sci-Fi], **Star Trek Nemesis 2002** [Action, Adventure, Thriller, Sci-Fi]
X-Men 2000 [Action, Adventure, Sci-Fi]	**X2 2003** [Action, Adventure, Thriller, Sci-Fi], **Fantastic Four 2005** [Action, Adventure, Fantasy, Sci-Fi, Short], **Spider-Man 2 2004** [Action, Adventure, Fantasy], Nu 2003 [Drama, Short], **ATL 1999** [Comedy, Drama, Crime, Music, Romance]	**Superman II 1980** [Action, Sci-Fi], **Thunder 1983** [Action, Drama, Crime], **X2 2003** [Action, Adventure, Thriller, Sci-Fi], **Iron Man 2008** [Action, Adventure, Sci-Fi], **Terminator 3, Rise of the Machines 2003** [Action, Thriller, Sci-Fi]
The Sound of Music 1965 [Drama, Biography, Family, Music, Musical, Romance]	**Mamma Mia! 2008** [Comedy, Music, Musical, Romance], Show 2003 [Action, Comedy, Crime, Thriller], **Across the Universe 2007** [Drama, Music, Musical, Romance], Pig 2010 [Horror], **Its a Wonderful Life 1946** [Drama, Family, Fantasy]	**Willy Wonka and the Chocolate Factory 1971** [Family, Fantasy, Music, Musical], **Casablanca 1942** [Drama, War, Romance], **Its a Wonderful Life 1946** [Drama, Family, Fantasy], **Singin in the Rain 1952** [Comedy, Music, Musical, Romance], **A Christmas Story 1983** [Comedy, Family]
Mystic River 2003 [Drama, Crime, Thriller, Mystery]	Now 1965 [Music, Short, Documentary], **Training Day 2001** [Action, Drama, Crime, Thriller], **Mac 1992** [Drama], **21 2008** [Drama, Crime, Thriller], **Blue Velvet 1986** [Drama, Crime, Thriller, Mystery]	**21 2008** [Drama, Crime, Thriller], **Atonement 2007** [Drama, War, Mystery, Romance], **Jarhead 2005** [Drama, Biography, War], **The Door 2012** [Horror, Drama, Family, Fantasy, Thriller, Mystery, Sci-Fi, Short], **Red Dragon 2002** [Crime, Thriller]

Table 6.3: Examples of the Nearest Neighbors from the *Movies* dataset.

MOVIES		
Expert 0: animation, soundtrack, studio, remastered, recording, feature, graphics, audio, productions, animated, series, musical, artwork, compilation, version, featuring, interactive, playstation, artists, premiere, theatrical, cinematography, animations, entertainment, unreleased	**Expert 1:** drama, comedy, hollywood, actor, actress, age, teen, children, dramas, mother, remake, horror, death, sitcom, opera, tale, princess, bollywood, shakespeare, wife, broadway, television, fiction, thriller, stories, comedies, romance, family, sex, live, series, documentary, animated, romantic , adventure, mystery, father, fantasy, crime, sequel, reality	**Expert 2:** manuscript, century, medieval, poet, testament, writings, biography, author, memoirs, nineteenth, treatise, philosopher, literature, bestselling, texts, historical, poem, preface, history, narratives, romanticism, allegorical, book, centuries, autobiography, historians, thinkers
WIKIPEDIA		
Expert 0: companies, applications, technologies, firms, equipment, tools, manufacturers,software, motor, computers, management, options, auto, automaker, toyota, product, sale, microsoft, user, buy.	**Expert 1:** music, album, song, radio, film, pop, television, hip-hop, soundtrack, airplay, movies, recordin, tv, billboard, compilation, bbc, chart, musical, band, aired, broadcast, uhf, channel,comic, operas, bands, studio, tunes, indie, broadcasts	**Expert 2:** criminal, crimes, accused, activists, communist, ethnic, opposition, democratic, palestinian, communists, serbs, regime, allies, yugoslavia,israeli, leftist, anti, political
PLACE TYPES		
Expert 0: italy, thailand, temple, sri, inn, yorkshire, terrace, buddhist, thai, indian, beach, wine, india, cook, malaysia, condo, durham, turkey, village, restaurant, tofu, dining, cornwall	**Expert 1:** ferrari, cessna, jaguar, falcon, musica, mexicana, guitarra, banda, porsche, flight, grupo, airshow, supercars, guitarist, coupe, supercar, peugeot, beetle, jazz,benz, mustang, flamenco, mercedes, amphibian.	**Expert 2:** hair, costumes, cute, kitten, kitty, stockings, dolls, doll, toys, costume, makeup, fashion, nail, puppy, lipstick, teddy, lingerie, smile, sexy, zombie, retro, nails, blouse

Table 6.4: Examples of the facets from the *Movies*, *Wikipedia*, and *Place types* datasets .

6.4 Summary

This chapter has introduced MoEGloVe, an unsupervised method for jointly learning a number of facet-specific low-dimensional entity embeddings from BoW representations. The model uses an MoE formulation, which relies on multiple GloVe models as experts. Each expert is encouraged, by the gating network, to specialize in one aspect of the data. The proposed method is scalable and can be applied to large domains due to the use of MoE and GloVe. We evaluated our method on five datasets, two of which are much larger than the datasets that were considered in Chapters 4 and 5 . We presented the experimental results, which show that learning facet-specific spaces can be highly beneficial. Our results show that the MoEGloVe method outperforms GloVe and the incremental approaches from Chapter 5, especially when GloVe is the base representation. Notably, the incremental approaches perform consistently better than the base representation when using MDS, but not when using GloVe. In general, from the experiments conducted, we can clearly see that both the incremental approaches using MDS, and the MoEGloVe method, improve the classification results substantially and consistently in comparison with the standard MDS and GloVe representations, respectively. However, in terms of scalability, MoEGloVe is more scalable than the incremental approaches even when the base representation is GloVe; this comes as a result of using the MoE model. Finally, our qualitative analysis presents evidence that the facet-specific spaces resulting from MoEGloVe are able to capture aspects of similarity much more clearly than the full space.

Chapter 7

Conclusions and Future Work

7.1 Introduction

Motivated by the facts that similarity is a multi-faceted notion and that learning single entity embeddings does not reflect this multi-faceted nature, we have addressed in this thesis the problem of learning facet-specific entity embeddings. We have argued that the resulting representations model similarity in a more natural way than standard entity embeddings, and that this is important to improve entity classification. As a starting point, we adopted a broader view of conceptual spaces to learn facet-specific entity embeddings without supervision. Little research has yet been done on learning such representations and particularly on learning data-driven conceptual spaces in an unsupervised way. Our proposed models were evaluated on a number of domains using different types of classification algorithms. This chapter concludes the thesis as follows. Section 7.2 summarizes the work of the thesis and relates our contributions to the research hypothesis as well as summarizing our findings. In Section 7.3 we revisit the four research questions considered in the thesis. Finally, Section 7.4 highlights some possible paths for future work.

7.2 Thesis Summary and Contributions

In our modern world, a large number of entities are added to the volume of online knowledge every day, including books, events, movies, and many more. Each one of those entities is associated with a descriptive information, in forms including pictures, text, audio, and video. Since this type of knowledge is very valuable to large numbers of applications, these entities need to be represented in a way that allows computer systems to understand, retrieve, and classify them, maximizing the benefits of the associated information. Learning entity representations is therefore a very active area within AI. Entity embedding is one way to represent entities by mapping them into a dense continuous vector space, using different types of inputs. Various methods have already been proposed for learning entity embeddings from text descriptions. Such embeddings are commonly used for inferring properties of entities, for recommendation and entity-oriented search, and for injecting background knowledge into neural architectures, among others. Entity embeddings essentially serve as a compact encoding of a similarity relation, but similarity is an inherently multi-faceted notion. By representing entities as single vectors, existing methods leave it to downstream applications to identify these different facets, and to select the most relevant ones. This limitation of existing entity embeddings models motivated us to seek a way of learning entity embedding that models the different facets of similarity.

In his book, Gärdenfors [48] proposes the conceptual spaces, which are geometric meaning representations in which similar entities are represented by similar vectors. This differs from standard representation learning methods for inducing vector space embeddings from text corpora in two crucial ways. First, the dimensions of a conceptual space correspond to salient semantic features, known as quality dimensions, whereas the dimensions of learned embeddings typically lack any clear interpretation. This has been partially addressed in previous work, which has shown that it is possible to identify directions in learned vector spaces which capture semantic features. Second, conceptual spaces are normally organized into a set of facets, each of which

reflects a different aspect of the entities and is associated with a separate vector space. Based on these differences, we argue that learning entity embeddings in a form of conceptual spaces will model the different facets of similarity. Little research has yet been done on learning such representations. In this thesis we considered the problem of learning data-driven conceptual space representations in an unsupervised way. We proposed three unsupervised models that use text descriptions associated with a set of entities to learn representations for those entities. We argued that these representations model similarity better than standard entity embeddings and subsequently improves entity categorization and property prediction.

Our research hypothesis was presented in Chapter 1 as follows: "The initial vector space embeddings generated by traditional ways of learning entity representation are not interpretable, lack clear structure, and do not reflect the different facets of similarity in their structure. Previous research, especially that inspired by the conceptual spaces framework, has shown that it is possible to learn a semantic representation of a vector space by learning interpretable dimensions, i.e. to identify semantic features as directions in the vector space. We hypothesize: 1) that by adapting the conceptual spaces framework and structuring the entity embedding space, either explicitly or implicitly, into separate facets, the quality of these learned semantic features will improve; also, 2) that by having multiple low-dimensional vector spaces for entities such that each one specializes in one of the facets (i.e. facet-specific representation), the properties and categories of entities can be predicted more reliably than from standard representations."

In Chapter 2, we reviewed the literature on conceptual spaces and entity embeddings. We also reviewed the literature on some other intersecting or related areas such as disentangled representation learning and the subspace clustering problem. Additionally, we explained in detail the machine learning algorithms that were used or considered within the research. In Chapter 3, we introduced datasets for evaluating our proposed methods and explained how we obtained them. The classification tasks associated with

each dataset, and the methods used to obtain the vector space representations were also explained.

In Chapter 4, we proposed an unsupervised approach that takes a vector space representation and splits it up, implicitly, into facets. The starting point for this model is the method proposed by Derrac and Schockaert [36], which learns the semantic features of a certain domain as interpretable directions, i.e. quality-dimension like representation, in the vector space. To illustrate this method, first they extract the terms that appears sufficiently frequently in the text descriptions of the entities of a specific domain and consider these the salient features of that domain. Then, for each candidate word a linear classifier is learned that splits the space by separating entities that have this semantic feature from the rest. The performance of the classifier is the indicators of how well the direction models the semantic feature. One problem with these directions is that many of the semantic features that correspond to them are only meaningful for a subset of entities. Our model extends the method of Derrac and Schockaert to learn two levels of features instead of one, i.e. we learn primary features and then for each primary feature we learn sub-features, in a hierarchical fashion. Our hypothesis is that by doing this we can improve the quality of the directions and identify facet-specific features, meaning that we can implicitly decompose the vector space into facets. Our approach can be summarized in the following steps. We first obtain the primary features of the considered domain by clustering the directions. Then, we associate each primary feature, i.e. each cluster, with a direction that is obtained by training a linear classifier that separates the entities that have the primary feature (positively classified) from the other entities (negatively classified). To find the sub-features for each primary feature we re-apply the same steps but we train the linear classifier to obtain the directions using only the positively classified entities. To evaluate the quality of the feature directions we trained depth-1 and depth-3 decision trees on the feature-based representations generated by our model from five datasets. The performance of the decision tree on our feature-based representations was compared with the performance of a model that uses only primary features, as well as average-link agglomerative hierarch-

ical clustering, and a model in which the feature directions are chosen randomly. The results shows that our method outperforms the other methods. Our expectation was that the primary features would correspond to facets; however, we found that some of the obtained primary features corresponded instead to quality dimensions. The primary features that represent facets are either thematic facets or subsets of non-thematic facets, i.e. sub-facets of non-thematic facets. This indicates that a small number of well-defined facets can be obtained by directly clustering the directions.

Therefore, in Chapter 5, we proposed a model that transfers a standard entity embedding to a conceptual-space like representation by explicitly decomposing an n-dimensional vector space into r-dimensional facets. As before, our first step involved using the method from [36] to learn the interpretable directions in the vector space; however, instead of using these feature directions directly to obtain facets we used pre-trained word embedding vectors that corresponded to the features. We introduced an overlap score and used this score to build a dissimilarity matrix, with the aim of grouping paradigmatically similar words (based on word embedding) that have dissimilar directions in the same cluster. To avoid finding redundant facets we applied our method in an incremental way, each time choosing the facets that had general and diverse features. Then, to model the candidate facet as a subspace we learned a new feature direction for each term in that facet space by training a logistic regression classifier. We added a constraint to the training objective of the logistic regression classifier to force the coefficient vectors to lie in an r-dimensional space. To find the next facet we repeated the same steps but instead of using the n-dimensional space we removed the dimensions of the previous facets, e.g. to learn the second facet we used an $n - r$ dimensional space. The incremental nature of the approach was crucial to obtaining different facets rather than having multiple versions of the most dominant facets. The model was evaluated on a variety of datasets using different classifiers. We compared the performance of our model with that of the original space and the previous model. To test the impact of the overlap-based dissimilarity matrix and the incremental ap-

proach we compared our model with variations in which these techniques are not used. The results showed that the resulting facet-specific embedding outperforms the the full space for almost all classification tasks and types of classifiers. Moreover, our model was able to find diverse and non-thematic facets.

In Chapter 6, a more scalable model was proposed to learn facet-specific entity embedding directly from the BoW representation. The MoE formulation was used to train a number of experts, where each expert corresponded to a GloVe model that learned a low-dimensional facet, i.e each expert specialized in one facet. Using pre-trained word embedding vectors, the gating network assigned the vocabulary to each expert. In this way, we learned conceptual-space like representations by learning the quality dimensions and the facets simultaneously. To train the parameters we used the EM algorithm. This algorithm first estimates a probability distribution for each word over experts; then, the gating network minimizes the difference between this probability and the one predicted by the gating network. The model was evaluated on five different datasets, two of them relatively large, to test the scalability of the model. As in the previous chapter, we compared the performance of several classifiers when using our facet-specific representations and when using the full space. We also compared the facet-specific representations with the incremental approach from Chapter 5. The results showed that the improvement of the facet-specific vector spaces over GloVe is substantial in many tasks, which again emphasizes the benefits and the positive impact of facet-specific representations. This model also outperforms the incremental approach, as the incremental approach performs less well than the full space when using GloVe. Regarding the obtained facets, the model was able to find diverse set of facets for each domain.

To conclude, the experimental results obtained from our proposed models show an improvement in entity classifications when using facet-specific embeddings, which supports our hypothesis. Our models were able to capture different facets of similarity and learn higher quality directions. We expect that such representations will have a positive impact on many of the tasks that depend on similarity between entities as they captures similarity more faithfully than standard vector space embeddings such as concepts induction, and recommendation systems.

7.3 Research Questions and Main Findings

In this section the research questions will be revisited and discussed in terms of the relation between each question and the research that was conducted in this thesis.

Research Question 1: *For a given entity embedding, can we discover a variety of meaningful facets by clustering the learned feature directions, or is an external supervision signal required for this purpose?*

To answer this question, in Chapter 4 we tried clustering the learned feature directions directly; in Chapters 5 and 6 we tried using a pre-trained word embeddings vector as an external supervision signal. Our main finding is that clustering the feature directions well leads mostly to thematic facets; in order to obtain different types of facets, an external supervision signal is needed.

Research Question 2: *Since the conceptual space structure can be viewed as a hierarchical one, can we structure, i.e. implicitly decompose, the given vector space into facets by identifying the features directions in a hierarchical fashion?*

To handle the problem of decomposing the vector space into facets, in Chapter 4, we started by implicitly decomposing the vector space into facet-like regions. We relied on identifying the feature directions in a hierarchical fashion to obtain the facets and learn the facet-specific features using the facet's region. We found that separating the

space by considering only the entities classified as positive by the facet's associated linear classifier is not suitable for facets that are relevant to most of the entities.

Research Question 3: *How and to what extent, can we, in an unsupervised way, explicitly decompose a given vector space embedding into low-dimensional vector spaces that capture meaningful facets?*

To answer this question we tried different models (such the ones in Chapter 4 and in Appendix A) until we reached the model proposed in Chapter 5. Our main findings are: first, that we need an external supervision signal. Second, that finding facets in an incremental way is a necessary step in order to find different facets rather than multiple variants of the dominant facet. However, in terms of the classification results, we found in Chapter 6 that the incremental approach is not necessarily suitable for all entity embeddings models.

Research Question 4: *To what extent can a higher-quality representation be obtained by directly learning facet-specific spaces from BoW representations, rather than by decomposing an existing vector space?*

One important point we needed to take into consideration when we formulated a model was scalability. From the experiments in Chapters 4 and 5 we found that the models proposed in those chapters are not sufficiently scalable to deal with large sets of entities, not only because of the use of the MDS but also because of the way the directions were obtained. Therefore, in Chapter 6 we used GloVe to learn the facet-specific embedding, as this model learns the features and the entity representations simultaneously. Another finding from Chapters 4 and 5 is that the assumption that clusters in a word embedding space can describe the facets is too strong; rather, it depends on the type of the clustering algorithm used. As a result, to solve the problem of directly learning facet-specific entity embeddings from the BoW, we employed MoE, using the gating network instead of the clustering algorithm and simultaneously learning the facets using a weighted version of GloVe. Our experiments show that this model was able to identify facets and learn the low-dimensional embeddings for each facet. We evaluated

the model on relatively large datasets to confirm the scalibality of the model.

7.4 Future Work

In this section, we discuss some of the possible ways in which the research in this thesis could be extended further in the future.

Alternative representations

While we have mostly focused on BoW input representations in this thesis, in future work it would be interesting to see how similar strategies could be applied to document embedding methods based on contextualized language models. One state-of-the-art language model is Bidirectional Encoder Representations from Transformers (BERT) [38]. Using BERT to represent documents may be challenging as the documents could be often longer than BERT's maximum input length. The methods proposed in [153, 72] are examples of methods that go beyond sentence-level representation and attempt to learn document representation.

Impact of facet-specific entity embedding on other NLP tasks

For the purposes of this thesis we have evaluated the proposed models only on classification tasks. An important future direction will be to investigate the effectiveness of such representation on other tasks (e.g. entity clustering) and in end-user applications such as recommendation systems, and entity retrieval. Another interesting NLP task that could be improved by using multi-facet embeddings is relation extraction, similar to [116] where they learn, in a supervised way, multi-facet embeddings for sentence patterns using a knowledge base and text corpus, which are then used to infer relations between entity pairs.

Incorporating additional external knowledge

The question of whether employing extra knowledge could improve the discovered facets is worth investigating. One way to do this could be to use structured data extracted from Wikidata; for example, for a given movie, Wikidata contains information

about genre, language, cast members, country of origin, and so on. This investigation could be performed using the method proposed in Chapter 6, as GloVe allows textual description to be combined with other sources of information, as in [69].

Determining the number of facets

One limitation of our work in Chapters 5 and 6 is that the number of facets is arbitrary. Based on tuning the number of facets in our model in Chapter 6, we believe that finding the optimal number of facets will have a positive impact on the classification results. Several methods have been proposed to find the optimal number of clusters that can be used [99, 155], at least as a starting point, to find the optimal number of facets for each dataset.

Further reducing the gap between our representations and conceptual spaces

In our work, all the facets have the same dimensions, whereas in conceptual spaces, facets have different dimensions and different weights. The different dimensions of the facets depend on the information encoded in each facet. The different weights, on the other hand, reflects the degree of importance of each facet for identifying the concept. Determining the dimension of each facet is a challenging yet interesting problem. Some subspace clustering algorithms can deal with subspaces of different dimensions. Although the state-of-the-art subspace clustering algorithms tested in this study were unable to find meaningful facets, we could further investigate them to see how they estimate the number of dimensions of each subspace and whether we can integrate such estimation methods into our proposed models.

Bibliography

[1] T. Ager, O. Kuzelka, and S. Schockaert. Modelling salient features as directions in fine-tuned semantic spaces. In *Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL)*, pages 530–540, Brussels, Belgium, Oct 2018. Association for Computational Linguistics.

[2] C. C. Aggarwal and C. Zhai. *Mining text data*. Springer Science & Business Media, 2012.

[3] C. C. Aggarwal and C. Zhai. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer, 2012.

[4] M. A. Ali, Y. Sun, X. Zhou, W. Wang, and X. Zhao. Antonym-synonym classification based on new sub-space embeddings. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, volume 33, pages 6204–6211, Honolulu, Hawaii, USA, Jan 2019. AAAI Press.

[5] C. Allen and T. M. Hospedales. Analogies explained: Towards understanding word embeddings. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 223–231, Long Beach, California, USA, Jun 2019. Proceedings of Machine Learning Research.

[6] R. Alshaikh, Z. Bouraoui, and S. Schockaert. Learning conceptual spaces with disentangled facets. In *Proceedings of the 23rd Conference on Computational*

Natural Language Learning (CoNLL), pages 131–139, Hong Kong, China, Nov 2019. Association for Computational Linguistics.

[7] R. Alshaikh, Z. Bouraoui, S. Jeawak, and S. Schockaert. A mixture-of-experts model for learning multi-facet entity embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pages 5124–5135, Barcelona, Spain (Online), Dec 2020. International Committee on Computational Linguistics.

[8] R. Alshaikh, Z. Bouraoui, and S. Schockaert. Hierarchical linear disentanglement of data-driven conceptual spaces. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3573–3579, Yokohama, Japan (Online), Jul 2020. IJCAI.org.

[9] D. Anderson. Wordninga: Probabilistically split concatenated words using nlp based on english wikipedia unigram frequencies. `https://github.com/keredson/wordninja`, 2019.

[10] H. Banaee, E. Schaffernicht, and A. Loutfi. Data-driven conceptual spaces: Creating semantic representations for linguistic descriptions of numerical data. *Journal of Artificial Intelligence Research*, 63:691–742, 2018.

[11] M. Baroni, G. Dinu, and G. Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 238–247, Baltimore, Maryland, USA, Jun 2014. Association for Computational Linguistics.

[12] L. Bechberger and K.-U. Kühnberger. A thorough formalization of conceptual spaces. In *Proceedings of the Joint German/Austrian Conference on Artificial Intelligence*, pages 58–71, Dortmund, Germany, Sept 2017. Springer.

[13] E. Bingham and H. Mannila. Random projection in dimensionality reduction: Applications to image and text data. In *Proceedings of the seventh ACM*

SIGKDD international conference on Knowledge discovery and data mining, pages 245–250, San Francisco, CA, USA, Aug 2001. Association for Computing Machinery.

[14] S. Bird and E. Loper. Nltk: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, pages 214–217, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[15] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[16] J. Bolt, B. Coecke, F. Genovese, M. Lewis, D. Marsden, and R. Piedeleu. Interacting conceptual spaces i: Grammatical composition of concepts. In *Conceptual Spaces: Elaborations and Applications*, pages 151–181. Springer, 2019.

[17] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 27th Advances in Neural Information Processing Systems (NIPS)*, pages 2787–2795, Lake Tahoe, Nevada, USA, Dec 2013.

[18] Z. Bouraoui and S. Schockaert. Learning conceptual space representations of interrelated concepts. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1760–1766, Stockholm, Sweden, Jul 2018. IJCAI.org.

[19] Z. Bouraoui, S. Jameel, and S. Schockaert. Inductive reasoning about ontologies using conceptual spaces. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4364–4370, San Francisco, California, USA, Feb 2017. AAAI Press.

[20] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. CRC press, 1984.

[21] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[22] J. Camacho-Collados, M. T. Pilehvar, and R. Navigli. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *Artificial Intelligence*, 240:36–64, 2016.

[23] R. J. Campello, D. Moulavi, and J. Sander. Density-based clustering based on hierarchical density estimates. In *Proceedings of the Advances in Knowledge Discovery and Data Mining, 17th Pacific-Asia Conference (PAKDD)*, pages 160–172, Gold Coast, Australia, Apr 2013. Springer.

[24] R. J. Campello, D. Moulavi, A. Zimek, and J. Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 10(1):1–51, 2015.

[25] A. Chella, M. Frixione, and S. Gaglio. Conceptual spaces for computer vision representations. *Artificial Intelligence Review*, 16(2):137–152, 2001.

[26] A. Chella, M. Frixione, and S. Gaglio. Anchoring symbols to conceptual spaces: the case of dynamic scenarios. *Robotics and Autonomous Systems*, 43(2-3):175–188, 2003.

[27] M. Chen. Efficient vector representation for documents through corruption. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, Apr 2017. OpenReview.net.

[28] T. Q. Chen, X. Li, R. B. Grosse, and D. K. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Proceedings of the Advances in*

Neural Information Processing Systems (NeurIPS), pages 2610–2620, Montréal, Canada, Dec 2018. Curran Associates, Inc.

[29] W. Chen, X. Xie, J. Wang, B. Pradhan, H. Hong, D. T. Bui, Z. Duan, and J. Ma. A comparative study of logistic model tree, random forest, and classification and regression tree models for spatial prediction of landslide susceptibility. *Catena*, 151:147–160, 2017.

[30] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 2172–2180, Barcelona, Spain, Dec 2016. Curran Associates, Inc.

[31] X. Chen, H. Xie, F. L. Wang, Z. Liu, J. Xu, and T. Hao. A bibliometric analysis of natural language processing in medical research. *BMC medical informatics and decision making*, 18(1):14, 2018.

[32] M. A. Cox and T. F. Cox. Multidimensional scaling. In *Handbook of data visualization*, pages 315–347. Springer, 2008.

[33] R. Das, M. Zaheer, and C. Dyer. Gaussian LDA for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 795–804, Beijing, China, Jul 2015. The Association for Computer Linguistics.

[34] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[35] M. Denil, A. Demiraj, N. Kalchbrenner, P. Blunsom, and N. de Freitas. Modelling, visualising and summarising documents with a single convolutional neural network. *CoRR*, abs/1406.3830, 2014.

[36] J. Derrac and S. Schockaert. Inducing semantic relations from conceptual spaces: a data-driven approach to plausible reasoning. *Artificial Intelligence*, 228:66–94, 2015.

[37] A. R. Deshpande and L. Lobo. Text summarization using clustering technique. *International Journal of Engineering Trends and Technology*, 4(8):3348–3351, 2013.

[38] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186, Minneapolis, MN, USA, Jun 2019. Association for Computational Linguistics.

[39] P. Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[40] S. Dreiseitl and L. Ohno-Machado. Logistic regression and artificial neural network classification models: a methodology review. *Journal of biomedical informatics*, 35(5-6):352–359, 2002.

[41] S. T. Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

[42] R. Ebrahimpour, E. Kabir, and M. R. Yousefi. Face detection using mixture of mlp experts. *Neural Processing Letters*, 26(1):69–82, 2007.

[43] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2765–2781, 2013.

[44] B. Esmaeili, H. Wu, S. Jain, A. Bozkurt, N. Siddharth, B. Paige, D. H. Brooks, J. G. Dy, and J. van de Meent. Structured disentangled representations. In

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS), pages 2525–2534, Naha, Okinawa, Japan, April 2019. PMLR.

[45] K. Ethayarajh, D. Duvenaud, and G. Hirst. Understanding undesirable word embedding associations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1696–1705, Florence, Italy, Jul 2019. Association for Computational Linguistics.

[46] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[47] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1606–1611, Hyderabad, India, Jan 2007.

[48] P. Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. MIT Press, 2000.

[49] G. Giguère. Collecting and analyzing data in multidimensional scaling experiments: A guide for psychologists using spss. *Tutorials in Quantitative Methods for Psychology*, 2(1):27–38, 2006.

[50] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, Montréal, Canada. Curran Associates, Inc.

[51] J. W. Grau and D. K. Nelson. The distinction between integral and separable dimensions: Evidence for the integrality of pitch and loudness. *Journal of Experimental Psychology: General*, 117(4):347–370, 1988.

[52] T. L. Griffiths, M. I. Jordan, J. B. Tenenbaum, and D. M. Blei. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural*

Information Processing Systems (NIPS), pages 17–24, Vancouver, Canada, Dec 2003. MIT Press.

[53] I. Güler and E. D. Übeylı. A mixture of experts network structure for modelling doppler ultrasound blood flow signals. *Computers in Biology and Medicine*, 35 (7):565–582, 2005.

[54] J. Guo, W. Che, H. Wang, and T. Liu. Revisiting embedding features for simple semi-supervised learning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 110–120, Doha, Qatar, Oct 2014. Association for Computational Linguistics.

[55] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[56] J. Hartmann, J. Huppertz, C. Schamp, and M. Heitmann. Comparing automated text classification methods. *International Journal of Research in Marketing*, 36 (1):20–38, 2019.

[57] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier. An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 388–397, Vancouver, Canada, Jul 2017. Association for Computational Linguistics.

[58] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, April 2017. OpenReview.net.

[59] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[60] M. C. Hout, M. H. Papesh, and S. D. Goldinger. Multidimensional scaling. *Wiley Interdisciplinary Reviews: Cognitive Science*, 4(1):93–103, 2013.

[61] D.-A. Huang, L.-W. Kang, Y.-C. F. Wang, and C.-W. Lin. Self-learning based image decomposition with applications to single image denoising. *IEEE Transactions on multimedia*, 16(1):83–93, 2013.

[62] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[63] S. Jain, E. Banner, J.-W. van de Meent, I. J. Marshall, and B. C. Wallace. Learning disentangled representations of texts with application to biomedical abstracts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4683–4693, Brussels, Belgium, Oct 2018. Association for Computational Linguistics.

[64] S. Jameel and S. Schockaert. Entity embeddings with conceptual subspaces as a basis for plausible reasoning. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI)*, pages 1353–1361, The Hague, The Netherlands, Aug 2016. IOS Press.

[65] S. Jameel and S. Schockaert. Word and document embedding with vmf-mixture priors on context word vectors. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, pages 3319–3328, Florence, Italy, Jul 2019. Association for Computational Linguistics.

[66] S. Jameel, Z. Bouraoui, and S. Schockaert. Member: Max-margin based embeddings for entity retrieval. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 783–792, Tokyo, Japan, Aug 2017. Association for Computing Machinery.

[67] N. Jaworska and A. Chupetlovska-Anastasova. A review of multidimensional scaling (mds) and its utility in various psychological domains. *Tutorials in quantitative methods for psychology*, 5(1):1–10, 2009.

[68] S. S. Jeawak, C. B. Jones, and S. Schockaert. Using flickr for characterizing the environment: an exploratory analysis. In *Proceedings of the 13th International Conference on Spatial Information Theory (COSIT)*, L'Aquila, Italy, Sep 2017. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[69] S. S. Jeawak, C. B. Jones, and S. Schockaert. Embedding geographic locations for modelling the natural environment using flickr tags and structured data. In *Proceedings of the 41st European Conference on Information Retrieval Research, (ECIR)*, pages 51–66, Cologne, Germany, April 2019. Springer.

[70] V. John, L. Mou, H. Bahuleyan, and O. Vechtomova. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, pages 424–434, Florence, Italy, Jul 2019. Association for Computational Linguistics.

[71] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[72] M. Joshi, O. Levy, L. Zettlemoyer, and D. S. Weld. Bert for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP*, pages 5802–5807, Hong Kong, China, Nov 2019. Association for Computational Linguistics.

[73] K. Kameshwaran and K. Malarvizhi. Survey on clustering techniques in data mining. *International Journal of Computer Science and Information Technologies*, 5(2):2272–2276, 2014.

[74] K.-i. Kanatani. Motion segmentation by subspace separation and model selection. In *Proceedings of the Eighth IEEE International Conference on computer Vision (ICCV)*, pages 586–591, Vancouver, Canada, Jul 2001. IEEE Computer Society.

[75] J. H. Kang, K. Lerman, and A. Plangprasopchok. Analyzing microblogs with affinity propagation. In *Proceedings of the First Workshop on Social Media Analytics*, pages 67–70, Paris, France, Jun 2010. Association for Computing Machinery.

[76] H. Kim and A. Mnih. Disentangling by factorising. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 2654–2663, Stockholm, Sweden, Jul 2018. PMLR.

[77] H. K. Kim, H. Kim, and S. Cho. Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. *Neurocomputing*, 266:336–352, 2017.

[78] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, Banff, Canada, April 2014.

[79] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown. Text classification algorithms: A survey. *Information*, 10(4):150, 2019.

[80] T. M. Lakshmi, A. Martin, R. M. Begum, and V. P. Venkatesan. An analysis on performance of decision tree algorithms using student's qualitative data. *International journal of modern education and computer science*, 5(5):18, 2013.

[81] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1188–1196, Beijing, China. PMLR.

[82] O. Levy and Y. Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 302–308, Baltimore, MD, USA, Jun 2014. The Association for Computer Linguistics.

[83] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems (NIPS)*, pages 2177–2185, Montréal, Canada, Dec 2014. Curran Associates, Inc.

[84] O. Levy, Y. Goldberg, and I. Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225, 2015.

[85] M. Lewis and J. Lawry. Hierarchical conceptual spaces for concept combination. *Artificial Intelligence*, 237:204–227, 2016.

[86] A. Lieto, E. Mensa, and D. P. Radicioni. A resource-driven approach for anchoring linguistic resources to conceptual spaces. In *Proceedings of the Conference of the Italian Association for Artificial Intelligence*, pages 435–449, Genova, Italy, Nov 2016. Springer.

[87] A. Lieto, D. Radicioni, V. Rho, and E. Mensa. Towards a unifying framework for conceptual represention and reasoning in cognitive systems. *Intelligenza Artificiale*, 11(2):139–153, 2017.

[88] A. Lieto, C. Lebiere, and A. Oltramari. The knowledge level in cognitive architectures: Current limitations and possible developments. *Cognitive Systems Research*, 48:39–55, 2018.

[89] J. Lilleberg, Y. Zhu, and Y. Zhang. Support vector machines and word2vec for text classification with semantic features. In *Proceedings of the 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 136–140, Beijing, China, Jul 2015. IEEE Computer Society.

[90] B. Y. Lin, X. Chen, J. Chen, and X. Ren. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages

2829–2839, Hong Kong, China, Nov 2019. Association for Computational Linguistics.

[91] P. Liu, X. Qiu, X. Chen, S. Wu, and X.-J. Huang. Multi-timescale long short-term memory neural network for modelling sentences and documents. In *Proceedings of the 2015 conference on empirical methods in natural language processing (EMNLP)*, pages 2326–2335, Lisbon, Portugal, Sep 2015. The Association for Computational Linguistics.

[92] P. Liu, X. Qiu, and X. Huang. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2873–2879, New York, NY, USA, Jul 2016. IJCAI/AAAI Press.

[93] P. Liu, X. Qiu, and X. Huang. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1–10, Vancouver, Canada, Jul 2017. Association for Computational Linguistics.

[94] F. Locatello, S. Bauer, M. Lucic, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, volume 97, pages 4114–4124, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[95] R. Logan, N. F. Liu, M. E. Peters, M. Gardner, and S. Singh. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5962–5971, Florence, Italy, Jul 2019. Association for Computational Linguistics.

[96] T. Long, R. Lowe, J. C. K. Cheung, and D. Precup. Leveraging lexical resources for learning entity embeddings in multi-relational data. In *Proceedings of the*

54th Annual Meeting of the Association for Computational Linguistics (ACL), pages 112–117, Berlin, Germany, Aug 2016. Association for Computational Linguistics.

[97] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge university press, 2008.

[98] S. Masoudnia and R. Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

[99] M. A. Masud, J. Z. Huang, C. Wei, J. Wang, I. Khan, and M. Zhong. I-nice: A new approach for identifying the number of clusters and initial cluster centres. *Information Sciences*, 466:129–151, 2018.

[100] L. McInnes and J. Healy. Accelerated hierarchical density based clustering. In *Proceedings of the 2017 IEEE International Conference on Data Mining Workshops (ICDM)*, pages 33–42, New Orleans, LA, USA, Nov 2017. IEEE Computer Society.

[101] B. McWilliams and G. Montana. Subspace clustering of high-dimensional data: a predictive approach. *Data Mining and Knowledge Discovery*, 28(3):736–772, 2014.

[102] X. Meng, F. Wei, X. Liu, M. Zhou, S. Li, and H. Wang. Entity-centric topic-oriented opinion summarization in twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 379–387, Beijing, China, Aug 2012. Association for Computing Machinery.

[103] Y. Miao, L. Yu, and P. Blunsom. Neural variational inference for text processing. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, volume 48, pages 1727–1736, New York City, NY, USA, Jun 2016. PMLR.

[104] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations (ICLR)*, Scottsdale, Arizona, USA, May 2013.

[105] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pages 3111–3119, Lake Tahoe, Nevada, USA, Dec 2013. Curran Associates, Inc.

[106] L. Miralles-Pechuán, D. Rosso, F. Jiménez, and J. M. García. A methodology based on deep learning for advert value calculation in cpm, cpc and cpa networks. *Soft Computing*, 21(3):651–665, 2017.

[107] B. Mitra, F. Diaz, and N. Craswell. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pages 1291–1299, Perth, Australia, 2017. International World Wide Web Conferences Steering Committee.

[108] P. Moerland. Some methods for training mixtures of experts. Technical report, IDIAP, 1997.

[109] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[110] E. Nalisnick, B. Mitra, N. Craswell, and R. Caruana. Improving document ranking with dual word embeddings. In *Proceedings of the 25th International Conference Companion on World Wide Web (WWW)*, pages 83–84, Montréal, Canada, 2016. International World Wide Web Conferences Steering Committee.

[111] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[112] M. Nickel and D. Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[113] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean. Zero-shot learning by convex combination of semantic embeddings. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, Banff, Canada, April 2014.

[114] R. M. Nosofsky and T. J. Palmeri. Learning to classify integral-dimension stimuli. *Psychonomic Bulletin & Review*, 3(2):222–226, 1996.

[115] M. H. Papesh and S. D. Goldinger. A multidimensional scaling analysis of own- and cross-race face spaces. *Cognition*, 116(2):283–288, 2010.

[116] R. Paul, H.-S. Chang, and A. McCallum. Multi-facet universal schema. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 909–919. Association for Computational Linguistics, Apr. 2021.

[117] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[118] A. Rangrej, S. Kulkarni, and A. V. Tendulkar. Comparative study of clustering techniques for short text documents. In *Proceedings of the 20th International Conference Companion on World Wide Web (WWW)*, pages 111–112, Hyderabad, India, 2011. Association for Computing Machinery.

[119] S. Rothe and H. Schütze. Word embedding calculus in meaningful ultradense subspaces. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 512–517, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.

[120] S. Ruder, P. Ghaffari, and J. G. Breslin. A hierarchical model of reviews for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processin (EMNLP)*, pages 999–1005, Austin, Texas, USA, Nov. 2016. Association for Computational Linguistics.

[121] G. Salton. Recent studies in automatic text analysis and document retrieval. *Journal of the ACM*, 20:258–278, 1973.

[122] J. Sander, X. Qin, Z. Lu, N. Niu, and A. Kovarsky. Automatic extraction of clusters from hierarchical clustering representations. In *Advances in Knowledge Discovery and Data Mining, 7th Pacific-Asia Conference (PAKDD)*, volume 2637, pages 75–87, Seoul, Korea, 2003. Springer.

[123] K. Saravanan and S. Sasithra. Review on classification based on artificial neural networks. *International Journal of Ambient Systems and Applications (IJASA)*, 2(4):11–18, 2014.

[124] S. Schockaert and J. Derrac. Commonsense reasoning based on betweenness and direction in distributional models. In *Proceedings of the 2015 AAAI Spring Symposia*, Palo Alto, California, USA, March 2015. AAAI Press.

[125] S. Schockaert and H. Prade. Interpolation and extrapolation in conceptual spaces: A case study in the music domain. In *Proceedings of the Web Reasoning and Rule Systems - 5th International Conference, RR*, volume 6902, pages 217–231, Galway, Ireland, Aug 2011. Springer.

[126] V. Shwartz, E. Santus, and D. Schlechtweg. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 65–75, Valencia, Spain, Apr. 2017. Association for Computational Linguistics.

[127] S. Singh and P. Gupta. Comparative study id3, cart and c4. 5 decision tree algorithm: a survey. *International Journal of Advanced Information Science and Technology (IJAIST)*, 27(27):97–103, 2014.

[128] R. A. Sinoara, J. Camacho-Collados, R. G. Rossi, R. Navigli, and S. O. Rezende. Knowledge-enhanced document embeddings for text classification. *Knowledge-Based Systems*, 163:955–971, 2019.

[129] R. Socher, D. Chen, C. D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems (NIPS)*, pages 926–934, Lake Tahoe, Nevada, USA, Dec 2013. Curran Associates, Inc.

[130] M. Steyvers. Multidimensional scaling. *Encyclopedia of cognitive science*, 2006.

[131] B. Tang, M. I. Heywood, and M. Shepherd. Input partitioning to mixture of experts. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, volume 1, pages 227–232, 2002.

[132] D. Tang, B. Qin, and T. Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing (EMNLP)*, pages 1422–1432, Lisbon, Portugal, Sep 2015. The Association for Computational Linguistics.

[133] T. Thongtan and T. Phienthrakul. Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL): Student Research Workshop*, pages 407–414, Florence, Italy, July 2019. Association for Computational Linguistics.

[134] M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482, 1999.

[135] P. D. Turney and P. Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188, 2010.

[136] A. Tversky. Features of similarity. *Psychological review*, 84(4):327, 1977.

[137] E. D. Übeyli, K. Ilbay, G. Ilbay, D. Sahin, and G. Akansel. Differentiation of two subtypes of adult hydrocephalus by mixture of experts. *Journal of medical systems*, 34(3):281–290, 2010.

[138] C. Van Gysel, M. de Rijke, and E. Kanoulas. Learning latent vector spaces for product search. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM)*, pages 165–174, Indianapolis, IN, USA, Oct 2016. Association for Computing Machinery.

[139] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (gpca). *IEEE transactions on pattern analysis and machine intelligence*, 27(12):1945–1959, 2005.

[140] V. K. Vijayan, K. Bindu, and L. Parameswaran. A comprehensive study of text classification algorithms. In *Proceedings of the 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1109–1113, Udupi, India, Sep 2017. IEEE.

[141] B. Wang, A. Wang, F. Chen, Y. Wang, and C.-C. J. Kuo. Evaluating word embedding models: Methods and experimental results. *APSIPA transactions on signal and information processing*, 8, 2019.

[142] D. Wang, C. Ding, and T. Li. K-subspace clustering. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML)*, pages 506–521, Bled, Slovenia, Sep 2009. Springer.

[143] L. Wang. *Support vector machines: theory and applications*, volume 177. Springer Science & Business Media, 2005.

[144] W. Wang, J. Yang, R. Muntz, et al. Sting: A statistical information grid approach to spatial data minin. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB)*, volume 97, pages 186–195, Athens, Greece, Aug 1997. Morgan Kaufmann Publishers Inc.

[145] Z. Wang and Z. Qu. Research on web text classification algorithm based on improved cnn and svm. In *Proceedings of the 2017 IEEE 17th International Conference on Communication Technology (ICCT)*, pages 1958–1961, 2017.

[146] D. Xu and Y. Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, 2015.

[147] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *Proceedings of the 9th European Conference on Computer Vision (ECCV)*, volume 3954, pages 94–106, Graz, Austria, May 2006. Springer.

[148] C. You, D. Robinson, and R. Vidal. Scalable sparse subspace clustering by orthogonal matching pursuit. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3918–3927, Las Vegas, NV, USA, June 2016. IEEE Computer Society.

[149] L.-C. Yu, J. Wang, K. R. Lai, and X. Zhang. Refining word embeddings using intensity scores for sentiment analysis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(3):671–681, 2017.

[150] Z. Yu, H. Wang, X. Lin, and M. Wang. Learning term embeddings for hypernymy identification. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 1390â1397, Buenos Aires, Argentina, 2015. AAAI Press.

[151] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM*

SIGKDD international conference on knowledge discovery and data mining, pages 353–362, San Francisco, CA, USA, Aug 2016. Association for Computing Machinery.

[152] L. Zhang, S. Zhang, and K. Balog. Table2Vec: Neural word and entity embeddings for table population and retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1029–1032, Paris, France, Jul 2019. ACM.

[153] X. Zhang, F. Wei, and M. Zhou. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5059–5069, Florence, Italy, Jul 2019. Association for Computational Linguistics.

[154] R. Zhao and K. Mao. Fuzzy bag-of-words model for document representation. *IEEE Transactions on Fuzzy Systems*, 26(2):794–804, 2017.

[155] S. Zhou, Z. Xu, and F. Liu. Method for determining the optimal number of clusters based on agglomerative hierarchical clustering. *IEEE Transactions on Neural Networks and learning systems*, 28(12):3007–3017, 2016.

Appendices

A Learning Facets as Low Dimensional Subspaces Using GloVE

A.1 Introduction

In our initial experiments, we used the GloVe model to learn the full space and simultaneously learn a low-rank projection matrix for each facet. We tried to learn a vector space with k facets using a variation of GloVe. Our intuition was that each of the facets would correspond to a low-dimensional subspace.

A.2 Model Description

We write R_l for a projection matrix corresponding to facet l. In other words, $w_i R_l$ is the representation of the entity i according to facet l. For the facets to correspond to low-dimensional subspaces, the matrix R_l has to be of low rank. This was imposed in the model using nuclear norm regularization (as minimizing the nuclear norm is computationally easier than minimizing the rank).

$$G = \sum_{j=1}^{m} \phi(G_{j1}, ..., G_{jk}) + \lambda \sum_{l=1}^{k} \|R_l\|_* \tag{1}$$

where

$$G_{jl} = S\left(\sum_{i:x_{ij}>0} f(x_{ij})\left(\mathbf{e_i R_l} \cdot \tilde{\mathbf{w}}_\mathbf{j}\mathbf{R_l} + b_i + \tilde{b}_j - \log x_{ij}\right)^2\right) \tag{2}$$

where S is the sigmoid function and ϕ is a differentiable approximation to the minimum. we used:

$$\phi(G_{j1}, ..., G_{jk}) = (G_{j1}^{-p} + ... + G_{jk}^{-p})^{-\frac{1}{p}} \tag{3}$$

Here $\lambda > 0$ is a parameter determining to what degree we want to force the aspects to be low-dimensional. The value of p reflects how closely we want to approximate the minimum. The larger the value of p, the more the model will behave as if there is a minimum. If the p is too large, this will make it more difficult to find the minimum of the objective function G, which could lead to weak performance. If the p is too small it may mean that facets are not obtained, i.e. that the model attempts to model each feature to a small degree in each of the facets, rather than picking one facet for each property.

A.3 Model Limitations

The limitations with this model are as follows. (1) The model is very sensitive to the parameters λ and p, and it was very hard to chose the right values even for a toy example. (2) The decision to assign one feature to each facet was very shallow for most of the features, i.e. the differences between $G_{j1}, ..., G_{jk}$ were very small. (3) The convergence of the model was very slow. (4) The results that we obtained were not promising. Table 1 shows examples of the facets we obtained from this model. After trying out many variations of this model, the main problem we faced was how to force the model to assign words to facets and learn the entity embeddings simultaneously. We found that the mixture of experts (MoE) architecture allowed us to distribute these two tasks to two components – the gating networks and the experts – in a way that encouraged each expert to specialize on one facet.

MOVIES			
Facet 0: kids, shes, copy, comedy, feels, believable, rent, kill, message, horror, terrible, office, tone, reviewers, incredible, younger, ridiculous, boys, atmosphere, dvds, visual, smart, disc, adventure, impressed, difference, driven, unexpected, police, journey, grade, lame, bits, widescreen, political, mystery, crap, tears, tragic, likable, humanity, subtitles, ages, library, digital, cult, shocking	**Faet 1:** hell, predictable, missed, narrative, charming, finish, tough, documentary, positive, cheap, tension, situations, wasted, favorites, bother, theatre, scenery, nudity, comedic, french, intelligence, majority, teenage, mom, computer, christmas, americans, fighting, flying, teens, learned, changed, development, sat, skip, portrait, blu, humans, edge, gripping, funnier, lol, bluray, previews, cop, silly	**Facet 2:** superb, open, product, awful, sweet, honest, subject, romantic, portrayal, biggest, concept, exciting, critics, award, parents, sexual, cute, situation, information, creative, bunch, killing, edition, charm, twists, chemistry, wondering, issue, books, latest, warm, summer, refreshing, cry, mysterious, creepy, comedies, artistic, design, thrown, epic, bonus	**Facet 3:** amazon, effects, storyline, language, budget, opinion, actress, relationship, glad, oscar, emotional, compelling, issues, fascinating, masterpiece, filmmakers, girls, society, crazy, loud, premise, complex, earth, included, stunning, suspense, throw, game, super, adult, romance, touching, culture, details, intelligent

Table 1: Examples of the facets from the *Movies* dataset

B Basic Machine Learning Algorithms

Machine learning involves a set of algorithms that can learn and infer from data [109]. Machine learning plays a prominent role in many applications in different domains, including NLP, finance, computer vision, information security and many others. Its approaches can be broadly grouped into supervised, unsupervised, and reinforcement learning algorithms. In supervised learning, the algorithms learn from training data and generalize what has been learned to unseen data. Unsupervised machine learning is applied to unlabelled data to uncover its hidden structure. Reinforcement learning algorithms depend on training the model to make decisions through trial and error until a suitable solution is found. There are rewards for right decisions and penalties for wrong ones, and the model tries to maximize the rewards until it reaches the optimal solution. In this thesis, we use only supervised and unsupervised machine learning algorithms, therefore, only these are reviewed. We focus on explaining in detail the algorithms used, and we briefly explain the ensemble learning techniques that will be used to enhance the performance of the supervised and unsupervised algorithms in this thesis.

B.1 Unsupervised Machine Learning

Unsupervised machine learning algorithms can be categorized into different groups based on the task they perform, e.g. clustering, association rule mining, and density estimation. We will focus only on clustering.

B.1.1 Clustering Algorithms

Clustering algorithms aim to discover the hidden structure of the data by grouping similar unlabelled data points based on certain similarity metrics. Clustering is a useful approach in many applications in text mining, such as document organization [118]

and text summarization [37]. Generally, clustering algorithms can be categorized as follows [73, 146]:

Distribution-based algorithms

The assumption behind this type of method is that the data points in the cluster have been generated from a particular kind of distribution, e.g. Gaussian distribution. Gaussian mixture model clustering is a type of distribution-based method.

Centroid-based algorithms

The algorithm iteratively determines the centroid of each cluster, and the assigning of a data point to a certain cluster depends on how far this data point is from the center of the cluster. k-means and affinity propagation fall into this category.

Hierarchical clustering algorithms

At different distance scales, different clusters can be obtained; these clusters form a tree-like structure. This tree can be visualized using a dendrogram, which gives an insight into the number of clusters and the distance cut-off needed to obtain flat clusters. Agglomerative clustering is one example of hierarchical clustering.

Density-based algorithms

In this category, the cluster is defined as a set of data points that belong to the same dense region in the data space. DBSCAN and HDBSCAN are examples of these methods.

Grid-based algorithms

The algorithm is based on dividing the data space into a number of cells and then testing the density of each cell. If the cell is not dense enough (based on a threshold), the algorithm eliminates that cell. A cluster is formed from adjacent cells of high density. STING [144] is an example within this category.

Graph-based algorithms

These methods partition a set of graphs into clusters (between-graph) or to partition the nodes in one graph into clusters (within-graph). Kernel k-means falls under this

category.

In this research, we rely on several different clustering algorithms to discover initial facets in two of our proposed methods to learn facet-specific entity embeddings. In the rest of this section, we explain these clustering algorithms in more detail.

- **Affinity Propagation** [46] is a flat non-parametric (i.e. the number of clusters does not needs to be predetermined) centroid-based model. Affinity propagation takes a similarity matrix as input and tries to find exemplars of the data (i.e. representative examples of each category) and then considers these exemplars as centroids of the clusters. To find the clusters, the algorithm groups the data points in a way that maximizes the sum of the similarities between the data points and their exemplars. Messages are exchanged between data points in order to find the best set of exemplars. These messages are real values that encode: (i) the ability of the data point to be an exemplar extracted from the similarity matrix, (ii) evidence that this point is a suitable exemplar to another data point, extracted from a responsibility matrix, and (iii) the availability of this data point to be an exemplar, from the availability matrix.
  Let us write $\{1, ..., N\}$ to denote the data points, and let S be a given similarity matrix. The elements $s(i, k)$ of this matrix are real values that indicate the similarity between the data points i and k, i.e. to what extent the data point k is suitable to be an exemplar for the data point i. For each data point k, the value $s(k, k) = p$, where p is called the global preference. If p is high, then the probability that k is chosen as an exemplar is high. The preference aims to control the number of clusters (a large preference leads to a large number of clusters) and to avoid the trivial solution of choosing all the data points as exemplars. As a first step, each point is represented as a node in a network (where the messages are transmitted through the network edges) and initializes the so-called responsibility matrix R and the availability matrix A to zero. Second, it updates R and

A based on equations (4) and (5) until the models converge.

$$\forall\, i,k : r(i,k) = s(i,k) - \max_{\{k':k'\neq k\}} [s(i,k') + a(i,k')] \tag{4}$$

$$\forall\, i,k : a\,(i,k) = \begin{cases} \sum_{i':i'\neq i} \max\,[0, r\,(i',k)] & \textit{for } k = i \\ \min\left[[0, r\,(k,k)] + \sum_{i':i'\notin i,k} max\,[0, r\,(i',k)]\right] & \textit{for } k \neq i \end{cases} \tag{5}$$

By the end, a final set of exemplars K is determined, such that $K = \{\ k \mid r\,(k,k) + a\,(k,k) > 0\}$. The rest of the data points are then assigned to the most similar exemplar based on the similarity matrix.

Affinity propagation is used in a variety of areas, including text analysis [75, 102], medical research [31], and computer vision [61]. In this research, we have chosen affinity propagation over other algorithms (such as k-means), because of its stability and because finding the right preference is often easier than determining the number of clusters.

- **HDBSCAN** [24] is a hierarchical density-based clustering algorithm that defines the clusters as the dense regions in the data, so that the clusters can have arbitrary shapes, unlike in affinity propagation, where the clusters are always spherical. HDBSCAN, unlike k-means and affinity propagation, tries to eliminate the outliers and does not assign these points to any cluster. The initial step of the algorithm is to search through the data for areas of high density isolated among low-density areas. To do this, the core distance for each point x is defined, denoted as $core_k(x)$, which means the distance of the point x to its k-nearest neighbors, where k is a parameter. Then, to prevent the outliers becoming linkages between two dense regions and forming one cluster, another distance metric between data points, called mutual reachability distance, is defined. For two points x_1 and x_2, this can be written as follows:

$$\max\{core_k(x_1), core_k(x_2), d(x_1,x_2)\} \tag{6}$$

where $d(x_1, x_2)$ is the distance between the two points based on a chosen distance metric, such as Euclidean distance. Following this, a weighted graph is created, where the data points are represented as vertices, and the edges between the points are the mutual reachability distance values. The aim of the algorithm is then to obtain a hierarchy, from fully connected components to fully disconnected components. The efficient way to accomplish this is by finding the minimum spanning tree, which is a minimal set of edges that fulfil two conditions: 1) removing any edge from the set leads to disconnection of components, and 2) there are no edges connecting the vertices of lower weight than the ones in this set. The top-down hierarchy tree is then made by sorting the edges in ascending order. Finally, the algorithm starts to iteratively split the tree into clusters, only if the clusters have at least n points in them. The main disadvantage of HDBSCAN is that it is computationally expensive and does not scale to large datasets [100].

- **Agglomerative clustering** [2] is a bottom-up hierarchical clustering algorithm. It starts by considering each point as an individual cluster, then merges these clusters based on a proximity matrix. The algorithm steps are as follows: 1) each point in $X = \{x_1, x_2, ..., x_n\}$ is transformed to a singleton cluster to form a collection of clusters $C = \{c_1, c_2, ...c_n\}$; 2) the proximity matrix between the clusters is calculated; 3) a pair of clusters c_i and c_j is merged to create a new cluster c_{ij} if they are the nearest neighbors, i.e. $D(c_i, c_j)$ is the smallest distance between all the possible pairs, where D is the distance function defined by the proximity matrix; 4) after each merging step, the proximity matrix is updated to calculate the distance between the updated collocation of the clusters C; 5) steps 3) and 4) are iteratively repeated until all the clusters are merged into one large cluster.

  There are different approaches to calculating the distances between the clusters, including:

  - Single linkage: define the distance between two pairs of clusters c_i and c_j

as the minimum distance of all the pairs of data points $(x, y) \in c_i \times c_j$.

- Complete linkage: define the distance between two pairs of clusters c_i and c_j as the maximum distance of all the pairs of data points $(x, y) \in c_i \times c_j$.
- Average linkage: define the distance between two pairs of clusters c_i and c_j as the average distance of all the pairs of data points $(x, y) \in c_i \times c_j$.
- Ward: define the distance between two pairs of clusters c_i and c_j as the sum of the square distance of all the pairs of data points $(x, y) \in c_i \times c_j$.

Using any of these approaches, agglomerative clustering merges the clusters with the lowest distance, i.e. highly similar clusters. The resulting hierarchy can be visualized as a dendrogram. Based on the dendrogram, we still need to determine the number of clusters. There are two ways to extract clusters using the dendrogram, either by manually selecting the clusters from the dendrogram or by defining a horizontal cut-off on the dendrogram, after which the clusters above this cut-off are extracted [122]. Agglomerative clustering is used in many research areas, particularly in text mining [2] (e.g. document clustering [97] and topic extraction [118]).

Quantifying the performance of the clustering algorithm and measuring the quality of the resulting clusters is not as easy as evaluating the performance of classification algorithms. This evaluation takes into consideration how similar the resulting clusters are to certain ground truth classes or whether these clusters satisfy particular assumptions. When the ground truth is given, different metrics can be used, such as homogeneity score (used to check whether all the data points in the cluster belong to a single class) and completeness score (used to evaluate to what extent all the members of a single class end up in one cluster). Measures that can be used when there are no ground truth labels include the silhouette coefficient and the Calinski–Harabasz index. In this thesis, we use clustering algorithms in two of our proposed methods and mostly evaluate the clusters based on our definition of the facets and our knowledge of the domain.

B.2 Supervised Machine Learning

The idea behind supervised learning is to learn by example. The algorithm takes a set of example pairs (X,Y) and tries to build a model that learns how to map X to Y, i.e. $Y = f(X)$. This model is then used to predict the output for any new input value. Supervised learning tasks can be categorized as either regression or classification. In regression, the output is a value from an ordered set (typically real numbers), while the output in classification is categorical. Classification problems can be binary (when there are only two classes to predict), multiclass (if the number of classes is more than two), or multilabel (when each example can belong to multiple classes).

As this thesis uses classification algorithms to evaluate the discovered facet-specific entity embeddings, we will focus on explaining some of the basic, yet extensively used, classification algorithms.

B.2.1 Basic Classification Algorithms

- **Decision Tree** [2] is a non-parametric supervised machine learning method that can be used to solve both classification and regression problems. The decision tree aims to predict the class or the value of the variable by building a training model in the form of a tree, where each node of the tree is associated with a feature/attribute and a value for that feature. Decisions are then taken based on a set of rules induced from the training data. The resulting tree consists of decision nodes, which have two or more branches, and leaf nodes, which represent the outcome. The first decision node is called the root of the tree. The model splits the data into homogeneous subsets; these splits are decided based on impurity measures that measure the homogeneity of each sample of the data. The model uses impurity measures to decide where each different attribute of the data fits, i.e. as a root node or at a certain level of the tree. There are many measures of impurity, such as information gain, entropy, Gini and reduction in variance. Here,

we will focus on Gini and entropy, as they are commonly used for categorical variables.

Gini, or the Gini index, is the probability of misclassifying a randomly selected instance from the data if the label of that instance were randomly assigned based on the distribution of the class labels in the dataset. The higher the Gini probability, the lower the homogeneity. **Entropy** is used to measure the disorder in the information gained by performing a certain split. The higher the entropy, the lower the homogeneity. Due to its simplicity, Gini is more efficient than entropy in terms of the computational cost.

The different types of decision tree algorithms include ID3, C4.5 and classification and regression trees (CART) [127]. The selection of the algorithm depends on the type of data (continuous or discrete) and the type of classification problem (binary or multiclass); this thesis uses the CART algorithm. **CART** [20] can handle categorical and numerical variables. It builds a binary tree in which each node has only two edges, and it usually uses the Gini impurity metric to select which attribute of the data should be used by the root node, repeating the same step to choose the attributes for the other nodes. It can also be used for regression, where it generates a regression tree to predict a real value instead of a class. One of the main advantages of CART is that it can identify the most important features, using cost complexity pruning to cut the non-informative branches of the tree [80, 127]. In general, decision tree learning is fast, as it does not include complex computation; however, it can easily overfit and is very sensitive to noise [79, 140].

- k**-Nearest Neighbors (kNN)** [79, 97] is another non-parametric supervised machine learning method that can be utilized to solve both classification and regression problems. kNN is one of the simplest machine learning algorithms, which does not need any prior knowledge about the distribution of the data to make decisions and does not learn any model to generalize it; instead, it stores all the training examples to be used during the testing phase. To classify a new ex-

ample, kNN measures the distance between this example and the examples from the training set, then chooses the nearest k neighbors to that example; the mode of its labels is then the class of the new example, i.e majority voting. Another possible voting scheme is to give more weight to closer neighbors, which reduces the impact of the dominant class in comparison to in majority voting. In the case of regression, the mean of the K labels is the outcome. Despite the simplicity of kNN, it is computationally costly when applied to large training set, because it stores the entire training set [56]; yet, large numbers of training examples are needed, as kNN suffers from over-fitting when applied to high-dimensional data [56, 39].

- **Support Vector Machine (SVMs)** [2] is a supervised machine learning method that can be utilized to solve both classification and regression problems. When using SVM, each example from the training set is represented as (x, y), where x is a feature vector representation of the example, and y is the class label. Then, the method tries to find a hyperplane in that space that separates points from different classes by maximizing the margin, i.e. the perpendicular distance between that hyperplane and the closest points to it from both sides; these points are called support vectors. In a linear SVM, the hyperplane can be written as:

$$w \cdot x + b = 0 \tag{7}$$

where $w = (w_1, w_2, ..., w_n)$ is a weight vector and b is a bias. SVM can also handle non-linear classification problems by projecting the training into a higher-dimensional space where the different classes can be linearly separable. This projection is performed via kernel functions, such as Gaussian kernel or polynomial kernel [143, 55]. One of the main advantages of SVM, especially linear SVM, is that it is less prone to overfitting the training data than many other methods [56] and can perform well with few training examples [97]. SVM is widely used in text classification [89, 145] or to evaluate text representation models and document embeddings [69].

- **Logistic Regression Linear Classifier** [40, 2] is a supervised machine learning model widely used and studied in different domains [59, 29]. Logistic regression predicts discrete values only; for example, it can predict whether a student will pass or fail (a binary classification problem) or predict the type of an animal (a multiclass classification problem), unlike linear regression, which predicts continuous values [55]. Logistic regression is a linear method [3], which produces predictions that are then transformed to a probability estimation between 0 and 1, using the sigmoid function. The sigmoid function maps real values to values between 0 and 1, and can be written as:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{8}$$

where e is the base of the natural logarithms, z is the model prediction of the data, which is the linear combination of x (the input values), w (the coefficients vector), and b (the bias).

$$z = wx + b \tag{9}$$

The resulting probability estimation is then mapped to a class label based on a selected threshold called the decision boundary. For example, in a binary classification task, if the threshold is 0.5, then the outcome will be as follows:

$$p \geq 0.5, \mathit{class} = 1$$
$$p < 0.5, \mathit{class} = 0$$

- **Artificial Neural Network (ANN)** [40, 2] is a machine learning model widely used in supervised learning. The ANN is inspired by the way the human brain works. It consists of a collection of neurons organized in layers, where each neuron takes an input and processes it (e.g. applies a non-linear function) and then passes the output to the neurons in the next layer; this is known as feed-forward. Each input in the network is weighted, and this weight is tuned during the training phase. To train a neural network on a classification problem, for example, the output of the model is compared with the desired output; then,

to reduce the error in the output, a backpropagation algorithm is used to adjust the weight of each node in the network. Different architectures exist, but the most commonly used architectures for text classification are convolutional neural networks (CNNs) and recurrent neural networks (RNNs). ANNs intuitively combine two steps: learning meaningful representations from raw inputs and making predictions from the resulting representations. Since our focus in this thesis is on unsupervised strategies for learning meaningful entity representations, we will rely on simple classification algorithms rather than ANNs. Our setting is particularly suitable for cases where only limited amounts of training examples are available, in which case training a deep neural network is typically not feasible. Moreover, ANN models are less efficient than other simple classification algorithms, require more effort to fine-tune the hyperparameters, and are hard to interpret [123, 106, 3].

B.2.2 Classification Evaluation Metrics

Many metrics can be used to evaluate the performance of classifiers; the most popular are accuracy, precision, recall, the confusion matrix, and F1-score.

The confusion matrix is used to measure the accuracy of the classifier by counting the correctly classified and misclassified instances for each class. Table 2 shows the structure of a confusion matrix for a binary classification task.

Table 2: Confusion matrix for a binary classification task.

	Predicted label	
Actual label	Positive	Negative
Positive	True Positives	False Negatives
Negative	False Positives	True Negatives

True positives (TP) are the cases where the actual labels and the model's predictions of the examples are both positives; true negatives (TN) are the cases where both the actual

labels and the model's predictions of the examples are negatives; false positives (FP) are the cases in which the actual labels of the examples are negatives and the predicted labels are positives; false negatives (FN) are the cases in which the actual labels of the examples are positives and the predicted labels are negatives.

Accuracy is the ratio of correct predictions to all predictions made by the classifier and can be defined as:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{10}$$

Accuracy is a good metric only if the classes are balanced.

Precision is the proportion of correctly predicted positive data points to total predicted positive data points. Precision is a useful metric when the FP is very costly, for instance, when classifying non-spam email as a spam, which could lead to the loss of important emails.

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

Recall is the proportion of correctly classified positive data points to all the data points in the actual class, used when the cost of the FN is very high.

$$Recall = \frac{TP}{(TP + FN)} \tag{12}$$

F1-score is the harmonic mean of precision and recall, used when the classes are imbalanced; this is usually the case in binary classification tasks. Thus, in this thesis, F1-score is used to evaluate all the binary classification tasks employed to evaluate our proposed methods.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{(Precision + Recall)} \tag{13}$$

Cohen's kappa coefficient is a metric that is used mostly in psychology to measure the agreement between two raters. In machine learning, Cohen's kappa score is used to measure the classifier performance, particularly when the classes are imbalanced. The two raters in the case of a classification problem are the observed accuracy (Accuracy) on one hand and the expected accuracy (ExAccuracy) on the other hand. It basically

compares the performance of the classifier with the performance of a classifier that chooses the class randomly. Using the confusion matrix, the Kappa score is calculated through the following steps. First, the Accuracy is calculated; this is the total number of data points classified correctly by the classifier for all the classes divided by the total number of data points using Equation 10. Second, the ExAccuracy is calculated; this is the the accuracy that can be achieved by any random classifier and can be calculated as follows:

$$ExAccuracy = \frac{((TP + FN) * (TP + FP) + (FP + TN) * (FN + TN))}{(TP + TN + FP + FN)^2} \quad (14)$$

Finally, the Cohen's kappa is obtained using the following formula:

$$Kappa = (Accuracy - ExAccuracy)/(1 - ExAccuracy) \quad (15)$$

B.3 Ensemble Learning

Ensemble learning is a method in which multiple machine learning models, such as classifiers, and clustering algorithms, are combined to solve a particular task, to improve the model's predictivity. These combined models can use the same learning algorithm, for example, multiple kNN classifiers, or different learning algorithms. We will now explain some widely known ensemble learning techniques.

B.3.1 Ensemble Learning Techniques

Bagging: is a non-trainable ensemble learning technique that tries to prevent the model from overfitting on the training data, i.e. reduces the variance error, by dividing the dataset into smaller sample populations and then training one of the base learners on each sample. The final output will be based on averaging the outcomes from the base learners for regression, while in classification problems, the outcome will be based on voting.

Boosting: is a non-trainable technique where the base learners are constructed sequentially to allow each learner to benefit from the previous learner by giving a higher weight to the instances mislabelled by the previous learner and decreasing the weight for those correctly classified. The final prediction is a weighted sum in regression tasks and the majority vote in classification tasks.

Stacking: Unlike in boosting and bagging, the final outcome in the stacking technique is determined by another learner, called the meta-learner (meta-classifier or meta-regressor). The meta-learner is also a machine learning algorithm; it takes the predictions of the base learners as the input and then gives the final prediction. The training data is split into two sets; the first set is used to train the base learners, and the second is then used for testing those learners and making the predictions. Finally, the meta-learner is trained on these predictions and makes the final decision.

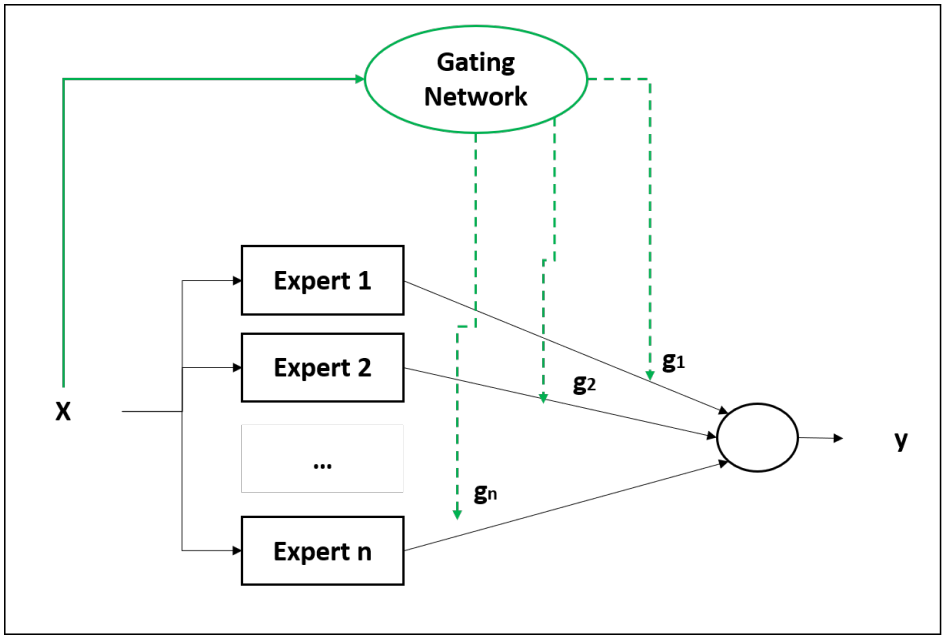


Figure 1: General Architecture of Mixture of Experts.

Mixture of experts (MoE): is a trainable ensemble learning technique that combines

several models, where each model specializes in one aspect of the input space [62]. The model achieves this by training a neural network that performs a soft partition on the input space, called *a Gating Network*; then, for each partition class of the space, a separate neural network model, i.e. expert, is trained. Many ways of dividing the data between the experts have been suggested [98]. In the traditional MoE, the gating network's role is to evaluate how each expert learns the prediction for each instance of the data, simultaneously deciding how to implicitly segment and distribute data between the experts when training them [137, 62]. Other types of MoE explicitly partition the feature space using a clustering approach before starting to train each expert in one of the resulting subspaces [131]. The explicit decomposition of the feature space makes the responsibilities of each expert much clearer, and deciding the number of experts is straightforward, as this is the same as the number of clusters. However, this type of MoE only considers the nature of the task, while the MoE of implicitly localized experts considers both the task and the experts' performance. Moreover, the lack of interaction between experts in explicit decomposition may lead to an incomplete solution of the task [98]; thus, in this thesis, we only use the implicit approach.

To illustrate how the traditional MoE works [108], Figure 1 shows the architecture of the MoE, consisting of n experts and a gating network in which all experts receive the same input x. The gating network gives probabilities $\{g_1, g_2, ..., g_n\}$, where g_i is the probability that the ith expert is competent for this input. The architecture of the experts and the gating network depends on the task itself and is usually either a linear model or a multilayer neural network:

$$y(x) = \sum_{i=1}^{n} g_i(x)(y_i(x)) \tag{16}$$

where $y(x)$ is the weighted mean of the experts' output, and $g_i(x)$ is the weight given by the gating network for the ith expert. The probability g_i is then obtained using a softmax function:

$$g_i(x) = \frac{\exp(z_i)}{\sum_{j=1}^{n} \exp(z_j)} \tag{17}$$

where z_i is the output of the linear model or the multilayer neural network. The gating network will then learn $g_i(x)$ based on the ith expert's performance in modeling x.

One way to train the MoE is using gradient-descent-based training algorithms; for example, in [42], backpropagation was used to train the MoE to enhance face-detection accuracy. The EM algorithm is another way to train the MoE [53, 71]. EM is an iterative method for finding maximum likelihood estimates for a model's parameters in the presence of missing data points or hidden latent variables. EM performs two steps repeatedly: the E-step (where the algorithm estimates the hidden latent variables) and the M-step (where the estimation of the hidden latent variables is used to find the parameters of the model that maximize the likelihood of the data). The way EM works makes it fit with the MoE structure, as it decouples the parameter estimations for the two components of the MoE model [108, 98].