Disentangling low-dimensional vector space representations of text documents

A thesis submitted in partial fulfilment

of the requirement for the degree of Doctor of Philosophy

Thomas J. Ager

January 2020

Cardiff University
School of Computer Science & Informatics

Declaration

This thesis is the result of my own independent work, except where otherwise stated, and the views expressed are my own. Other sources are acknowledged by explicit references. The thesis has not been edited by a third party beyond what is permitted by Cardiff University's Use of Third Party Editors by Research Degree Students Procedure.

Signed ..... (candidate)

Date .....

Statement 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of PhD.

Signed ..... (candidate)

Date .....

Statement 2

This thesis is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by explicit references.

Signed ..... (candidate)

Date .....

Statement 3

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate)

Date .....

Word Count .....

(Excluding Abstract, acknowledgements, declarations, contents pages, appendices, tables, diagrams and figures, references, bibliography, footnotes and endnotes)

To my parents.
for their invaluable and unwavering support.

Abstract

In contrast to traditional document representations such as bags-of-words, vector space representations that are currently most popular tend to be lower-dimensional. This has important advantages, e.g. making the representation of a given document less dependent on the exact words that are used in its description. However, this comes at an important cost, namely that the representation is "entangled", meaning that the features of the representation are not individually meaningful or interpretable. The main aim of this thesis is to address this problem by disentangling vector spaces into representations that are composed of individual, important and meaningful features in the given domain. In a domain of IMDB movie reviews, where each document is a review, the disentangled feature representation would be separated into features that describe how "Scary", "Romantic", ..., "Comedic", the movie is. This thesis builds on an initial approach introduced by Derrac [19], where it was observed that low-dimensional vector space representations of documents often model important features of the domain as directions. The method begins by obtaining direction vectors for all terms using a linear classifier to find a hyper-plane that separates entities that have a term and do not have a term. Then, from each hyper-plane the orthogonal vector is taken as a direction that goes from documents that least have the word (as they are furthest from the hyper-plane on the negative side) to documents that most have it (as they are furthest from the hyper-plane on the positive side). The importance of these terms is determined by how well the linear classifier performs on a standard classification metric, e.g. accuracy, which approximates how linearly separable the entities are that contain the term in the vector space. The most important terms are taken, and documents are ranked on these directions by calculating the dot product between the orthogonal vector to the hyper-plane and each document vectors. This results in a value is obtained that corresponds to how far up documents are on the direction, enabling documents to be ranked on how much they 'have' a particular meaningful feature, e.g. movies can be ranked on how "Scary" they are.

The work by Derrac [19] obtained semantic features from Multi-Dimensional Scaling (MDS) document embeddings and validated their work by classifying entities using a rule-based classifier (FOIL), resulting in rules of the form "**IF** *x* is more scary than most horror films **THEN** *x* is a horror film.". The first main contribution of this thesis is a deeper investigation and improvement of this method, in particular by experimentally validating that the features derived from the initial embeddings are clear and semantic across a variety of unsupervised document embedding models and domains. Variants of the method are introduced that are shown to improve performance, and the results are quantitatively tested using low-depth decision trees.

Neural network architectures have advanced to the state-of-the-art in many tasks, following the idea that their hidden layers can be viewed as document embeddings, the second main contribution of the work is in deriving semantic features from the hidden layers of neural networks. In particular, a qualitative analysis of two kinds of neural networks is conducted, feed-forward networks and stacked auto-encoders. Auto-encoders are stacked by using their hidden-layer as the input to another auto-encoder. Characteristics of learned feed-forward networks are identified, in particular that they can learn higher-quality representations of features that are relevant to the considered classification problem. In the case of auto-encoders, it is found that as auto-encoders are stacked, increasingly abstract features can be derived from them. For example, in the initial auto-encoder layers, features like "Horror" and "Comedy" can be separated well by the linear classifier, meanwhile in the latter layers features like "society" and "relationships" are easy to separate. After identifying directions that model important features of documents in each stacked auto-encoder, symbolic rules are induced that relate specific features to more general ones. These rules are interesting, as they can be used to produce a domain theory of the domain where abstract terms are explained in terms of specific ones, for example:

```
IF Emotions AND Journey THEN Adventure          (1)
```

In addition to introducing variants of the method in [19], this thesis also introduces an additional post-processing step for improving disentangled feature representations by fine-tuning the original embedding to prioritize faithfully modelled directions. Methods for learning document embeddings are mostly aimed at modelling similarity. It is found that there is an inherent trade-off between capturing similarity and faithfully modelling features as directions. Following this

observation, a simple method to fine-tune document embeddings is proposed, with the aim of improving the quality of the feature directions obtained from them. Crucially, this method is also fully unsupervised, requiring only a bag-of-words representation of the objects as input. In particular, clusters of terms are identified that refer to salient properties of the considered domain, and a simple neural network model is used to learn a new representation in which each of these properties is more faithfully modelled as a direction. It is found that in most cases this method improves the ranking of entities, and results in increased performance when disentangled feature representations are used as input to classifiers.

Overall, this thesis quantitatively and qualitatively validates that disentangled feature representations of meaningful features can be derived from low-dimensional vector spaces of documents, across a variety of domains and document embedding models. New variants of the method introduced by Derrac [19] are evaluated and found to be effective. Neural networks are investigated, and in the case of stacked auto-encoders, features become more abstract as auto-encoders are stacked, and a method to obtain a domain theory that links specific properties to more general ones is introduced. In the case of feed-forward networks, a variety of interesting characteristics are observed, in particular that feed-forward networks make features that are more relevant to the task more separable. Finally, a new method is introduced to fine-tune existing document embeddings such that their features are more meaningful.

Acknowledgements

It has been a long journey from the start of this PhD to the end of it, and I owe almost all of my success to my supervisor Steven Schockaert, and my unofficial co-supervisor Ondrej Kuzelka. I would like to thank them especially for all their guidance and their many ideas about the work undertaken in the thesis.

I would also like to thank my colleagues at Cardiff University, who provided a community to engage with when I needed an audience for my presentations, or to discuss my thoughts about the work. It is certainly in large part thanks to them that I was able to be where I am today.

My deepest thanks go to my parents, who were able to support me through the final years of my PhD. Finally I thank God, who guided me when I needed it most.

Contents

List of Publications

The work introduced in this thesis is based on the following publications.

- Ager, T., Kuzelka, O., Schockaert, S. (2018). Modelling salient features as directions in fine-tuned semantic spaces. CoNLL 2018 - 22nd Conference on Computational Natural Language Learning, Proceedings, 530-540. https://doi.org/10.18653/v1/k18-1051
- Ager, T., Kuzelka, O., Schockaert, S. (2016). Inducing symbolic rules from entity embeddings using auto-encoders. CEUR Workshop Proceedings, 1768.

List of Figures

List of Tables

Chapter 1

Introduction

Applications that enable user-generated content e.g. Wikipedia, Social Media sites (Facebook, Twitter), Product and movie review sites (IMDB, Rotten Tomatoes, Amazon) and content-aggregation sites (Reddit, Tumblr) have resulted in widely available unstructured text data. Historically strong results were achieved on Natural Language Processing (NLP) tasks using expert knowledge, for instance in the form of a knowledge base [45]. However, as the availability of unstructured text data has risen in recent years, machine-learning models that can process large amounts of text data have achieved state-of-the-art results on a variety of NLP tasks[1]. For example, machine translation [78] tasks, question answering tasks [14], or text classification tasks, that e.g. identify if social media posts or product reviews have a positive or negative sentiment [13], identify social media posts that are useful to crisis responders during crises [13], or predict depression in social media users [1].

A critical question in achieving strong performance on tasks using machine-learning is how to process unstructured data into a representation that can be used by machine-learning models. This thesis in particular deals with how to achieve good representations of unstructured text documents. An example of a domain where unstructured text documents are used is IMDB movie reviews, where a movie review is a "document". One simple and common method to represent a large amount of unstructured text documents is as a "bag-of-words", which is a matrix where rows are documents and columns are words. The value of each word for a document is the word's frequency in that document. This has the advantage of each feature being clear and understandable, however this representation does not include a variety of important information, for example the context of words.

Ideally, a representation of documents is flexible in the information it can represent, and is

[1] https://github.com/sebastianruder/NLP-progress

learned from a large amount of data. To do this, typically a low-dimensional vector space representation is induced. Low-dimensional vector space embeddings of documents represent semantic meaning spatially. For example, in a low-dimensional vector space representation of movies, scary horror movies would be close to each other, and romantic movies would be spatially distant from them. The problem with these vector spaces is that despite the fact that these representations clearly capture meaning in a non-trivial way, their features do not typically have any particular meaning.

In neural representation learning, the concept of learning a disentangled representation was introduced by [7], and the term "disentanglement" has been used in a variety of ways. In this thesis, a disentangled representation is one that has meaningful features. Some examples of disentangled representations are e.g. in the case of images, a representation of handwritten digits where separate features represent the "style" of the digit, and the digit that was written [15]. In the case of text domains, a disentangled representation was learned that has separate features for the sentiment of the text and the content of the text [34], and in a medical text domain a representation was learned with features corresponding to key medical aspects[31]. Disentanglement has many benefits, such as manipulation of a representation such that only the sentiment is changed while leaving the content intact [41], or providing an inductive bias in machine-learning models.

Typically, a disentangled representation is learned using particular neural network architectures and learning methods that, e.g. require features to be independent from each other [31] [54]. The objective of this thesis is to obtain a disentangled representation from low-dimensional vector spaces. To do so, it follows work by Derrac [19] that introduced a variety of methods to obtain semantic relationships embedded spatially in low-dimensional document embeddings. In particular, they introduced a method to obtain semantic features, where for example, a semantic features in a domain of movie reviews are how "Scary", "Comedic", or "Romantic" a movie is.

The process to obtain these semantic features from a learned vector space is as follows: First, linear classifiers (e.g. linear Support Vector Machines) are used with the vector-space as input, and a hyper-plane is found that separates documents where the word occurred from documents where the word did not occur. Second, the orthogonal direction from that hyper-plane is taken, resulting in a direction that starts far from the hyper-plane on the negative side (those that least have the feature) and ends far from the hyper-plane on the positive side (those that most have

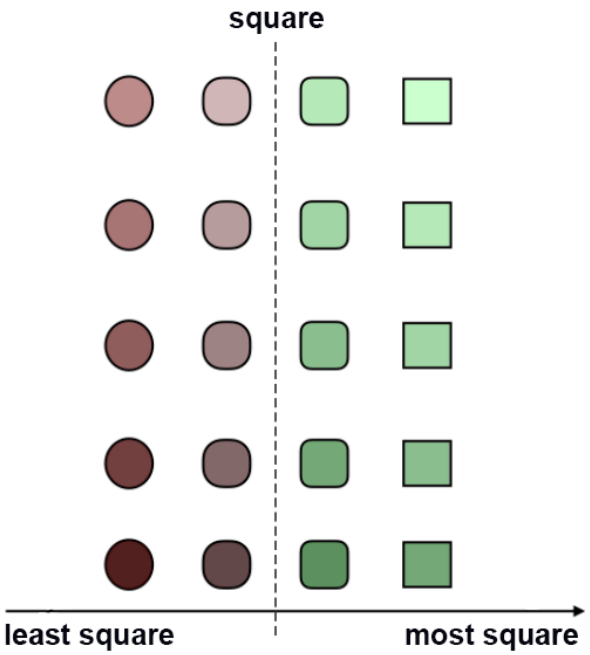

Figure 1.1: An example of a hyper-plane in a toy domain of shapes. The hyper-plane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples. Those closest to the hyper-plane are less square than those further away. The arrow along the bottom of the figure is the direction, and rankings on that direction can be derived from how far along the shapes are on that direction.

the feature). Finally, entities are ranked on that direction using the dot-product and that ranking is used as a feature. An accompanying visual in a two-dimensional toy domain of shapes is shown in Figure 1.1.

Directions in document embedding models that go from documents that least represent a word, to those that most represent it, can be useful in a wide variety of applications. The most immediate example is perhaps that they allow for a natural way to implement critique-based recommendation systems, where users can specify how their desired result should relate to a given set of suggestions [74]. For instance, [75] propose a movie recommendation system in which the user can specify that they want to see suggestions for movies that are "similar to this one, but scarier". If the direction of being scary is adequately modelled in a document embedding model of movies, such critiques can be addressed in a straightforward way. Similarly, in [39] a system was developed that can find "shoes like these but shinier", based on a document embedding

model that was derived from visual features. Semantic search systems can use such directions to interpret queries involving gradual and possibly ill-defined features, such as "*popular* holiday destinations in Europe" [32]. While features such as popularity are typically not encoded in traditional knowledge bases, they can often be represented as document embedding model directions. As another application, directions can also be used in interpretable classifiers, for example, Derrac [19] learned rule based classifiers from ranks induced by the feature directions.

Disentangled representations are linked to the objective of interpretability. As machine-learning has extended into real-world domains like medicine and policing, the legality and risk of implementing systems that do not have easily understood features (e.g. vector spaces) has resulted in concerns of safety [2] (providing the wrong decision in a high-risk domain) and fairness [50] (using properties like the race of a person to e.g. deny a loan). Further, the European Union introduced a legal "Right to explanation" [26], requiring that machine-learning models must be able to explain why they have made a decision about a person. Solutions to explain models after they have been learned have been proposed, as well as transparent machine-learning models [23]. Post-processing a disentangled representation could potentially be effective for use in simple machine-learning models, as each feature is meaningful. However, the work in this thesis is focused on obtaining a disentangled representation, rather than using disentangled representations for explanation or the purposes of transparency.

The method introduced by Derrac [19] forms the basis of the work in this thesis. This Chapter continues as follows: In Section 1.1 the research hypothesis for the thesis is posed. This is followed by an introduction of three research questions in Section 1.2, each corresponding to a relevant chapter in the thesis. The contributions of each chapter is then described in Section 1.3, and the structure of the thesis is laid out in 1.4.

1.1 Hypothesis

Vector space representations of documents can be disentangled in an unsupervised way such that their dimensions correspond to clear semantic features. Furthermore, these disentangled representations provide a useful inductive bias to classifiers, and can be used to investigate the hidden layers of neural networks to gain valuable insights into how they represent documents.

1.2 Research Questions

Question 1: Can directions be identified that correspond to clear semantic features, and can they be used to achieve good disentangled representations across a wide range of document embeddings and domains?

Question 2: To what extent can these directions and associated disentangled representations be used to gain qualitative insights into the characteristics of different neural networks?

Question 3: Is it possible to obtain higher-quality disentangled representations by fine-tuning the initial vector space, while remaining in an unsupervised setting?

1.3 Contributions

In the work by Derrac [19], three domains were investigated and one document embedding model was used. In Chapter 4 the method is quantitatively and qualitatively validated using data from five different domains and a variety of document embeddings. Additionally, variants of the method by Derrac [19] are introduced and shown to increase performance for a majority of domains and embeddings. Disentangled feature representations obtained using these methods are quantitatively validated using low-depth Decision Trees, limited to a depth of one, two or three. In particular, these decision trees classify documents on key domain tasks, e.g. genres of movies. An example of one such Decision Tree, classifying if a movie is a "Sports" movie, is shown below in Figure 1.2. Low-depth decision trees are used as their predictive performance can be used as a proxy for disentanglement, as if a key domain task can be classified using a a one-depth decision tree with a single feature from the disentangled representation as input, this means the method must have disentangled a key semantic feature. For example, when classifying if a movie belongs to the Horror genre using a decision tree that uses only a single feature, high performance of that tree suggests that the semantic feature used directly models if a movie is a Horror movie. Similarly, in a tree limited to a depth of two, features must be particularly relevant e.g. when classifying if a movie belongs to the Horror genre, strong performance of a depth-2 decision tree suggests that the semantic features are used in the tree that are highly predictive (individually or jointly) of this class, e.g. properties like "Scary" or "Bloody".

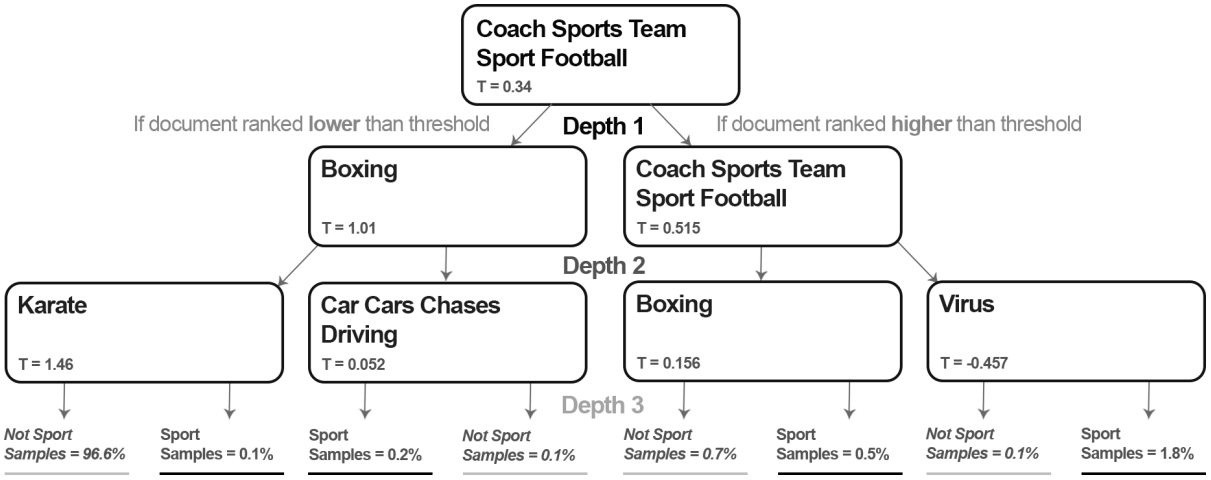

Figure 1.2: An example of a Decision Tree classifying if a movie is in the "Sports" genre, where nodes use semantic features derived using the method in this thesis).

In Chapter 5 the method described in Chapter 4 is used to qualitatively investigate the hidden layers of neural networks, specifically feed-forward networks and auto-encoders. In-order to investigate these networks, the output of the activation values of the hidden layers of the trained models are viewed as vector space representations, and disentangled feature representations are derived from them. Feed-forward networks are trained on supervised data to classify key domain tasks, and the disentangled feature representations derived from these feed-forward networks are quantitatively tested using depth-3 decision trees. The predictive performance of these decision trees is compared to the neural networks, and it is found that not much predictive performance is lost in most cases, and in one case, the disentangled feature representation obtained from the neural network when used as input to low-depth decision trees out-performs the neural network it was learned from. A qualitative investigation of the disentangled features from feed-forward networks and auto-encoders is conducted and they are found to be meaningful. Further, characteristics of the networks are identified, in particular that they represent new semantic features that are more relevant to the class. The auto-encoders are also stacked to obtain a sequence of increasingly low-dimensional vector space embeddings, and increasingly abstract semantic features are derived from them. A method is introduced to induce rules that describe relationships between semantic features from each layer, and the application of this method to building a domain theory is explored. This work was published in NeSy'16, the Eleventh International Workshop on Neural-Symbolic Learning and Reasoning.

In Chapter 6 we identify an issue with the use of the similarity centred objective used to build

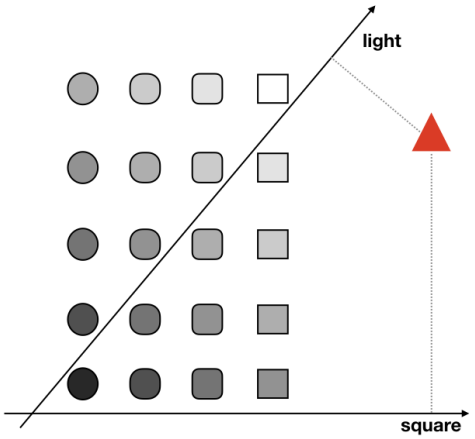

Figure 1.3: Toy example showing the effect of outliers in a two-dimensional embedding of geometric shapes.

the vector space embedding. Specifically, although the vector spaces can be re-organized into meaningful directions, the similarity objective can sometimes be counterproductive. An example of this in a two-dimensional toy domain is shown in Figure 1.3, where the similarity objective results in an outlier. Following this, a method is introduced to fine-tune the vector space, improving the directions in the vector space embedding at the expense of modelling similarity: First, Positive Pointwise Mutual Information (PPMI) scores for the words that label the interpretable features are obtained. Then, a target ranking for each feature is found by using isotonic regression to obtain values inbetween the PPMI scores and the rankings of the entities. This target ranking is used to train a single layer neural network with a non-linear activation function that attempts to match the rankings of entities to the target ranking. The intention is not to achieve 100% accuracy, but instead rearrange the rankings so that similarity based information is de-prioritized if it allows us to learn more meaningful directions. This results in a performance increase in some low-depth decision trees, and a qualitatively investigation shows that the entity rankings become more specific and meaningful for the features. This work was published in The SIGNLL Conference on Computational Natural Language Learning (CoNLL) 2018.

1.4 Thesis Structure

- **Chapter 2** gives an overview of methods for processing unstructured text data, standard machine-learning classifiers, and it provides a background on interpretable representations.
- **Chapter 3** introduces and explains the datasets used in this thesis, as well as giving an introduction to the hyper-parameters used for the machine-learning models in this thesis.
- **Chapter 4** quantitatively and qualitatively investigates how the method introduced by Derrac [19] can produce disentangled representations. In particular, we introduce the use of new scoring functions in this method, specifically NDCG, the use of standard k-means, and comprehensively analyse the relative performance of all variants across different document representations and domains.
- **Chapter 5** qualitatively investigates the use and application of the method in Chapter 4 to neural networks, in particular feed-forward networks and auto-encoders.
- **Chapter 6** introduces a method to improve the interpretable feature representation by prioritizing these features over the similarity information that is captured in the vector space.
- **Chapter 7** provides conclusions on the contributions of this thesis and outlines a number of possible avenues for future work.

1.5 Summary

This thesis experimentally validates and improves on a methodology for obtaining disentangled feature representations for text documents. It is validated using document classification tasks, and disentangled features from neural networks are qualitatively analysed. The method is unsupervised, acts as a post-processing step on vector space representations, and disentangles important semantic relationships in the space.

Chapter 2

Background

2.1 Introduction

In this chapter a general process to make predictions from raw text data is explained. The steps of this process are expanded on in the later sections.

In this thesis we focus on the task of obtaining from raw text data that is effective for the task of document classification. Document classification is the task of distinguishing between documents in a domain, where documents are e.g. movies in a domain of movie reviews, people in a domain of twitter posts, or reviews in a domain of product reviews. First, the text is pre-processed so that noise is removed, "noise" in this case referring to information in the text that is not useful when solving the domains document classification tasks. For example, html tags like [bf], or unnecessary punctuation and grammar.

One popular and simple representation is the bag-of-words. The bag-of-words represents a document as a vector where each element corresponds to a unique word in the corpus. The values of these elements are usually some statistic related to the word's importance in the document e.g. word frequency. One disadvantage of this representation is that it does not retain the context of words. Another disadvantage is that this representation is sparse, as each document vector has an element for every word in the corpus and only some of them will have a frequency above zero. This means that to store it and process it in memory efficiently, specialized data structures and machine-learning techniques must be used. However, bag-of-words representations have the advantage of being easy to understand for humans as each element of the vector representation for each entity corresponds to a word.

Ideally the number of dimensions would be reduced while retaining the information. One

method of doing this would be hand-crafted feature selection, where words which are identified by experts as not meaningful are not included. However, making such a hard selection of which words are relevant and which words are not is challenging and high-risk. An alternative approach is to use the vector similarity between docuemnts, e.g. the similarity between their frequency BOW vectors, to produce a low-dimensional vector space, such that the similarity between the vectors in the low-dimensional space approximates the similarity between the corresponding BOW-vectors. However, this results in vectors whose elements, or "features" are no longer interpretable in the sense the bag-of-words is.

As mentioned in Section 1.3 the main focus of this thesis is in disentangling the information in vector spaces into semantic features. This chapter introduces the process of obtaining bag-of-words and vector space representations from text data and using them to solve machine-learning problems, as well as giving an introduction to related work in interpretability of machine learning representations and models. To outline the process, first as covered in Section 2.2.2 the data is preprocessed so that unnecessary information is removed. Then some basic representations are obtained in Section 2.3.1, followed by more complex vector space representations in Section 2.5. To complete a standard machine-learning pipeline, Section 2.4 covers different machine-learning methods. Finally, interpretable representations and classifiers are covered in Section 2.6 to provide context to the work presented in this thesis.

2.2 Text Data

This thesis is focused on producing disentangled representations from text data. In this section, the basics of what the text data is, terminology associated with it and how it is preprocessed is described.

2.2.1 Text Domains

"Text domain" refers to a subject area that is unique in its vocabulary and structure. One example is the newsgroups domain (See Section 3.2), which is composed of online news discussion groups. Below an example post is provided from the newsgroups domain that contains unique

jargon like "NOEMS", "EMM386" and a unique structure e.g. signing the post with the persons name and a personal tagline for contacting them.

> Has anyone else experienced problems with windows hanging after the installation of DOS 6? I have narrowed the problem down to EMM386.
>
> If if remove (or disable) EMM386, windows is ok. If EMM386 is active, with NOEMS, windows hangs. If I use AUTO with EMM386, the system hangs on bootup.
>
> Dave.
>
> –
>
> ———————————————————————-
>
> David Clarke ...the well is deep...wish me well...
>
> ac151@Freenet.carleton.ca David_Clarke@mtsa.ubc.ca clarkec@sfu.ca

These particularities to the domain are what makes the distinction between domain-specific text data to general text data. A machine-learning model will develop a representation of how to solve the task dependent on this data. If the model is given general text data examples to learn from, then it will miss out on domain-specific quirks that can help it solve the task e.g. when learning to identify if a newsgroups post belongs to the subject of windows. If the general examples do not use jargon like "AUTO" and "EMM386" then this important information will not be used. However, if the examples given are over-specialized then the model may place excess importance on domain-specific quirks that are not actually meaningful, e.g. if in the training examples of posts about windows most users signed off their text-posts with an email that includes ".ca", meaning they are from canada, then the model may identify all posts that include ".ca" emails as about windows despite this simply being a strange quirk in the data.

2.2.2 Pre-processing Text Data

The datasets used in this thesis are composed of documents. In some of these datasets, these documents correspond to entities. An entity representation is different to a document representation in that each document corresponds to an entity in the domain. For example, a document

dataset of IMDB movie reviews has a document for each movie review. However, an entity dataset would have documents that correspond to each *movie*, where those documents are a concatenation of movie reviews.

Being able to automatically remove this noise is an essential step of building a representation and solving machine-learning problems. The first stage of obtaining a bag-of-words is building a vocabulary W_w, composed of unique words $w \in W$ from the corpus. In this vocabulary, it is important that words are identified e.g. if the word "Dog" was considered to be different to the word "dog." then the vocabulary would be too noisy. There are standard methods for removing noise in a dataset. We describe them in the following bullet-points:

- Convert all text to lower-case, e.g. "The man and The Dog" converted to "the man and the dog"
- Remove all punctuation including excess white-space, e.g. "the man, and the dog..." converted to "the man and the dog"
- Using a predefined list of "stop-words", remove words that are not useful, e.g. "the man and the dog" converted to "man dog"
- Remove infrequent words, e.g. "man dog, dgo, dog man" converted to "man dog, dog man".
- Domain-specific pre-processing to remove metadata, e.g. removing emails from the end of movie reviews.

In this work and the representations used in this work, the rules above are applied to the corpus beforehand. In terms of removing infrequent words, words that did not occur more than once are removed. In the next section, we cover some methods for text representation and explain their basic advantages.

2.3 Representations

Machines cannot parse the meaning of text like news articles, product reviews or social media posts without a representation of the meaning in a computational structure e.g. a matrix. We

will consider representations in which documents are represented as fixed-dimensional vectors $r = (x_1, x_2, ..., x_n)$. Here, the components of this vector can be thought of as features, and ideally each of these features x are meaningful in the domain. For example, meaningful features when determining the value of a house would be the number of bedrooms x_1, and the number of toilets x_2. An example vector from these examples would be $(6, 3)$ for a house with 6 bedrooms and 3 toilets.

2.3.1 Bag-Of-Words

This section is about a simple representation of text data called a bag-of-words. The bag-of-words can scale to an extreme amount of data, that comes with the following assumption: the context of words is unnecessary information to perform well on the task. How correct this assumption is depends on the task, but despite this view being overly-simplistic the application and use of the bag-of-words (BOW) is broad. There are multiple ways to represent words in the BOW format, but the most common is by the frequency of the words in a document.

The bag-of-words (BOW) assigns a value to each word based on its number of occurrences in the document. For example, a short document like "there was a dog, and a man, and the man, and the dog" would be translated into word frequencies "(there: 1, was: 1, a: 2, and: 3, the: 2, man: 2, dog: 2)". This representation is simple, but ignores word context, grammar and punctuation.

The structure of the bag-of-words is a matrix, where rows are documents and columns are words in the corpus vocabulary. Specifically, text documents in a domain $d \in D$ have an associated vocabulary of unique words across all documents $w \in W$. The bag-of-words B_D is a matrix where each document is a row, and each column is a word, where the value of each word for a document is the word's frequency in that document $d = (wf_1, wf_2, ..., wf_n)$ where $wf(d)$ is equal to the frequency of a word in a document and n is equal to the number of unique words in the vocabulary for all documents $w \in W$. In terms of the general structure given above, our representation r is the bag-of-words, and the features $r = (x_1, x_2, ..., x_n)$ are the word frequencies.

Term Frequency Inverse Document Frequency (TF-IDF)

When representing documents using frequency, longer documents have overall higher values than shorter ones. Ideally, documents that have a similar meaning but at a smaller scale are treated equally. To do this, documents need to be normalized relative to each other. Further, frequency gives importance to terms that are frequent across all documents, but these are not useful for distinguishing between documents. For example, in a domain of movie reviews, the word "movie" despite being frequent in a majority of documents is not useful. Instead, it would be better that terms that are not useful for distinguishing between documents were not given a high value, and documents that are unique to a smaller number of documents was given a high value. For example, if the term "gore" was frequent in only five different movies out of 15,000 then it is clearly important for those movies.

The idea that words which are infrequent overall but frequent for some documents are important can be applied to a bag-of-words using the Term Frequency Inverse Document Frequency (TF-IDF) formula. The first part of TF-IDF is Term Frequency TF_d, w, which is a normalization of frequency that solves the first problem of larger documents being treated as more important than shorter ones.

$$TF_{(w,d)} = \frac{wf(d)}{\sum_n wf_n(d)}$$

where $wf(d)$ is the number of occurrences of word w in document d and n is the number of words overall in the vocabulary. The next part of TF-IDF is Inverse Document Frequency, which is a measure that rewards terms that have a low Document Frequency.

$$IDF_w = \frac{d_n}{df(w)}$$

Where $df(w)$ is the amount of documents the word w has occurred in and d_n is the amount of documents in the corpus. Note that while Term Frequency measures the frequency of a term in a document relative to that documents length, Document Frequency measures the overall occurrences of the term across all documents. Essentially, it measures how rare that term is for

a document. Finally, the TF-IDF is just the Term Frequency multiplied by the Inverse Document Freqency.

$$\textit{TF-IDF} = TF \times IDF$$

Positive Pointwise Mutual Information (PPMI)

Pointwise Mutual Information (PMI) comes from information theory, and is a metric that measures how *dependent* two variables are i.e. what are the chances of the variables occurring at the same time relative to the chance of them occurring independently. In this case, it can be used to measure how dependent a word is on a document. Obviously it is not possible to determine a precise probability that a word will occur, so in practice the frequency of the word is used to derive an approximation of the chance it will occur. In applications, we can understand that the word "the" is not independent from the document - it is a word that is just as likely to occur in one document than another because its occurrence is not dependent on the document. However, in a domain of movie reviews a word like "thrilling" would be more dependent on its associated text document, as it would tend to occur for movies which are thrilling. The PMI value for a word w in a document d is given by:

$$\textit{pmi}(w, d) = \log\left(\frac{p_{w(d)}}{p_w \cdot p_d}\right)$$

where $P_{w(d)}$ is equal to the chance of the word occurring in the document assuming they are dependent on each other

$$P_{w(d)} = \frac{w(d)}{\sum_w \sum_d w(d)}$$

and $w(d)$ is the frequency of a word for a document. To calculate the chance that a word will occur, we simply take the chance the word will occur in any document (estimated by its summed frequency) over all frequencies, and for the document we take the chance that the

document will occur (represented by the sum of the frequencies of all words that occur in it) over all frequencies:

$$P_w = \frac{\sum_d w(d)}{\sum_w \sum_d w(d)} \qquad P_d = \frac{\sum_w w(d)}{\sum_w \sum_d w(d)}$$

As this value can sometimes be negative when words are less correlated than expected, we use Positive Pointwise Mutual Information (PPMI), as we are only interested in words which are positively correlated.

$$ppmi_{w(d)} = \max(0, pmi)$$

The PPMI BOW is the representation used often in this thesis for a simple representation of meaning in the domain. It forms the basis of more complex representations and is also sufficient as a simple interpretable representation.

2.4 Text Document Classification

Machine learning algorithms can be split into two distinct categories, supervised and unsupervised. Supervised problems have some data that is labelled, and some that is not labelled. The goal of a supervised task is to assign labels to the data that is not labelled, by learning with the data that is labelled. For example a twitter post can be classified as positive or negative. Unsupervised problems do not have any labelled training examples, and instead try to solve a problem just from unlabelled data. An example of an unsupervised problem is clustering, the task of finding e.g. similar documents and grouping them together.

A classification problem has labels (or "classes") where each example either has a label or does not have. Labels can be understood as categories in the domain, e.g. in the domain of sentiment analysis on movie reviews, labels could be "very good", "good", "average", "bad", "very bad". Given a set of possible labels, documents D and document/label pairs are assigned a binary truth value $(d, c) = 0, 1$. From these values, a classifier finds a function that assigns unlabelled documents $d \in D$ to predicted labels (d, c_p). This function approximates an unknown target

function that can accurately label any document. For example, in a domain of movie reviews, each review is labelled as either positive or negative, and a function must be found that can determine if unlabelled movie reviews are positive or negative.

2.4.1 Types of Classification Problems

There are three kinds of classification problems that are used in this thesis. The first are binary problems. This is where there are two possible classes, and they are mutually exclusive. An example of this is a task that classifies if a movie review is positive or negative. The second kind of classification problem is multi-class. A multi-class problem has more than two mutually exclusive classes, where each example belongs to only one class e.g. classifying the age-rating of a movie. The final kind of classification problem is multi-label. In a multi-label classification problem examples can belong to multiple classes at once. For example, classifying if a movie contains particular themes like "blood", "romance", or "relationships". A movie can contain both "blood" and "romance", but some movies only contain "blood" and some movies only contain "romance".

2.4.2 Decision Trees

Decision Tree learners represent classification models using trees, whose internal nodes are associated with features and a threshold value T. In the case where a bag-of-words representation is used as input, the nodes of this Decision Tree will correspond to words in the corpus vocabulary. When a decision tree classifies an example, the way the tree is traversed is determined by if the value given by the example is larger than the threshold T. Leaf nodes determine if the classification decision is positive or negative. Eventually the traversal reaches the bottom of the tree, called a leaf node, and the final decision made on the threshold of the leaf node is the classification of the document.

The tree can be viewed as a hierarchy of importance for the class, with the most important features for classification at the top and the less-important ones below. Decision Trees have nodes that correspond to features, so if these features are simple and easy to understand then the tree is also interpretable [73].

2.4.3 Linear Support Vector Machines

Where documents are viewed as points in a vector space, where the dimensions of that space are the features, a linear support vector machine finds a hyper-plane that maximizes the margin between entities belonging to different classes. To classify new entities, they are placed in this space and labelled according to which side of the line they fall on. A parameter can be tuned for this classifier, the C parameter. The C parameter determines how much misclassification is penalized, in other words, how much bias there is in the model. A large C value results in low bias, and high variance, as misclassifications are heavily penalized. A low C value results in high bias and low variance, as misclassifications are not highly penalized.

2.4.4 Neural Networks

Neural networks are composed of layers, and each layers is composed of nodes. Nodes are connected by weights, which have an associated value the output of nodes are multiplied by. Each node has an activation function, which is typically the same for every node in a layer. This function is a transformation of the value input to the node. Some example activation functions are *tanh*, *sigmoid*, and *relu*.

$$\textit{tanh}(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)} \quad \textit{sigmoid}(x) = \frac{1}{1 + exp(-x)} \quad \textit{relu}(x) = \textit{max}(0, x)$$

Each node also has an activation threshold, which the value it receives must reach in-order to be output along the weights of the network. Generally, a neural network is composed of an input layer, which has the same amount of nodes as the number of input features. Then, there are hidden layers, which may vary in dimensionality, and finally an output layer. Each time an example is processed by the network, information flows through the nodes and along the connections to the output layer. Each example has some desired output, e.g. in a binary classification task, a single output node would be used and the correct class assignment for that example is the desired output. From this output layer, a loss is calculated. One example loss function is the mean squared error, which calculates the average of the squared differences between what is predicted and what the desired value is for the prediction. Then, the gradient of

the loss function is calculated with respect to each weight, and weights are updated to minimize loss. Typically, the process for calculating the gradient is estimated, and all the weights are updated using that estimation. Once the network has been trained, examples are input to the final network and a simple threshold on the output layer is applied to determine if an example belongs to the class or not (e.g. probability > 0.5).

One kind of neural network is the feed-forward network, where nodes only connect to nodes in subsequent layers. In this way, the feed-forward network always feeds information forwards. In standard applications of a feed-forward network, layers are "fully connected", meaning that each node is connected to every node in the subsequent layer.

Hidden layers of neural networks can be viewed as vector spaces, where the result of learning is that the position of entities in that hidden layer are better spatially organized for solving the given task. For example, entities that belong to one class may be spatially grouped together. This is an important advantage of neural network models in the context of text classification: these models jointly solve the task of representation learning (i.e. transforming some initial representation into one which is better suited for the given task) and the task of learning the actual classification model.

2.4.5 Overfitting

If a machine-learning model is given training data, then the model could simply learn a function that memorises the training data. Ideally the model instead captures meaning in the domain that enables it to generalize well to examples it has not learned from. This is known as the bias/variance trade-off. High variance models prioritize the correct classification of existing examples, e.g. in the case of a non-linear function that ensures each example in the training data is classified correctly by increasing complexity. This is at the cost of bias, or the amount of assumptions made by the model. A high bias model would make many assumptions about the target function, e.g. using a linear function, and because of this the function may generalize well to new examples if those assumptions are correct.

In the case of a neural network, this behaviour can be stopped by limiting the number of neurons available in the hidden layer, forcing the network to generalize the representation into a lower-dimensional vector space. However, the problem of overfitting to the examples given rather than

learning a way to solve the problem in a general way is a persistent one in machine-learning tasks. To give an example, when learning with a bag-of-words the model may realize that each document was written by a different user, and that users name is recorded in the document text. A simple function would be to say:

IF user_name_1 is > 0, THEN class = 1.

However, this is not actually learning any domain knowledge, it is simply overfitting to noise.

To better select hyperparameters that avoid overfitting, the data for a supervised problem is usually split into three parts:

Training data The training data are the examples that the model learns from. It is used only when creating the model, and is not used after the model has finished learning.

Test data The examples that the model uses to check if the function learned is correct.

Validation data A decision tree may perform better if it is shallow and limited in depth rather than unlimited in depth, as it will not introduce nodes that are overly specific to the training data. Validation data is used for parameter tuning, e.g. when determining how much to limit the depth of a decision tree, ho good the parameter is would be evaluated on how well the model performs on the validation set. The separation of validation data from test data is just to ensure that we are not overfitting the parameters on specific examples.

2.4.6 Evaluation Metrics

To evaluate a model, the difference between the real labels of documents and the predicted features of documents are compared. However, the value of the model is in its ability to predict the labels of documents that are unlabelled.

Here, we assume we are classifying a single binary class, where positive labels are denoted by 1 and negative labels by 0. The simplest way to evaluate a model is by its accuracy a, where C_n is the number of correct predictions, and P_n is the number of all predictions.

$$a = \frac{C_n}{P_n}$$

However, this can give a misleadingly high score if for example, the dataset is unbalanced with many more negative labels than positive ones, and the model predicts only negatives. An

example of where this would be the case is when classifying out of all social media posts, which ones are important for emergency responders to investigate. Although there are very few positive instances of this class, identifying those is very important. In the case of a model predicting only negatives, the accuracy would be high as the number of correctly predicted negatives *tn* is high, but the model has not actually learned anything, which we can tell by looking at the number of correctly predicted positives *tp*. For a metric that can take this into account, we must consider the number of incorrectly predicted positives (negatives classified as positive) *fp* and the number of incorrectly predicted negatives *fn*.

In this situation, recall is an informative measures. Recall *rec* is the proportion of true positives *tp* identified correctly.

$$rec = \frac{tp}{tp + fn}$$

In the case of a model predicting only negatives, the *rec* would be zero. Recall is useful in these situations where we are interested in how many false negatives fn there are. However, if the model is instead prioritizing positive predictions too much rather than negative ones, we can use precision pre

$$pre = \frac{tp}{tp + fp}$$

F1 score is the harmonic mean of recall and precision, it is used to balance and measure the recall and precision at the same time where they are equally important.

$$F1 = 2 \cdot \frac{pre \cdot rec}{pre + rec}$$

2.5 Vector Spaces

The bag-of-words (BOW) based on frequency statistics has the benefit of being easy to understand on a granular level, as each feature is a distinctly labelled word. However, it is difficult to deal with words in the test data that haven't been seen during training, you can't reliably learn the importance of words that have few examples. Ideally, the information in a bag-of-words could be represented in a lower number of dimensions while preserving the most relevant information as much as possible.

Low-dimensional vector spaces are generally learned by taking semantic information in a sparse e.g. BOW representation, and encoding it such that documents that are semantically similar are close together. However, these dense vector space representations usually no longer have features (i.e. dimensions) which are meaningful to humans. This is a trade-off when going from a sparse representation to a dense representation, the features are no longer meaningful.

This can lead to unexpected disadvantages when classifying text with a simple classifier, e.g. a low-depth decision tree. In a bag-of-words, terms that are particularly important for classifying the task could be selected as important features at the top of the tree. However, in a low-dimensional vector space the information that is suitable for classification is not sufficiently separated into a distinct feature; rather it is encoded in the spatial relationships of the vector space.

The main focus of this thesis is in how to disentangle the semantic information encoded spatially in a vector space into semantic features. This is essentially producing a new representation that captures the same information as the initial vector space, but instead has features that are semantically meaningful similar to how a bag-of-words has individual features for each word. However, the features are not words but instead semantic properties in the space. In this case, "properties" refers to aspects of documents in the domain, for example, in a domain of movie reviews the property of "comedy" describes how comedic movies are.

2.5.1 Principal Component Analysis

Principal Component Analysis (PCA) is a linear dimensionality reduction method. Given a set of objects described by feature vectors e.g. a bag-of-words, it produces a vector space of a specified size n, where dimensions are ordered by semantic importance.

Essentially, PCA works by linearly combining features in-order to create new features that can differentiate entities well and are uncorrelated with previous features. This results in a new low-dimensional representation that retains information and has distinct semantic features. However, as these features are a linear combination of the previous features, they are generally not interpretable [24].

2.5.2 Multi-Dimensional Scaling

Multi-Dimensional Scaling (MDS) is a dimensionality reduction algorithm. In this work, non-metric MDS is used. In the same way as PCA, the size of the output space is specified. As input, MDS takes a dissimilarity matrix of entities, where both rows and columns are entities and the values are the dissimilarity between those entities. To calculate the dissimilarity between two entities e_i and e_j the normalized angular difference is used.

$$ang(e_i, e_j) = \frac{2}{\pi} \cdot arccos(\frac{v_{ei} \cdot v_{ej}}{||v_{ei}|| \cdot ||v_{ej}||}) \tag{2.1}$$

From a bag-of-words, the way to construct this dissimilarity matrix is by finding the dissimilarity between bag-of-words features for each entity. A disadvantage of this method is that the dissimilarity matrix grows quadratically in the number of entities, which means that it may not fit in memory for larger datasets. The end-result of MDS is a representation where entities that are semantically similar according to the input matrix are spatially close to each other, and semantically different entities are spatially distant from each other.

2.5.3 Word Embeddings

Word embeddings are a vector space representation for words. They are typically learned using a large corpus of unstructured text, e.g. Wikipedia. There are many ways to obtain word-vectors, one traditional approach is matrix factorization [12]. Recently modern methods like GloVe [57] and Word2Vec [52] have been widely adopted. These methods learn representations of words using the context of their surrounding words. Essentially, the meaning of each word is determined only by context. These representations have been extremely useful, and have semantic coherence, as shown by being able to model relations between words, e.g. analogical relations represented using vector operations, where vec(word) is the word vector for a word, vec(King) - vec(Man) $\approx$ vec(Queen).

2.5.4 Doc2Vec

Doc2Vec [42] extends the neural network method of learning word vectors introduced by Word2Vec [52] and extends it such that a document representation is learned in tandem. Essentially, as well as learning from the word's context, the words are also learned according to what documents they are in. The document representation is built in the same way as the word representation, gradually being informed by the word context and document context.

2.6 Interpretable Representations

2.6.1 Conceptual Spaces

The inspiration of the work by Derrac [19] was conceptual spaces. Within the field of cognitive science, feature representations and semantic spaces both have a long tradition as alternative, and often competing representations of semantic relatedness [72]. Conceptual spaces [22] to some extent unify these two opposing views, by representing objects as points in vector spaces, one for each facet (e.g. color, shape, taste in a conceptual space of fruit), such that the dimensions of each of these vector spaces correspond to primitive features.

The main appeal of conceptual spaces stems from the fact that they allow a wide range of cognitive and linguistic phenomena to be modelled in an elegant way. The idea of learning semantic spaces with accurate feature directions can be seen as a first step towards methods for learning conceptual space representations from data, and thus towards the use of more cognitively plausible representations of meaning in computer science. Our method also somewhat relates to the debates in cognitive science on the relationship between similarity and rule based processes [27], in the sense that it allows us to explicitly link similarity based categorization methods (e.g. an SVM classifier trained on semantic space representations) with rule based categorization methods (e.g. the decision trees that we will learn from the feature directions).

Fundamentally, both of these views seek to find the essential components that determine why all entities vary in the domain, and use them as features. In the case of text processing which we investigate in this work, the factors of variation found correspond to clusters of words that

represent properties in the domain. The representation is considered disentangled if the features obtained are interpretable and predictive when used in key domain tasks.

2.6.2 Topic Models

The method in this thesis produces a disentangled feature representation where each feature is semantically coherent. This is somewhat similar to Topic models like Latent Dirichlet Allocation (LDA), which learns a representation of text documents as a multinomial distributions over latent topics, where each of these topics corresponds to a multinomial distribution over words [10]. Topics tend to correspond to salient features, and are typically labelled with the most probable words according to the corresponding distribution.

Compared to topic models, vector space models have the advantage that they are versatile in how they can be learned, enabling e.g. structured knowledge from the domain, or different kinds of data like images to be taken into account. Some authors have also proposed hybrid models, which combine topic models and vector space models. For example, the Gaussian LDA model represents topics as multivariate Gaussian distributions over a word embedding [17]. Beyond document representation, topic models have also been used to improve word embedding models, by learning a different vector for each topic-word combination [46].

Compared to topic models, our work leverages clustering and similarity methods to obtain feature labels, and is a post-processing step that disentangles vector spaces. This gives the method in this work the advantage of broad applicability, especially to vector spaces that are learned in a variety of different ways e.g. using neural networks. LDA has also been extended, for example to incorporate additional information, e.g. aiming to avoid the need to manually specify the number of topics [69], modelling correlations between topics [9], or by incorporating meta-data such as authors or time stamps [61, 77]. Nonetheless, such techniques for extending LDA offer less flexibility than neural network models, e.g. for exploiting numerical attributes or visual features. For comparison, in our experiments the standard topic model algorithm Latent Dirchlet Allocation (LDA) is used as a baseline to compare to the new methodology that transforms standard Vector Space Model representations.

2.6.3 Generative Adversarial Network and Variational Autoencoders

Generative Adversarial Network (GAN) [25] are neural networks that learn representations using a discriminator and a generator, where the generator encodes a probabilistic model from which is aimed at generating entities that are similar to those from a given training set. This generator network is simultaneously trained with a discriminator network, which aims to predict whether a given entity is an actual example from the training set or was sampled from the generator. The generator network implicitly learns a latent space of the entities from the considered domain, which has been proven useful in a wide variety of tasks, despite generally not being interpretable. However, GAN's have been extended to produce an interpretable disentangled latent space, in particular InfoGan has shown that it can obtain interpretable features in the latent space where each feature corresponds to a salient factor, e.g. in a task of identifying what digit is written in an image of a handwritten digit, there are features for each digit and an additional digit used for the style of writing [15]. GAN's have also been applied in text [11, 37] with some success, despite being noted as 'particularly difficult to train' in the text domain [4] even with advancements in this direction [51]. The work in this thesis differs from the disentangled representations found in GAN's as it focuses on disentangling text document representations into semantic features that are relevant to text classification, and has a broad applicability to document-based text representations.

2.6.4 Sparse Representations

Methods to obtain sparse and interpretable word vectors have been developed by either adapting a learning method to include sparsity constraints e.g. non-negative sparse embeddings adapting matrix factorization with sparsity constraints [**?**] or [47] adapting neural networks. Alternatively, some work follows a similar line to ours in that they post-process existing dense embeddings [67] [56] [21]. In the former category, this approach has also been extended to sentences [70], and follows the idea that PCA and other dense representations are effective at compressing information into a small number of dimensions, although this results in semantically incoherent features. Instead, a larger representation with similar performance but more dimensions and high semantic coherency of its features is learned. In this way, information that was compressed into a small amount of dimensions previously has been disentangled into a larger number of fea-

tures. However, this can sometimes come with a minor loss of performance, particularly when using a lower number of dimensions. The features of these representations are labelled using the top n highest-scoring words on the feature. Sparse interpretable representations have also been derived from sentences [71].

There are also document representations that use sparsity constraints to obtain interpretable sparse representations like sparse PCA learned using the l1-norm, [30] [79] or Sparse MDS [66]. Compared to sparse representations, the methods in this thesis also attempt to post-process a dense representation in order to disentangle them, but it does not aim to produce a sparse representation that may perform poorly with a small number of features. Instead, the objective of the representation obtained in this thesis is to perform well with a small number of features, under the assumption that if we are able to identify key properties in the domain as features then we should only need a small number of features to perform well at key domain tasks like text classification.

One method that does not produce a sparse representation but still learns an interpretable representation is [38], where features correspond to concepts. In their work, an external lexical resource is used to define concepts that will correspond to features in the representation before training. This differs from our work in that we do not use any external resources apart from a bag-of-words from the domain to determine the features, rather they are determined by what the vector space representation itself prioritizes, as the features are derived directly from the semantic relationships that are spatially encoded in the representation.

2.7 Conclusions

Bag-of-words (BOW) representations are simple and meaningful, achieving strong results despite not being complex. BOW representations are also interpretable in principle, but because the considered vocabularies typically contain tens (or hundreds) of thousands of words, the resulting learned models are nonetheless difficult to inspect and understand. Further, the sparsity and size of this representation limits its applications. Topic models and low-dimensional vector space embeddings are two alternative approaches for generating low-dimensional document representations, with the usual advantage of topic models over vector-space models being that

their features are interpretable, as the features are labelled with a group of words. However, vector space models are used on a larger variety of tasks as they are very versatile.

In this thesis a disentangled representation is obtained from a low-dimensional vector space, where each feature is semantically coherent. Some variations of Generative Adversarial Networks can achieve a disentangled representation, but they are difficult to train on text data. Methods to obtain sparse interpretable representations in word-vectors are similar to this work in that they post-process a dense representation, but these methods are limited to word vectors and suffer in performance with low-dimensionality, which we identify as a desirable property of our representation. To the author's knowledge, there is no existing work on obtaining an interpretable document representation from dense vectors that does not utilize sparsity constraints.

This thesis continues as follows: given the background in this chapter, the datasets that will be used in text classification tasks and to produce the dense and interpretable representations are introduced. Then, the method to re-organize dense vector spaces into interpretable representations is deeply experimented on and quantitatively and qualitatively validated. Following this, the dense vector space representations of neural networks are investigated, with the intention to better understand these models with unexpected results. Finally, a method to improve both the semantic coherence and performance of these interpretable representations is introduced and quantitatively and qualitatively validated.

Chapter 3

Datasets and Semantic Spaces

3.1 Introduction

For the experiments in this thesis, five different domains are used, each with their own particular vocabulary and meaning of words in their vocabulary. This Chapter begins with a section to give insight into the datasets with explanations of each domain, accompanying examples, and their classes. This is followed by technical descriptions of preprocessing methods for the datasets. Finally, we introduce the bag-of-words and semantic space representations built from these preprocessed datasets that will be used in the remainder of the thesis.

3.2 Datasets

First, we go through the history and class names of the datasets to give context, and provide examples of unprocessed text from three domains in Table 3.1.

IMDB Sentiment Where documents are exclusively highly polar IMDB movie reviews, either rated <= 4 out of 10 or >= 7 out of 10. Reviews were collected such that it was limited to include at most 30 reviews from any movie in the collection, as some movies contained many more reviews than others. The corpus is split half and half between positive and negative reviews, with the task being to identify the sentiment of the review.

20 Newsgroups[1] Originating from online news discussion groups from 1995 called newsgroups, where group email-type discussions are made by users about particular topics within 20 different groups. In this dataset, each document is composed of a topic, where user posts are concatenated

[1] http://qwone.com/ jason/20Newsgroups/

Data Type	Unprocessed	Processed
Newsgroups	morgan and guzman will have era's 1 run higher than last year, and the cubs will be idiots and not pitch harkey as much as hibbard. castillo won't be good (i think he's a stud pitcher)	morgan guzman eras run higher last year cubs idiots pitch harkey much hibbard castillo wont good think hes stud pitcher
Sentiment	All the world's a stage and its people actors in it–or something like that. Who the hell said that theatre stopped at the orchestra pit–or even at the theatre door? Why is not the audience participants in the theatrical experience, including the story itself? This film was a grand experiment that said: "Hey! the story is you and it needs more than your attention, it needs your active participation"". ""Sometimes we bring the story to you, sometimes you have to go to the story."" Alas no one listened, but that does not mean it should not have been said."	worlds stage people actors something like hell said theatre stopped orchestra pit even theatre door audience participants theatrical experience including story film grand experiment said hey story needs attention needs active participation sometimes bring story sometimes go story alas one listened mean said
Reuters	U.K. MONEY MARKET SHORTAGE FORECAST REVISED DOWN The Bank of England said it had revised its forecast of the shortage in the money market down to 450 mln stg before taking account of its morning operations. At noon the bank had estimated the shortfall at 500 mln stg.	uk money market shortage forecast revised bank england said revised forecast shortage money market 450 mln stg taking account morning operations noon bank estimated shortfall 500 mln stg

Table 3.1: Text examples from three domains. For the movies and place-type domains, the original text was not available..

together. The groups that topics are categorized by are Atheism, Computer Graphics, Microsoft Windows, IBM PC Hardware, Mac Hardware, X-Window (GUI Software), Automobiles, Motorcycles, Baseball, Hockey, Cryptography, Electronics, Medicine, Space, Christianity, Guns, The Middle East, General Politics and General Religion, which also act as the classes for this dataset when being evaluated. Generally, it can be quite easy to identify if a document belongs to a particular group if it uses a keyword unique to that group, e.g. the word "chastity" will almost always mean that the document belongs to the "Christianity" class.

Reuters-21578, Distribution 1.0 Text from the Reuters financial news service in 1987, composed of a headline and body text. The classes were chosen with assistance from personnel at reuters[2], meaning that they can contain jargon. For that reason, explanations are provided with the original names in brackets. The classes are Trade, Grain, Natural Gas (nat-gas), Crude Oil (crude), Sugar, Corn, Vegetable Oil (veg-oil), Ship, Coffee, Wheat, Gold, Acquisitions (acq), Interest, Money/Foreign Exchange (money-fx), Soybean, Oilseed, Earnings and Earnings Forecasts (earn), BOP, Gross National Product (gnp), Dollar (dlr) and Money-Supply.

[2]For more detail on the history of the dataset: https://archive.ics.uci.edu/ml/datasets/reuters-21578+text+categorization+collection

Placetypes Taken from work by Derrac [19]. Originating from the photo-sharing website flickr, where photos are tagged (i.e. words describing the photos like "sepia" or "mountain") by users. 22,816,139 photos were considered, and tags that occurred in place-type taxonomies (Geonames, a taxonomy of man-made and natural features, Foursquare a mostly flat taxonomy of urban man-made places like bars and shops, and the site category for the common-sense knowledge base taxonomy OpenCYC) with more than 1,000 occurrences were chosen as documents. Each document, named after a flickr tag, is composed of all flickr tags where that tag occurred. There are three tasks, generated from the three different place type taxonomies. The Foursquare taxonomy, classifying the 9 top-level categories from Foursquare in September 2013, Arts and Entertainment, College and University, Food, Professional and Other Places, Nightlife Spot, Parks And Outdoors, Shops and Service, Travel and Transport and Residence. the GeoNames taxonomy limited to 7 classes, Stream/Lake, Parks/Area, Road/Railroad, Spot/Building/Farm, Mountain/Hill/Rock, Undersea, and Forest/Heath, and the OpenCYC Taxonomy, which we limited to 25 classes, Aqueduct, Border, Building, Dam, Facility, Foreground, Historical Site, Holy Site, Landmark, Medical Facility, Medical School, Military Place, Monsoon Forest, National Monument, Outdoor Location, Rock Formation, and Room. Naturally as these tasks were derived from taxonomies they are multi-label.

Movies Taken from work by Derrac [19]. The top 50,000 most voted-on movies were chosen for this dataset initially, and reviews were collected from four different sources (Rotten Tomatoes, IMDB, SNAP project's Amazon Reviews [3] and the IMDB sentiment dataset. Then, the top 15,000 movies with the highest number of words were chosen as documents, where each document is composed of all of that movies reviews concatenated together. Three tasks are used to evaluate this dataset: 23 movie genres, specifically Action, Adventure, Animation, Biography, Comedy, Crime, Documentary, Drama, Family, Fantasy, Film-Noir, History, Horror, Music, Musical, Mystery, Romance, Sci-Fi, Short, Sport, Thriller, War, Western. 100 of the most common IMDB plot keywords and Age Ratings from the UK and US, USA-G, UK-12-12A, UK-15, UK-18, UK-PG, USA-PG-PG13, USA-R.

[3] https:/ /snap.stanford.edu/data/web-Amazon.html.

3.3 Technical Details

In this section, we describe the vocabulary and document sizes for each domain. Each domain is preprocessed such that it is converted to lower-case, non-alphanumeric characters are removed and whitespace is stripped such that words are separated by a single space. Words are removed from a standard list of English stop-words from the NLTK library [8] and we filter out terms that do not occur in at least two documents, with an additional limit to the maximum number of words in a vocabulary set to 100,000.

For each task in each domain, each of the datasets are split into a 2/3 training data, 1/3 test data split. We additionally remove 20% of the training data and use that as development data for our hyper-parameters.

IMDB Sentiment[4] When the original corpus was produced, the 50 most frequent terms were removed. It contains 50,000 documents with a vocabulary size of 78,588. After removing terms that did not occur in at least two documents, the vocabulary size was reduced to 55384. the number of positive instances in the classes is 25,000.

20 Newsgroups[5] Obtained from scikit-learn. [6] Originally containing 18,846 documents, in this work it is preprocessed using sklearn to remove headers, footers and quotes. Then, empty and duplicate documents are removed, resulting in 18302 documents. The vocabulary size (unique words) is 141,321. The data is not shuffled. After filtering out terms that did not occur in at least two documents, we end up with a vocabulary of size 51,064. This is a larger change than the sentiment dataset, despite beginnging with a larger vocabulary, likely because newsgroups contains many terms that were not relevant to a majority of the documents, instead being particular to their groups. The number of positive instances averaged across all classes is 942, around 5%.

Reuters-21578, Distribution 1.0 Obtained from NLTK[7] originally containing 10788 documents. After removing empty and duplicate documents the result is 10655 documents. Originally contained 90 classes, but as they were extremely unbalanced all classes that did not have at least 100 positive instances were removed, resulting in 21 classes. The original vocabulary

[4]Obtained by: https://keras.io/datasets/, Originally from https://ai.stanford.edu/ amaas/data/sentiment/ [48]

[5]http://qwone.com/ jason/20Newsgroups/

[6]https://scikit-learn.org/0.19/modules/generated/sklearn.datasets.fetch_20newsgroups.html#sklearn.datasets.fetch_20newsgroup

[7]https://www.nltk.org/book/ch02.html

size is 51,001 and all words that did not occur in at least two documents were removed, resulting in a vocabulary size of 22,542. The number of positive instances averaged across all classes is 541, around 5%.

Placetypes It originally has a vocabulary size of 746,527 and 1383 documents. This is a very large vocabulary size to document ratio. The end vocabulary for this space was of size 100,000 due to the hard limit. This is roughly equivalent to removing all documents that would not be in at least 6 documents. As most classes in this domain are extremely sparse (less than 100 positive instances) no classes are deleted. As 8 of these remaining classes had a low number of positive occurrences, OpenCYC classes are removed that do not have positive instances for at least 30 documents, leaving us with 17. For the Geonames taxonomy, the same rule resulted in only 7 of 9 categories being used.

Movies Another large dataset with a vocabulary size of 551,080 and a document size of 15,000. However, after investigating the data made available by the authors, it was found that there were a number of duplicate documents. After removing these duplicate documents, there are 13978 documents. In the same way as the place-types, the vocabulary hit the hard limit of size 100,000.

3.4 Representations

For the bag-of-words representation used as a baseline, terms are additionally filtered out that do not occur in at least 0.001% of documents, as to scale with the amount of documents in each domain. From this filtered vocabulary, a bag-of-words is obtained by creating a matrix of documents and words, with the values of that matrix corresponding to how frequent each word was for each document. However, as frequency bag-of-words are not able to distinguish between frequent terms (e.g. "the") and important terms, words are weighed such that words which occur frequently in a small amount of documents are given a higher value than those that occur frequently in a large amount of documents. To do this, Positive Pointwise Mutual Information (PPMI) scores are used, following success in similar work by [19]. See Section 2.3.1 for more detail.

For the work in the following chapters, we wanted a variety of different vector space models. Below the choices for the vector space models that are formally described in Section 2.5 are

explained:

Multi-Dimensional Scaling (MDS) (See Section 2.5.2): Multi-Dimensional Scaling (MDS) is used for comparison, as it was the only space used in the work that introduced this method [18]. In this case, the input is a matrix of dissimilarity values between the PPMI vectors of documents of size nXn, where n is the number of documents.

Principal Component Analysis (PCA) (See Section 2.5.1): We use PCA as a linear transformation of the PPMI weighted BoW vectors, as it is a standard dimensionality reduction technique used historically and prevalently today to serve as a baseline reference.

Doc2Vec (D2V) (See Section 2.5.4): Doc2Vec is inspired by the Skipgram model [43]. It is distributional in the sense that the context of words and documents is used during its learning process. It is used here as a it is of a recent class of neural embedding models, which has been reported in the literature to perform well in document classification tasks. For the Doc2Vec space, the following hyper-parameters are tuned:

- The $windowsize(5, 10, 15)$ referring to the window of the words that are used as context during training
- The $mincount(1, 5, 10)$ referring to the minimum frequency of words
- The $epochs(50, 100, 200)$ of the network for each size space.

Average Word Vectors (AWV): Finally, we also learn a document embedding by averaging word vectors, using a pre-trained GloVe word embeddings (See Section 2.5.3) that was trained on the Wikipedia 2014 + Gigaword 5 corpus[8]. While simply averaging word vectors may seem naive, this was found to be a competitive approach for unsupervised representations in several applications [28]. We simply average the vector representations of the words that appear at least twice in the BoW representation. Strangely, we found that this performed better than weighing the words on frequency or PPMI.

To determine the best doc2vec model for each task in each domain, and to investigate the quality of these representations, a linear SVM is used that takes the document representations as input. This SVM is also hyper-parameter tuned to find the best C values $C(1.0, 0.01, 0.001, 0.000)$,

[8] https://nlp.stanford.edu/projects/glove/

and if the weights should be balanced such that positive instances are weighted in proportion to how rare they are $balanced(0, 1)$ (See Section

Unfortunately, doc2vec representations could not be obtained for the movies or place-types domains as the original full text was not available, only the bag-of-words.

Chapter 4

Disentangling unsupervised document embeddings

4.1 Introduction

Vector space models encode meaning spatially, but their features are not meaningful. However, they enable strong results to be achieved in a variety of domains and are flexible in how they can be learned, e.g. by integrating word-context to achieve strong results on sentiment tasks [57], learning visual data alongside text data to explain the content of images [49], and enforcing grammatical structure to perform better at question answering tasks [55].

This chapter is about disentangling document embeddings (vector space models of documents) into semantic features across a variety of domains and document embedding models. For example, where documents are concatenated movie reviews for a particular movie (See Section 3.2), features can be obtained like how "Scary" a movie is, or how "Romantic" a movie is.

This chapter follows work by Derrac [19], who first introduced a method to derive semantic features from a vector space representation. The method begins with the following assumption: if documents in a document embedding can be linearly separated based on binary word occurrence (where a document has a label of 1 if the word occurs and 0 otherwise), that word is semantically important in the domain. Such words can be found in an unsupervised way by training a linear model, e.g. a linear Support Vector Machine (SVM) (See Section 2.4.3), where the semantic importance of words in the domain can be evaluated using standard model evaluation metrics like F1-score (see Section 2.4.6), as this measures how linearly separable documents are for that term.

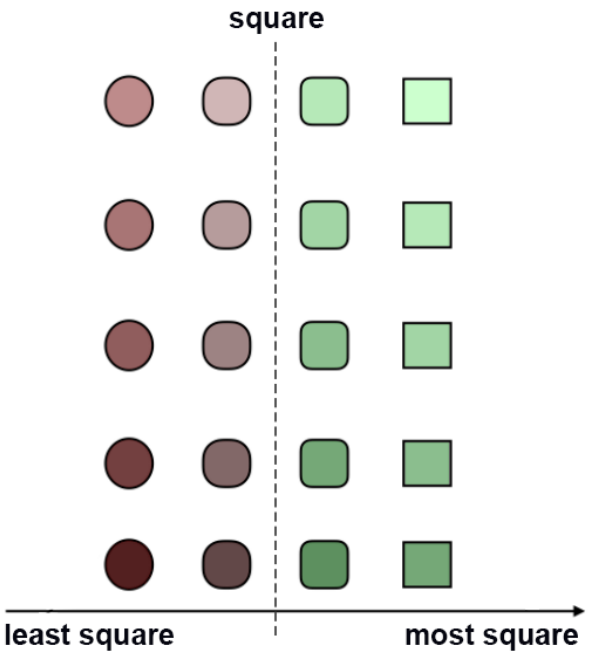

Figure 4.1: An example of a hyper-plane in a toy domain of shapes. The hyper-plane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples. Those closest to the hyper-plane are less square than those further away.

The Linear SVMs obtain a hyper-plane for each model to separate documents that contain the word and documents that do not contain it. An example of this is shown in a two-dimensional toy domain of shapes in Figure 4.1, where the dotted line represents a hyper-plane. As shown in this example, it can be assumed that documents furthest from the hyperplane on the negative side are the least representative of the feature being captured by the hyper-plane, in this case the 'squareness' of a shape, and the documents that are furthest from the hyper-plane on the positive side are the most representative, while those closest to the hyper-plane are more ambiguous. By simply taking the coefficients (orthogonal vector) of this hyper-plane, a direction can be obtained that goes from documents that are the most distant from the hyper-plane on the negative side, to those that are most distant from the hyper-plane on the positive side (see the direction shown at the bottom of the example in 4.1).

By measuring how far up a document is in the orthogonal direction vector for a word, ranking of documents on that word can be obtained, e.g. how 'Funny' it is relative to the other documents.

IMDB Movie Reviews	Flickr-Placetypes	20-Newsgroups
courtroom legal trial court	broadway news money hollywood	switzerland austria sweden swiss
disturbing disgusting gross	fir bark activism avian	ham amp reactor watts
tear cried tissues tears	palace statues ornate decoration	karabag armenian karabakh azerbaijan
war soldiers vietnam combat	drummer produce musicians performers	4800 parity 9600 bps
message social society issues	ubahn railways electrical bahn	xfree86 linux
events accuracy accurate facts	winery pots manor winecountry	umpires umpire 3b viola
santa christmas season holiday	steeple religion monastery cathedral	atm hq ink paradox
martial arts kung	blanket whiskers fur adorable	lpt1 irq chipset mfm
bizarre weird awkward	desolate eerie mental loneliness	manhattan beauchaine bronx queens
drug drugs dealers dealer	carro shelby 1965 automobiles	photoshop adobe
inspirational inspiring fiction narrative	relax dunes tranquil relaxing	reboost fusion astronomers galactic

Table 4.1: Example features from three different domains, where each cluster of words corresponds to a cluster direction.

After repeating this process for all documents on a word direction, features that rank documents on a variety of words can be obtained. To get a representation that contains less noise, only those words that are well spatially separated in the representation can be included. This is the first step of the approach described by Derrac [19] towards disentangling the features encoded spatially in a document embedding in-order to obtain semantic features. Specifically, first hyper-planes are obtained for all words based on binary frequency. Then, words that are semantically important are scored (e.g. by accuracy of the SVM classifier). Finally, orthogonal directions for the highest scoring word hyper-planes are obtained and documents are ranked on how far up they are on these highest-scoring directions. These rankings are then used as the new disentangled representation.

However, directions labelled with single words, although direct, can be ambiguous. For example, in a domain of IMDB movie reviews "numbers" could be referring to musical "numbers", or mathematical "numbers". To resolve this similar words can be clustered together e.g. "numbers" would be clustered with words as follows "numbers singing songs musical song dance dancing sings sing broadway". This can be done with an off-the-shelf clustering algorithm like k-means (see 4.3.3) that uses the word direction vectors as input. Then, a new direction can be obtained that more accurately models the feature described by the cluster by, for example, taking the cluster-centers output by the k-means algorithm as the new direction vectors. We show examples of the labels associated with these cluster features in a variety of domains in 4.1.

The overall goal of this chapter is to perform a deeper examination of the method introduced by

Derrac [19] to obtain semantic features. This chapter builds on the method introduced by Derrac [19] to disentangle a document embedding into semantic features. Derrac's quantitative investigation was of disentangled cluster feature representations derived from a Multi-Dimensional Scaling 2.5.2 document embedding model, used as input to rule-based classifiers. The domains investigated by Derrac were the Place-types and Movies domains, as well as a domain of text describing wines, and the evaluation was of a rule-based classifier that uses a feature representation as input. The first important contribution of this chapter is an extensive quantitative examination in Section 4.5 and qualitative investigation in Section 4.4 across five domains (as described in chapter 3), where disentangled feature representations are derived using four document embedding models. One contribution of the work in this thesis is a method to evaluate these semantic features. In this chapter to evaluate a disentangled feature representation, low-depth decision trees are trained to classify key domain tasks (e.g. classifying the 20 newsgroups categories) using these disentangled feature representations as input. A low-depth decision tree uses a limited number of features to classify, for example a decision tree limited to a depth of one can only use a single feature. If a low-depth decision tree can perform well on a key domain task using the disentangled feature representation as input, then it must be composed of relevant features in the domain. Another contribution of this chapter is the introduction of two variants to the method introduced by [19], specified in Section 4.3. The first variant is a new scoring metric, Normalized Discounted Cumulative Gain (NDCG) that measures the quality of the ranking rather than the performance of the linear classifier. The second variant is the introduction of standard k-means as an alternative clustering algorithm. It is found that both of these variants perform well in a majority of cases.

4.2 Background

4.3 Method

This section details the methodology to disentangle a document embedding model starting with the document vectors and their associated bag-of-words representations. The work in this chapter differs from the method introduced by Derrac [19] as it focuses on investigating the

quality of the semantic features in these disentangled representations. In particular, two variations are introduced and included as part of the experimental method.

4.3.1 Obtaining Directions and Rankings From Words

The method starts with a given document embedding induced from text documents and their associated bag-of-words. The representation of each document in the bag-of-words is composed of word frequencies $d = (wf_1, wf_2, ..., wf_n)$ where $wf(d)$ is the frequency of of the word in the document d, and n is equal to the number of unique words in the vocabulary W.

Obtaining directions for each word

Where *wdf* is the number of documents that a word occurs in, directions are only obtained for words higher than a threshold *wdf* $> T$. Each document is represented by a vector v in the document embedding. For each word w, a linear classifier[1] is trained to separate documents v where the word w occurs more than once, from documents where the word does not occur. This task is unbalanced, i.e. there are typically fewer documents that contain the word compared to those that do not contain it. For this reason, the weights of the training examples are balanced such that positive instances are weighted in proportion to how rare they are.[2]

The distance between the document vectors v and the hyperplane h_w is meaningful, as it can be expected that the documents which contain the word more frequently are further away from the hyper-plane on the positive side. Following this intuition, a direction for a word $\mathcal{D}_w$ in the space can be obtained by taking the vector perpendicular to the hyperplane h_w. We give an example of two directions in Figure 4.2. Here, shapes that are "more square" are far away from the hyper-plane on the right side, and shapes that are more circular are further away from the hyper-plane on the left side. To give an example in a real domain, a direction for the word "Scary" would have the most "Scary" movies are at the tip of the direction and those that are least 'Scary' at the base of the direction.

[1] Tested using a logistic regression classifier and a linear SVM, both achieved similar results.

[2] Using scikit-learn, class_weight:'balanced'

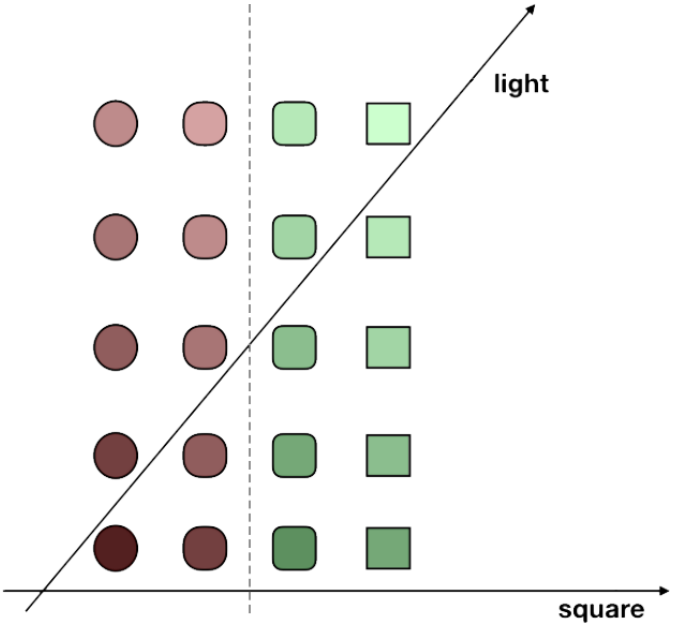

Figure 4.2: Another example of a hyper-plane in a toy domain of shapes. Here we show multiple directions, one for light and one for square The hyper-plane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples for the word square.

Ranking documents on directions

Let $\mathcal{D}_w$ be the vector which is perpendicular to the hyperplane h_w. This vector can be used to induce a ranking of the documents that intuitively captures how closely each document is related to the word w. Specifically, let us write rw_d for the dot product $\mathcal{D}_w \cdot p_d$. Then, d_1 is ranked higher than d_2 for word w if $rw_{d_1} > rw_{d_2}$. By this process, a feature is obtained that ranks documents on a word's direction. These rankings measure how relevant the document is in the spatial representation for the word, rather than just frequency e.g. a document that contains the word "scary" but isn't a scary movie (e.g. if it contained sentences like "it's scary how much money is spent on advertising movies like this") might not be ranked highly on the direction for 'scary', as the word 'scary' is not semantically important for the document. To put it another way, intuitively it can be understood to mean that the document d_2 'has' the feature to a greater extent than d_1, e.g. in a domain of movie reviews if a movie ranked highly on the word 'dull', the movie has more dullness than lesser ranked movies.

Summary

In this section, the methodology to obtain word-directions and their associated rankings was described. However, some words are more semantically important in the domain than others, and it is not yet clear which of these features are semantic. The following section describes how to remove word directions that are not well predicted by the linear model, under the assumption that if they are not well represented in the space they are not semantically important.

Another problem with these word features is that their meaning can be unclear, e.g. the word "serial" could be referring to a series of movies, or a "serial" killer[3]. To solve this problem, section 4.3.3 explains a method to associate word directions with similar word directions, and methods to cluster word directions together. We gave examples of these clusters of words in the introductory table 4.1.

4.3.2 Filtering Word Directions

Although we are able to obtain word directions for every word, not every word is semantic. This section describes how to filter out word-directions that are not semantic, in-order for the final representation to be composed of only semantic features.

The assumption made by Derrac in the original work [19] was that if a linear classifier does not predict the occurrence of a word in a document in its embedding well, it is not semantically important. Put another way, if the documents are separated well, it must mean that the word w being used in the description of d is important enough to affect the document embedding model representation of d. This can be evaluated by using a variety of scoring metrics to determine the performance of the linear classifier.

However, this work also introduces the use of a scoring metric that evaluates the quality of the direction $\mathcal{D}_w$, as even if the documents are well separable, then the ranking induced from the direction may not be correct. This metric compares how well the ranking induced by the hyperplane correlates with a BoW representation. If the ranking correlates strongly, it can be

[3]The real cluster of words that this example comes from is "gore gory bloody blood gruesome serial investigate deaths"

assumed that this means the word was strongly influential in the document embedding model, as the detail of the Bag-Of-Words information is embedded in the document embedding structure.

Below, we introduce three potential scoring metrics.

Cohen's Kappa. This is the only metric used in the work by Derrac [19]. This metric evaluates the performance of the classifier, and also deals with the problem that these words are often very imbalanced. For very rare words, a high accuracy might not necessarily imply that the corresponding direction is accurate, as if there are a large number of negative examples (as is the case with infrequent words) the classifier could simply predict that all documents do not contain the word to achieve a high score. For this reason, they proposed to use Cohen's Kappa score instead. In our experiments, however, it was found that this can be too restrictive, allowing us to sometimes obtain better results with the more simple accuracy metric.

Classification accuracy. If a model has high accuracy for a word w, it seems reasonable to assume that w describes a salient property for the given domain. However, despite balancing the weights of the original SVM used to obtain the hyper-plane, the value this metric places on correctly predicting negative classification compared to Kappa, it might favour rare words as it tends to be easier to obtain a high accuracy for these words.

F1-score. As described in Section 2.4.6 F1 score is the harmonic mean of recall and precision, and is used to balance and measure the recall and precision at the same time where they are equally important. It is used in this case under the assumption that it can resolve the imbalance issue of accuracy while scoring properties differently to the Kappa score.

Normalized Discounted Cumulative Gain This metric evaluates the quality of the rankings induced by the direction $\mathcal{D}_w$, rather than the performance of the linear classifier. To be specific, it evaluates how well a ranking induced from the word direction matches the Positive Pointwise Mutual Information (See Section 2.3.1) scores for the word in each document. Normalized Discounted Cumulative Gain (NDCG) is a standard metric in information retrieval that evaluates the quality of a ranking w.r.t. some given relevance scores [33].

First, the ranking of documents on a word R_w obtained using the dot product on a direction D_w is converted into a numeric integer ranking where each document is numbered based on its position in the ranking. The relevance scores are determined by the Pointwise Positive Mutual

Information (PPMI) score $PPMI(w, d)$, of the word w in the BoW representation of document d (See section 2.3.1).

$$\text{DCG}_R^w = \sum_{i=1}^{pr_d} \frac{ppmi_i^w}{log_2(i+1)}$$
$$\text{IDCG}_R^w = \sum_{i=1}^{|documents|} \frac{2^{ppmi_i^w} - 1}{log_2(i+1)}$$
$$\text{nDCG}_R^w = \frac{\text{DCG}_R^w}{\text{IDCG}_R^w}$$

To define NDCG, we can first define Discounted Cumulative Gain (DCG), where pr_d is equal to the numbered position of document d, and $ppmi_i^w$ is equal to the PPMI score for a word at position i in the ranking. Then, we can define the Ideal Discounted Cumulative Gain (IDCG), which is the best possible DCG for a position pr_d, where $|documents|$ are the documents for the term ordered according to the relevance scores up to position pr_d. NDCG is then simply the DCG normalized by the iDCG.

4.3.3 Clustering Features

It can sometimes be ambiguous what a feature-ranking means when it is labelled with only a single word, e.g. the word "courage" has an associated feature direction, but what that feature direction represents can only be understood in the context of a cluster of similar words "courage students teaches student schools teacher teach classes practice training learning overcome conflict teaching" showing that it is about courageous teachers and students overcoming challenges. The most naive way to obtain a cluster like this is to find similar word-directions:

Similarity Labelling

As a way to add context to single word directions, for each single word direction $\mathcal{D}_w$, the cosine similarity is calculated between it and every other word direction. Then, the top n most-similar words are used to label the original word direction.

Clustering

Given a number of clusters j, a clustering algorithm can use the word directions as input to produce j clusters, where each cluster c is composed of words $c = (w_1, w_2, ..., w_n)$, and a new cluster direction that is associated with each cluster $\mathcal{D}_C$. This cluster direction represents all the words in the cluster. In the same way as described in Section 4.3.1 a ranking can be obtained from this cluster direction to obtain a cluster feature. This has the same benefits of the previous method as more words are associated with features, making them easier to understand. However, cluster directions model new features formed by the clustering process, e.g. if the word directions for the words "Bloody" and "Gorey" are clustered together, then the corresponding cluster direction will model both of these terms. A cluster direction may be desirable if both "Bloody" and "Gorey" are words used in movie reviews to describe the feature of how much blood a movie contains, as the averaged cluster direction will better model that feature if it includes movie reviews that describe blood using the term "Bloody" and movie reviews that describe blood using the term "Gorey". Essentially, the cluster feature-direction could more accurately represent the semantics of a bloody film, compared to what is possible when considering either feature-direction individually. Clustering can also reveal properties that were not clearly captured by any of the individual words, e.g. the direction "Gripping" has a clear individual meaning, but when clustered with the terms "Emotional", "Families", and "Complex" the new clustered direction is modelling a new feature that none of the terms alone encompass, specifically that of an emotional drama featuring families and relationships.

On the other hand, it is possible that clustering may result in a cluster feature that is less relevant to a task. For example, a clustering algorithm may have a cluster $\{Romance, Love\}$ and choose to add to that cluster the feature-direction $\{Cute\}$, as the term "Cute" has been used in reviews for romance movies. However, the term "Cute" has also been used in reviews for movies containing cute animals. This would make the new clustered direction $\{Romance, Love, Cute\}$ perform worse at classifying the movie genre "Romance", but a bit better at classifying if a movie contains animals. It might thus be preferable to keep *"Cute"* in a separate cluster that represents animal movies rather than a cluster that represents romantic movies with cute animals. In a task where the objective is to cluster if a movie is in the "Romance" genre, it can be expected that clustering will perform worse than single directions, as there is no longer a feature that corresponds to that specific class.

In-order to help find good cluster directions that correspond to the task, we introduce the use of k-means alongside the clustering algorithm introduced in the work by Derrac [19]. For both clustering methods, the only parameter tuned is the amount of clusters:

Derrac's K-Means Variation

This is the clustering method used in the work this method was introduced in [19]. As input to the clustering algorithm, it considers the N best-scoring candidate feature directions v_w. Hyper-parameter selection is performed by varying the scoring method or amount of candidate feature directions N. The main idea underlying their approach is to select the cluster centers such that (i) they are among the top-scoring candidate feature directions, and (ii) are as close to being orthogonal to each other as possible.

The output of this clustering algorithm is a set of cluster directions $CD = D_c1, ..., D_cK$, where each cluster direction is labelled with i of words it is composed of $L(w_1, w_2, ..., w_n)$. Note that these a feature representation is obtained for these cluster directions as described in the previous section. In the following, we will write C_j to denote the centers of the directions corresponding to the words in the cluster L_j, and $\mathcal{D}_w$ for the direction corresponding to the word in that cluster.

The first cluster is chosen by taking the top-scoring direction for the chosen scoring metric. Then, the new center C is selected by choosing the word-direction that is least similar to all of the current centers:

$$C = \min(\max_{C_j \in CD}(cos_{\mathcal{D}_w i \in W_n}(\mathcal{D}_w i, v_{C_j}))) \tag{4.1}$$

Where W_n are the candidate word-directions, and CD is the current set of cluster centers. Once k clusters have been selected, each candidate direction $\mathcal{D}_w i$ is added to its most similar cluster $max_{C_j \in CD}(cos(\mathcal{D}_w i, C_j)$ and after adding that direction the cluster direction is updated to be the average of the current center and the newly added word $AVG(C_j, w_i)$. This continues until all candidate directions W_n are added to clusters. The cluster centers CD are taken as the final cluster centers.

K-Means

K-means is a standard baseline clustering algorithm. In the experimental results, it was found that Derrac's variation relies too much on the initial choice of cluster centers, meaning that key directions may be missed if terms related to them are not initially chosen. Avoiding this is difficult without extensive and sometimes arbitrary hyper-parameter optimization. K-means is used as an alternative baseline that does not have this problem. Default parameters and the The k-means algorithm is implemented in scikit-learn with default parameters [4].

Given the input X, traditional k-means begins with K centers chosen uniformly at randomly from X. In this work we use the K-means++ variation [5], where the first center is chosen uniformly at random from X and the remaining centers are chosen with the following probability, where $D(x)$ is the shortest distance from a data point $x \in X$ to the closest center that has already been chosen:

$$\frac{D(x)^2}{\sum_{x \in X} D(x)^2} \tag{4.2}$$

After this initialization, the standard k-means process is followed. The distance between each input x and center c is calculated. In-order for the Euclidean distance used in k-means to be meaningful, the vector directions are normalized. In particular the relationship between them is:

$$cos(angle) = 1 - dist/2 \tag{4.3}$$

Each input x is then assigned to its closest center c. Then, the centers are recomputed to be the mean of their assigned inputs. This process starting with the distance calculation is repeated until the centers do not change or a maximum number of iterations is reached (in this case, the default parameter of 300 iterations is used).

4.4 Qualitative Results

In this section, a variety of different parameters are investigated qualitatively in-order to better understand how they affect the semantic features in the disentangled representation. To give

[4]The default parameters in scikit-learn were used for k-means

an intuition of what is semantic in each domain, the highest scoring terms are shown from a variety of domains in Section 4.4.1. Then, the differences between different document embedding models in the domain of movies is investigated in Section 4.4.2. Next, in Section 4.4.3 the differences between scoring metrics is investigated by looking at the top scoring terms for each one. Finally in Section 4.4.3 the doc2vec representation is investigated in the newsgroups domain, as it could not be investigated in the movies domain.

4.4.1 The Highest Scoring Directions for each Domain

To give an understanding of the kind of directions found for each domain, the top-scoring ones are presented in Table 4.2. The choice of document embedding model and scoring metric for this table were determined by what performed best on an associated domain task when using the single term features as input to a depth-3 decision-tree. For the movies, the task that determined which document embedding model and scoring metric is shown was the Genre classification task, and for the Flickr tag place-types the task was classifying according to the Foursquare taxonomy. For the other domains, we similarly used the associated classification task. Note that for each term feature, the table shows the two words with the most similar direction, as described in Section 4.3.3.

There is an interesting difference between the sentiment directions and the movies directions in Table 4.2. In the movies domain, properties of movies e.g. genre related directions like "Horror" or "Romantic" are the highest scoring, meaning they are the most important spatially, and in-turn semantically in the representation. However, in the sentiment domain the properties that are highest scored are names of actors and actresses. Both of these domains are composed of movie reviews, but the documents in the former are a concatenation of a number of reviews across different sources, while the latter are individual reviews. Because of this, despite both domains being composed of movie reviews, the movies domain captures more general properties about movies. Similarly For the newsgroups domain, directions that are good at categorizing certain newsgroups are found, as the data is split into newsgroups e.g. the word 'celestial' applying to religious newsgroups, or the "diesel" and "porsche" directions relating to the newsgroup that discusses cars ("auto"). In the case of the place types domain, we generally find objects that occur in the photo "stream", "wilderness", "cliff", adjectives describing them "peaceful",

Movies *(50 MDS NDCG)*	**Sentiment** *(100 D2V NDCG)*	**Newsgroups** *(50 D2V NDCG)*	**Place-types** *(50 AWV Kappa)*	**Reuters** *(200 MDS NDCG)*
horror (scares, scary)	glenda (glen, matthau)	karabag (iranian, turkiye)	hike (peak, climbing)	franklin (fund, mthly)
hilarious (funniest, hilarity)	scarlett (gable, dalton)	leftover (flaming, vancouver)	stream (waterfall, fountains)	quarterly (shearson, basis)
bollywood (hindi, india)	giallo (argento, fulci)	wk (5173552178, 18084tmibmclmsuedu)	arquitectura (edificio, architektur)	feb (28, splits)
laughs (funnier, funniest)	bourne (damon, cusack)	1069 (mlud, wibbled)	peaceful (serene, tranquil)	22 (booked, hong)
jokes (gags, laughs)	piper (omen, knightley)	providence (norris, ahl)	cathedral (roman, religious)	april (monthly, average)
comedies (comedic, laughs)	casper (dolph, damme)	celestial (interplanetary, bible)	architektur (architectuur, fenster)	sets (principally, precious)
hindi (bollywood, india)	norris (chuck, rangers)	mlud (wibbled, 1069)	landscapes (soe, flickrdiamond)	16 (creditor, trillion)
war (military, army)	holmes (sherlock, rathbone)	endif (olwm, ciphertext)	stones (mygearandme, flickrsbest)	1st (qtr, pennsylvania)
western (outlaw, unforgiven)	rourke (mickey, walken)	gd3004 (35894, intergraph)	wilderness (glacier, peak)	26 (approve, inadequate)
romantic (romance, chemistry)	ustinov (warden, cassavetes)	rtfmmitedu (newsanswers, ieee)	nationalpark (hike, peak)	23 (offsetting, weekly)
songs (song, tunes)	scooby (doo, garfield)	eng (padres, makefile)	tranquil (serene, peaceful)	prior (recapitalization, payment)
sci (science, outer)	doo (scooby, garfield)	pizza (bait, wiretap)	waterfall (butterfly, stream)	avg (shrs, shr)
funniest (hilarious, funnier)	heston (charlton, palance)	porsche (nanao, mercedes)	geology (mineral, formations)	june (july, venice)
noir (noirs, bogart)	homer (pacino, macy)	gebcadredslpittedu (n3jxp, skepticism)	serene (tranquil, peaceful)	march (31, day)
documentary (documentaries, footage)	welles (orson, kane)	scsi2 (scsi, cooling)	cliff (cliffs, rocky)	regular (diesel, petrol)
animation (animated, animators)	frost (snowman, damme)	playback (quicktime, xmotif)	naturesfinest (goose, ilovenature)	4th (qtr, fourth)
adults (adult, children)	streisand (bridget, salman)	35894 (gd3004, medin)	foliage (branch, blossom)	27 (chemlawn, theyre)
creepy (spooky, scary)	davies (rhys, marion)	diesel (volvo, shotguns)	dome (mosaic, column)	14 (borrowing, borrowings)
gay (gays, homosexuality)	cinderella (fairy, stepmother)	evolutionary (shifting, hulk)	baroque (neoclassical, renaissance)	11 (chapter, ranged)
workout (intermediate, instruction)	boll (uwe, belushi)	techniciandr (obp, 144k)	concert (guitar, performance)	may (probably, however)
thriller (thrillers, suspense)	rochester (eyre, dalton)	8177 (obp, 144k)	waves (500d, diamondclassphotographer)	38 (33, strong)
funnier (laughs, funniest)	edie (soprano, vertigo)	shaw (medicine, ottoman)	cliffs (cliff, rocky)	m1 (m2, m3)
suspense (suspenseful, thrillers)	scarecrow (zombies, reese)	scorer (gilmour, lindros)	plaza (centro, streets)	dlr (writedown, debt)
arts (hong, chan)	kramer (streep, meryl)	xwd (xloadimage, openwindows)	flora (theunforgettablepictures, blueribbonwinner)	five (years, jones)
christianity (religious, religion)	marty (amitabh, goldie)	ee (275, xloadimage)	rocky (rugged, overlook)	bushels (soybeans, ccc)
musical (singing, sing)	columbo (falk, garfield)	com2 (com1, v32bis)	calm (serene, peaceful)	revs (net, 3for2)
gore (gory, blood)	kidman (nicole, jude)	examiner (corpses, brass)	branches (vivid, bushes)	29 (175, include)
animated (animation, cartoon)	juliet (romeo, troma)	migraine (ama, placebo)	mygearandme (platinumphoto, blueribbonwinner)	acquisition (make, usairs)
gags (jokes, slapstick)	garland (judy, lily)	parliament (parliamentary, armored)	stadt (ciudad, capitale)	payable (div, close)
sexual (sexually, sexuality)	hawn (goldie, matthau)	manhattan (bobbeviceicotekcom, beauchaine)	fauna (mammal, insect)	13 (dlrsbbl, groups)

Table 4.2: The top-scoring words for each domain

"tranquil", or awards that are given to sub-categories of photos like "mygearandme" (a group competition where users submit photographs). As this is the representation that scored well on the Foursquare task (See 3.2), it is unsurprising that the important directions are in these categories, as the classes are types of places. In this case, the nature-related directions are probably useful for classifying the "Parks and Outdoors" class. The reuters dataset is certainly a case where providing context is important, as the terms which were identified are mostly business jargon ("quarterly", "avg", "dlr", "1st (qtr)").

4.4.2 Comparing Document Embedding Models

In this section, the differences between the different document embedding models for the movies domain are illustrated. However, as the original text was not available for this domain, the doc2vec embeddings are not available. In these examples, document embedding models of size 200 and score-type NDCG are used as they generally perform well. Additionally, the top 20,000 most frequent words are used for learning the embeddings. In terms of quantitative performance, when MDS is used as input to a Linear SVM or Decision Tree, it performs better than the others in F1-score (See in the Quantitative Results section, Table 4.8). Theoretically, this means that it should contain unique natural directions that other document embedding models do not have.

In Table 4.3 common and unique terms between the document embeddings are shown. A term is common if the term occurs in the 2,000 top scoring terms of both the other two embeddings. A term is unique if neither the direction name, or the direction names provided as context in brackets occur in any of the other top 2,000 scoring terms for the other embeddings. This is to ensure that unique directions are shown, rather than unique terms that refer to the same direction. The commonalities between document embedding model are much more prevalent than the differences, with natural properties of the domain being represented in all of the different document embedding models.

When examining the table of results, the common terms are mostly salient properties relevant to the domain, e.g. "noir", "comedies", "western", "documentary". The space learned using MDS captures the most unique properties (39 versus 31 from PCA), and also captures some features that are relevant to the domain that others do not have, e.g. "kung (martial, jackie)", "com-

MDS	AWV	PCA	Common
berardinelli *(employers, distributor)*	billy *(thrown, dirty)*	amount *(leaving, pick)*	noir *(fatale, femme)*
crawford *(joan, davis)*	brother *(brothers, boys)*	fails *(fit, pick)*	gay *(homosexual, homosexuality)*
hitchcocks *(hitchcock, alfred)*	fonda *(henry, jane)*	pick *(fails, fit)*	prison *(jail, prisoners)*
warners *(warner, bros)*	building *(built, climax)*	stands *(fails, cover)*	arts *(rec, robomod)*
nuclear *(weapons, soviet)*	train *(tracks, thrown)*	surprisingly *(offer, fit)*	allens *(woody, allen)*
joan *(crawford, barbara)*	slaves *(slavery, excuse)*	copyright *(email, compuserve)*	jokes *(laughs, joke)*
kidnapped *(kidnapping, torture)*		length *(reflect, expressed)*	animation *(animated, cartoon)*
hop *(hip, rap)*		profanity *(reflect, producers)*	sherlock *(holmes, detective)*
kung *(martial, jackie)*		compuserve *(copyright, internetreviews)*	western *(westerns, wayne)*
ballet *(dancers, dancer)*		talents *(admit, agree)*	songs *(song, lyrics)*
gambling *(vegas, las)*		admit *(agree, talents)*	comedies *(comedic, laughs)*
alcoholic *(drunk, alcoholism)*		developed *(introduced, sounds)*	workout *(exercise, challenging)*
waves *(surfing, wave)*		intended *(bother, werent)*	laughs *(funnier, hilarious)*
jaws *(jurassic, godfather)*		constantly *(putting, sounds)*	drug *(drugs, addict)*
jungle *(natives, island)*		tired *(anymore, mediocre)*	sci *(science, fiction)*
employers *(berardinelli, distributor)*		produced *(spoiler, surprising)*	documentary *(documentaries, interviews)*
pot *(weed, stoned)*		involving *(believes, belief)*	students *(student, schools)*
canadian *(invasion, cheap)*		anymore *(continue, tired)*	thriller *(thrillers, suspense)*
murphy *(eddie, comedian)*		leaving *(fit, pick)*	allen *(woody, allens)*
comics *(comedian, comedians)*		makers *(producers, aspects)*	funniest *(hilarious, laughing)*
kidnapping *(kidnapped, torture)*		introduced *(developed, considered)*	gags *(jokes, slapstick)*
subscribe *(email, internetreviews)*		loses *(climax, suffers)*	adults *(children, adult)*
vegas *(las, gambling)*		negative *(positive, bother)*	animated *(animation, cartoon)*
distributor *(berardinelli, employers)*		expressed *(reflect, opinions)*	dancing *(dance, dances)*
wave *(waves, surfing)*		mildly *(mediocre, forgettable)*	teen *(teenage, teens)*
rhodes *(internetreviews, email)*		helped *(putting, allowed)*	soldiers *(soldier, army)*
hippie *(pot, sixties)*		reflect *(expressed, opinions)*	indie *(independent, festival)*
weed *(pot, stoned)*		opinions *(reflect, expressed)*	suspense *(suspenseful, thriller)*
caribbean *(pirates, island)*		frequently *(occasionally, consistently)*	creepy *(scary, eerie)*
eddie *(murphy, comedian)*		content *(agree, proves)*	italian *(italy, spaghetti)*
sixties *(beatles, hippie)*		allowed *(helped, werent)*	jews *(jewish, nazis)*
... 8 More		suffers *(lacks, loses)*	... 1480 more

Table 4.3: Unique terms between document embedding models

ics (comedian, comedians)", "kidnapping (kidnapped, torture)", "gambling (vegas, las)", despite also capturing some unique noise "berardinelli (employers, distributor)", "crawford (joan, davis)". The AWV space captures names, and properties which are interesting (train, slaves), but there are not many of them. Meanwhile PCA seems to capture many unique properties that are unrelated to the task, broadly in three categories: metadata from the review sites e.g. "copyright *(email, compuserve)*", "compuserve *(copyright, internetreviews)*", opinions related to the sentiment of the movie "negative *(positive, bother)*", "expressed *(reflect, opinions)*", "talents *(admit, agree)*" and words that are difficult to understand as they do not describe anything, but may be movie review jargon "stands *(fails, cover)*", intended *(bother, werent)*, "developed *(introduced, sounds)*".

4.4.3 Comparing Scoring Metrics

In Table 4.4 the same MDS document embedding as the previous section is used but the score-type is varied, in order to examine the differences and commonalities between score-types. Similarly to the previous section, the most consistently meaningful properties are those that are common to all score-types, e.g. "horror *(scares, scares)*", "laughs *(funnier, funnier)*", "thriller *(thrillers, thrillers)*". The top 2,000 NDCG terms performed best when used as input to a classifier on the genres task. Following this, it can be seen that a lot of properties are only found when using the NDCG scoring type compared to the other scoring types e.g. "gay *(homosexuality, sexuality)*", "satire *(parody, parodies)*", "marry *(married, marriage)*". This is quite different to the previous section, where despite MDS being the better performing document embedding type, it only contained some unique but meaningful terms. The top directions scored on the F1 metric are by and large difficult to understand, referring to names or specific aspects of the scene, e.g. "company *(sell, pay)*", "post *(essentially, purpose)*", "impression *(instance, reasons)*". Accuracy is similar, e.g. "bags *(listened, salvation)*", "summers *(verge, medieval)*", "woefully *(restless, knockout)*" is similar. For the Kappa scoring type, some unique sentiment related terms are found, has some unique sentiment related terms e.g. "flawless *(perfection, brilliantly)*", "shocked *(hate, warning)*" "garbage *(crap, horrible)*" but also seems to contain some metadata e.g. "featurette *(featurettes, extras)*", "critic *(reviewed, net)*", "reviewed *(rated, mail)*", and although it captures a couple of meaningful terms e.g. "guns *(gun, shoot)*", "flying *(air, force)*" it does not find unique meaningful and general properties the way NDCG does. These

NDCG	F1	Accuracy	Kappa	Common
gay *(homosexuality, sexuality)*	company *(sell, pay)*	kennedy *(republic, elected)*	definately *(alot, awesome)*	horror *(scares, scares)*
arts *(hong, chan)*	street *(city, york)*	bags *(listened, salvation)*	guns *(gun, shoot)*	laughs *(funnier, funnier)*
sports *(win, players)*	red *(numerous, fashion)*	summers *(verge, medieval)*	flawless *(perfection, brilliantly)*	jokes *(gags, gags)*
apes *(remembered, planet)*	project *(creating, spent)*	revolve *(sincerely, historian)*	mail *(reviewed, rated)*	comedies *(comedic, comedic)*
german *(germans, europe)*	mark *(favor, pull)*	locale *(foster, sharply)*	garbage *(crap, horrible)*	sci *(scifi, alien)*
satire *(parody, parodies)*	lady *(actress, lovely)*	cooler *(downward, reports)*	featurette *(featurettes, extras)*	funniest *(hilarious, hilarious)*
band *(rock, vocals)*	fire *(ground, force)*	spades *(ralph, medieval)*	complaint *(extra, added)*	creepy *(spooky, spooky)*
crude *(offensive, offended)*	post *(essentially, purpose)*	filmography *(ralph, experiments)*	mission *(enemy, saving)*	thriller *(thrillers, thrillers)*
dancing *(dance, dances)*	heads *(large, throw)*	quentin *(downward, anime)*	ruin *(wondering, heck)*	funnier *(laughs, laughs)*
restored *(print, remastered)*	water *(land, large)*	employers *(finishes, downward)*	wars *(forces, enemy)*	suspense *(suspenseful, suspenseful)*
drugs *(drug, abuse)*	road *(drive, trip)*	formal *(victory, kennedy)*	prefer *(compare, added)*	gore *(gory, gory)*
church *(religious, jesus)*	brother *(son, dad)*	tube *(esta, muscle)*	heroes *(packed, hero)*	gags *(jokes, jokes)*
sexuality *(sexual, sexually)*	party *(decide, hot)*	woefully *(restless, knockout)*	necessarily *(offer, draw)*	science *(sci, sci)*
sexually *(sexual, sexuality)*	badly *(awful, poorly)*	scientists *(hilarity, locale)*	portray *(portrayed, portraying)*	gory *(gore, gore)*
england *(british, english)*	limited *(aspect, unlike)*	overboard *(civilized, cinderella)*	critic *(reviewed, net)*	government *(political, political)*
ocean *(sea, boat)*	impression *(instance, reasons)*	rumors *(homosexuality, characteristics)*	reviewed *(rated, mail)*	suspenseful *(suspense, suspense)*
marry *(married, marriage)*	trip *(journey, road)*	salvation *(bags, cooler)*	saving *(carry, forced)*	frightening *(terrifying, terrifying)*
campy *(cult, cheesy)*	michael *(producers, david)*	actively *(assassination, overcoming)*	technical *(digital, presentation)*	military *(army, army)*
christian *(religious, jesus)*	memory *(forgotten, memories)*	stretching *(victory, hideous)*	statement *(exist, critical)*	slapstick *(gags, gags)*
melodrama *(dramatic, tragedy)*	james *(robert, michael)*	downward *(cooler, crawling)*	shocked *(hate, warning)*	scary *(scare, scare)*
sing *(singing, sings)*	thin *(barely, flat)*	rocked *(staple, demented)*	flying *(air, force)*	blu *(unanswered, ray)*
sentimental *(touching, sappy)*	pre *(popular, include)*	affectionate *(esta, muscle)*	danger *(dangerous, edge)*	internetreviews *(rhodes, rhodes)*
depressing *(bleak, suffering)*	faces *(constant, unlike)*	protest *(protective, assassination)*		cgi *(computer, computer)*
evidence *(investigation, accused)*	values *(exception, wise)*	confined *(cooler, downward)*		email *(web, web)*
adorable *(cute, sweet)*	unusual *(odd, seemingly)*	inhabit *(quentin, drawback)*		thrilling *(thrill, exciting)*
episodes *(episode, television)*	lovers *(lover, lovely)*	latin *(communities, mount)*		web *(email, email)*
teenager *(teen, teenage)*	frame *(image, effect)*	reception *(como, finishes)*		horror *(scares, scares)*
magical *(fantasy, lovely)*	mans *(ultimate, sees)*	uptight *(suspensful, stalked)*		laughs *(funnier, funnier)*
health *(medical, suffering)*	efforts *(generally, nonetheless)*	brink *(inexplicable, freddy)*		suspense *(suspenseful, suspenseful)*

Table 4.4: Terms unique to different scoring metrics in the movies domain

qualitative examples contribute to the idea that NDCG, by evaluating the ranking rather than the separability, ends up discovering unique properties that are applicable to general domain tasks.

Investigating Doc2Vec In The Newsgroups Domain

The previous examples were from the movies domain, and could therefore not include the doc2vec representation. For that reason, the 20 newsgroups domain is used to investigate the doc2vec space. In Table 4.5 a comparison is shown between an MDS space, used as a representative of document embedding representations that use Positive Pointwise Mutual Information as their initial data input, and doc2vec. Doc2vec has achieved strong performance on a variety of baselines [68]. The aim of this analysis is to explore whether word-vectors and word-context can find interesting unique directions compared to MDS obtained from a PPMI BOW. In general, it is found that MDS captures properties that are less relevant to the task compared to D2V, specifically related to insignificant conversational words, e.g. "hi *(*folks, everyone)", "sorry *(*guess, hear)" "say *(*nothing, anything)" that do not seem related to capturing e.g. information that can separate entities into their original newsgroups. It seems that doc2vec was better at recognizing these words as noise and uninteresting compared to MDS, which must have prioritized these words. Doc2Vec represents interesting unique properties, e.g. "cryptology *(*attendees, bait)", which is very relevant to the newsgroup topic of cryptography. As expected, the common words are relevant to the task. It can be expected that by using word vectors, Doc2Vec is able to more easily identify interesting words and de-prioritize words which are common to the English language despite potentially being more rare in a smaller dataset.

4.5 Quantitative Results

This section evaluates the semantic features using low-depth decision trees. To begin, the evaluation method is explained in Section 4.5.1. This is followed by which hyper-parameters that are used for the classifiers in Section 4.5.2. Finally, the experiments are introduced.

Following this, the experimental results are obtained. First, a summary of all the results is shown in Section 4.5.3. Then, the following experiments are put into perspective by obtaining results for a bag-of-words of PPMI values, topic models, and document embedding vectors as

D2V	MDS	Common
leftover (pizza, brake)	hi (folks, everyone)	chastity (shameful, soon)
wk (5173552178, 18084tmibmclmsuedu)	looking (spend, rather)	n3jxp (gordon, gebcadredslpittedu)
eng (padres, makefile)	need (needs, means)	skepticism (gebcadredslpittedu, n3jxp)
porsche (nanao, 1280x1024)	post (summary, net)	anyone (knows, else)
diesel (cylinders, steam)	find (couldnt, look)	gebcadredslpittedu (soon, gordon)
scorer (gilmour, lindros)	hello (kind, thank)	intellect (soon, gordon)
parliament (caucasus, semifinals)	david (yet, man)	please (respond, reply)
atm (padres, inflatable)	got (mine, youve)	thanks (responses, advance)
cryptology (attendees, bait)	go (take, lets)	email (via, address)
intake (calcium, mellon)	question (answer, answered)	know (let, far)
433 (366, 313)	interested (including, products)	get (wait, trying)
ghetto (warsaw, gaza)	list (mailing, send)	think (important, level)
lens (lenses, ankara)	sorry (guess, hear)	good (luck, bad)
rushdie (sinless, wiretaps)	heard (ever, anything)	shafer (dryden, nasa)
immaculate (porsche, alice)	cheers (kent, instead)	bobbeviceicotekcom (manhattan, beauchaine)
keenan (lindros, bosnian)	say (nothing, anything)	dryden (shafer, nasa)
boxer (jets, hawks)	number (call, numbers)	im (sure, working)
linden (mogilny, 176)	mailing (list, send)	sank (bronx, away)
candida (yeast, noring)	call (number, phone)	banks (soon, gordon)
octopus (web, 347)	thank (thanx, better)	like (sounds, looks)
czech (detectors, kuwait)	read (reading, group)	shameful (soon, gordon)
survivor (warsaw, croats)	phone (company, number)	could (away, bobbeviceicotekcom)
5173552178 (circumference, wk)	mail (send, list)	would (appreciate, wouldnt)
18084tmibmclmsuedu (circumference, wk)	doesnt (isnt, mean)	beauchaine (bobbeviceicotekcom, away)
3369591 (circumference, wk)	lot (big, little)	ive (seen, never)
mcwilliams (circumference, wk)	thats (unless, youre)	surrender (soon, gebcadredslpittedu)
coldblooded (dictatorship, czech)	believe (actually, truth)	problem (problems, fix)
militia (federalist, occupying)	youre (unless, theyre)	windows (31, dos)
cbc (ahl, somalia)	send (mail, mailing)	gordon (soon, gebcadredslpittedu)

Table 4.5: Comparing terms that are unique to a MDS document embedding model to terms that are unique to a Doc2Vec model in the domain of newsgroups.

input to a variety of classifiers in Section 4.5.4. In Section 4.5.5 the results of a disentangled feature representation composed of only single-term features are obtained and compared to the baselines. Finally, in Section 4.5.6 a disentangled feature representation composed of cluster features is obtained and compared.

4.5.1 Evaluation Method

Low-depth CART Decision Trees are used (See Background Section 2.4.2) limited to a depth of one, two or three. The idea behind this is that if the rankings of documents on features are semantic, then the decision trees with a limited depth should be able to perform close to or

equal performance to the original results for the initial document embedding. If only the the performance of the disentangled feature representation was being evaluated, a classifier that does not limit the number of features could be used. However, the aim of this section is to show that the disentangled feature representation contains semantic features.

Essentially, if depth-one decision trees which only use a single feature to make a classification decision can model a class well using the disentangled representation as input, then the information in the original document embedding that enabled it to perform well on the task has been disentangled into a single feature. Specifically, this means that a single feature in the disentangled representation is able to directly model the associated domain task. Similarly, if the disentangled representation performs well on a depth-three limited decision tree, then the tree must be composed of semantic features relevant to the task. It can be expected that the features used in this tree, if performance is high, are semantic, as they generalize well to new examples, showing that they model the class. Figure 4.3 shows one of the trees that was obtained from our feature based representation. If the document embedding is indeed disentangled into a feature representation composed of important properties for the domain, then these low-depth decision trees should generalize well to test data. To show that the initial vectors of the document embeddings are entangled and the disentangled feature representations are disentangled, low-depth decision tree results are obtained for the initial document embedding vectors.

4.5.2 Hyper-Parameters

As stated previously (See Section 3.4) in all experiments the document embedding models for which results are obtained are Doc2Vec, Principal Component Analysis (PCA), Multi-Dimensional Scaling (MDS) and Averaged Word Vectors (AWV). As possible choices for the number of dimensions, we used $(50, 100, 200)$. The Doc2Vec[5] implementation from the gensim library is used.

[5]Doc2Vec implemented in gensim.

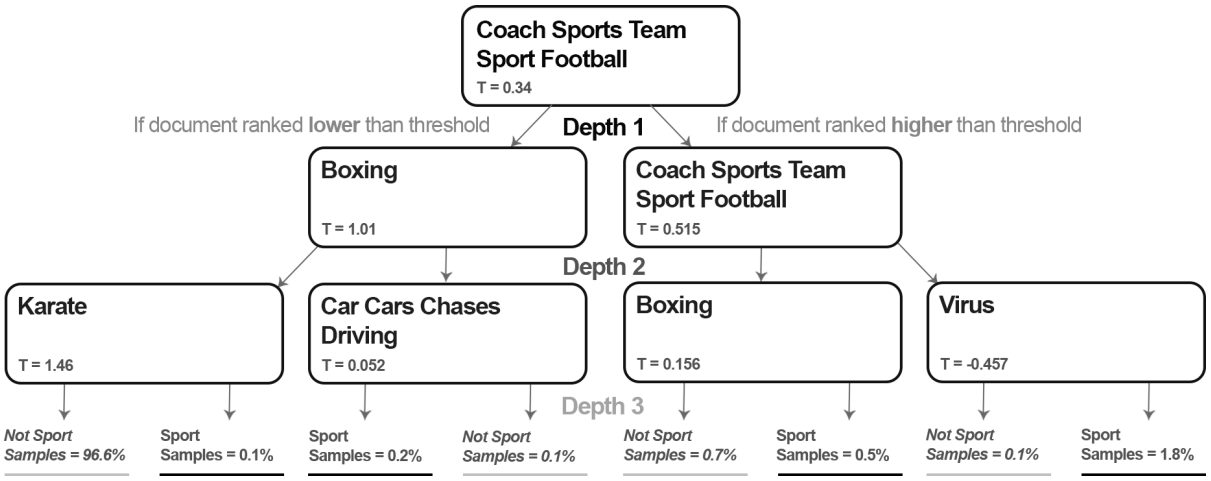

Figure 4.3: An example of a decision tree classifying if a movie is in the "Sports" genre. Each Decision Tree Node corresponds to a feature, and the threshold T is the required value of the feature for a document to traverse right down the tree instead of left. One interesting point to note is that the most important feature is used twice, the "coach, sports, team, sport, football" cluster and results in a majority of negative samples. Further, the nodes at depth three are more specific, sometimes overfitting (e.g. in the case of the "Virus" node, likely overfitting to a single movie about a virus) .

Classifier Parameters

There are two classifiers that are used across all experiments. CART Decision Trees[6] and linear SVM's [7].

Each SVM is also hyper-parameter tuned to find the best C values $(1.0, 0.01, 0.001, 0.000)$ and if the weights should be balanced such that positive instances are weighted in proportion to how rare they are[8]. In particular if weight balancing is used, where n is the number of documents, and y is the number of positive examples in the training data, the training sample is weighted equal to:

$$\frac{n}{2 * y} \tag{4.4}$$

Additionally, each Decision Tree is hyper-parameter tuned for the following parameters:

[6]Decision Tree implemented in scikit-learn

[7]Implemented in scikit-learn

[8]Weights implemented in scikit-learn

- The number of features[9] to consider when looking for the best split. $(all, auto, log2)$, where all means that all n features are considered, $auto$ means that the maximum features considered is $\sqrt{n}$, where n is the number of features, and $log2$ which means that the number of features considered is $log2(n)$

- The scoring criterion used to decide if the decision tree splits $(gini, entropy)$. i.e. if this score is higher after a node split, then that node is split. Where $gini$ is the gini impurity and $entropy$ is the information gain.[10]

- If the weights should be balanced.

Experiment One: Obtaining baselines

This first experiment is used to determine the best-performing document embedding models for performance when used as input to depth-one, depth-two, depth-three and unbounded decision trees, as well as a linear SVM. This provides a frame of reference for the later experiments in the case of the depth-limited trees and baseline performance on the task in the case of the linear SVM. In addition to the unsupervised document embedding models, two additional document models are included as reference: a bag-of-words of PPMI scores (BOW-PPMI) and a Latent Dirchlet Allocation (LDA) topic model. The BOW-PPMI is used as a reference for a standard performance baseline representation on the task, and the LDA topic model is used as a reference interpretable representation (where the information in the unstructured text is separated into topics). After the original filtering done to the vocabulary where either only the top 100,000 most frequent terms are used, or those that did not occur more than once are removed, the BOW-PPMI additionally has all terms filtered out that do not occur in at least 0.1% of documents. This is to limit the memory requirements of the representation, but also to ensure that noisy terms that can be overfit on are not included as features. To determine the best parameters for Doc2Vec, the following parameters are tuned:

- The $windowsize(5, 10, 15)$ referring to the context window of the words that used during training.

[9] "max_features" in scikit-learn

[10] "criterion" in scikit-learn

- The $mincount(1, 5, 10)$ referring to the minimum frequency of words included during training.
- The number of times the data is iterated on when training $epochs(50, 100, 200)$.

The best doc2vec embedding for each task is the one that performs best when used as input to a linear SVM. The idea is that if a linear SVM is able to predict the target categories well, then it should be possible to find meaningful directions in that space. Additionally, topic models are tuned. For the Latent Dirchlet Allocation topic models[11], the following parameters are tuned:

- The prior of the document topic distribution *(0.001, 0.01, 0.1)*.
- The prior of the topic word distribution *(0.001, 0.01, 0.1)*.

Experiment Two: The best single-term property features

This experiment shows the results for the single-term disentangled feature representation when used as input to low-depth decision trees. It may be the case that clustering directions does more harm to the feature representation than good, so these experiments are used to investigate the use of a single-term feature representation. This experiment also finds the best-performing parameters on the task for the single-directions. The end-result is that for each task, there will be a best performing document embedding model and corresponding number of dimensions of that model (e.g. Doc2Vec of size 50), and the following hyper-parameters are optimized for that model:

- The frequency cut-off of the terms $f(20000, 10000, 5000)$, where directions are only obtained for terms in the top f.
- The score cut-off of the directions $s(2000, 1000)$, where the feature-representation used as input to the classifiers is composed of rankings of entities on the top s directions.
- The score-type *t (F1, Kappa, Accuracy, NDCG)* which is the method used to score the directions.

[11] Topic models implemented in scikit-learn

The reason these hyper-parameters are optimized are because even the top-scoring directions are not always meaningful, as observed in the qualitative analysis in Table 4.4 for example.

Experiment Three: The best cluster features

In this experiment, results for the cluster-feature disentangled representation are obtained. Clustering the individual feature directions results in a lower number of linearly combined features labelled with clusters of words. As in the last experiment, the feature-representation composed of cluster-rankings is used as input to low-depth decision trees. The two different clustering methods, k-means[12] and Derrac's k-means variation (as described in Section 4.3.3), are used. The best single-term property parameters are used as the basis of this experiment, where for each classifier and task the clusters are obtained from the top s directions and score-type t, chosen by hyper-parameter optimization for that classifier and task. The only hyper-parameter optimized in this experiment is the amount of clusters.

4.5.3 Summary of all Results

Table 4.5.4 is a summary of the three experiments, listing results of different representations on trees limited to depth-one, two and three. As expected, the initial document embedding vectors did not perform well on this task. In contrast, single-directions and clusters of these single-directions obtained from these document embedding models out-perform the bag-of-words in most cases. One exception is the depth-two tree for the place-types domain, and another is the depth-one tree on the movies keywords task.

For the keywords task, in a depth-1 tree, finding words that directly correspond to particular keywords is simple with the BOW-PPMI representation. As the words that are able to classify these keywords will typically be infrequent (e.g. "new york city", "father-son relationship"), they are likely not as well-represented spatially in the document embedding. In this case, the PPMI representation is suitable, as it can find one-to-one matches with the classes without modelling similarity information. When using depth-two and depth-three limited decision trees, the score is no-longer as good for PPMI, which is likely due to overfitting.

[12]K-means implemented in scikit-learn

Sometimes decision trees of depth-two outperform those of depth one, but generally depth three trees perform best. In the case of the place-types, although topic models and PPMI representations are indeed the best, it is not by a wide-margin. Meanwhile when the single directions perform the best in these domains for other tree types they perform much better than the other approaches. Additionally, the number of entities in the place-type domain is the lowest among all domains and the tasks are very unbalanced, so it is possible that they overfit.

4.5.4 Initial Document Embedding Vectors

In Table 4.7 Decision Trees and SVM results are shown when the document embeddings, the baseline representation BOW-PPMI, and the topic model features are used as input. In the case of the topic model and doc2vec, these representations are tuned. These representations serve as a reference point for what is possible using standard linear models, as well as a measure of the disentanglement of the initial document embedding vectors. The initial document embedding vectors when used as input representations to depth three trees perform significantly worse for depth one trees. This is expected given the fact that these representations are entangled, so it is unlikely that a smaller tree can use the available dimensions to model a class. There are two tables, Table 4.7 that covers the results for the newsgroups in full detail (including more parameters, and the precision and recall scores), and Table 4.8 that shows results for all other domains but only shows the best dimensionality document. Note that in Table 4.7 some recall scores are very high. Recall scores are higher when the weights are balanced, as favouring positive instances results in more positives being identified correctly.

As seen in Table 4.7, the biggest differences in F1-score occur when the document embedding model changes, rather than when the dimensionality of document embedding model changes. For this reason, and to reserve space, only the best performing representation of each type are shown in Table 4.7.

In Table 4.7 for the newsgroups, the linear SVM only achieved strong performance when using doc2vec or PCA as input, and in the unbounded depth decision tree PCA performed far better than the other unsupervised document embeddings. Similarly for reuters and sentiment, the PCA representation performs better than the other unsupervised document embeddings. This suggests that the linear combinations used in the PCA representation result in a more linearly

	Genres			**Keywords**			**Ratings**		
Movies	D1	D2	D3	D1	D2	D3	D1	D2	D3
Space	0.301	0.358	0.354	0.185	0.198	0.201	0.463	0.475	0.486
Single directions	**0.436**	0.463	0.492	0.23	**0.233**	**0.224**	0.466	0.499	0.498
Clusters	0.431	**0.513**	**0.506**	0.215	0.22	0.219	**0.504**	**0.507**	**0.513**
PPMI	0.429	0.443	0.483	**0.243**	0.224	0.224	0.47	0.453	0.453
Topic	0.415	0.472	0.455	0.189	0.05	0.075	0.473	0.243	0.38
	Newsgroups			**Sentiment**			**Reuters**		
Rep	0.251	0.366	0.356	0.705	0.77	0.773	0.328	0.413	0.501
Single dir	0.418	**0.49**	**0.537**	0.784	0.814	**0.821**	**0.678**	**0.706**	0.72
Cluster	0.394	0.433	0.513	0.735	**0.844**	0.813	0.456	0.569	0.583
PPMI	0.33	0.407	0.444	0.7	0.719	0.73	0.616	0.699	**0.723**
Topic	**0.431**	0.423	0.444	**0.79**	0.791	0.811	0.411	0.527	0.536
	Foursquare			**OpenCYC**			**Geonames**		
Placetypes	D1	D2	D3	D1	D2	D3	D1	D2	D3
Rep	0.438	0.478	0.454	0.383	0.397	0.396	0.349	0.34	0.367
Single dir	**0.541**	0.498	**0.531**	0.404	**0.428**	0.39	**0.444**	**0.533**	**0.473**
Cluster	0.462	0.507	0.496	**0.413**	0.42	**0.429**	0.444	0.458	0.47
PPMI	0.473	**0.512**	0.491	0.371	0.351	0.352	0.361	0.301	0.242
Topic	0.488	0.433	0.526	0.365	0.271	0.313	0.365	0.3	0.219

Table 4.6: The F1-scores of trees limited to depth-one, two and three, for the best hyper-parameter variation of each representation

separable space. The performance gap only widens when looking at low-depth decision trees that use the PCA dimensions as input, which suggests that the individual dimensions of the other initial document embedding vectors are less informative than those of the PCA spaces. This is likely due to the fact that PCA tries to select the most informative dimensions, whereas the dimensions of the other representations are more or less arbitrary. However, this does not tell us anything about the quality of the directions that can be learned in both types of spaces.

In the single directions results, PCA is outperformed by MDS and other representations in F1 score for low Decision Tree depths in any of these domains, with the exception of the depth-two trees for sentiment. Despite MDS not having informative dimensions, our method is still able to obtain a distangled feature representation from this document embedding. There is a low correlation between performance on the raw dimensions of the space and performance with rankings on directions in low-depth Decision Trees. Despite the information encoded in the space, if it is not disentangled then the classifier will not perform well.

4.5.5 Single Term Features

In Table 4.9 the results for the feature representation composed of rankings of entities on single-term directions that performed best when used as input to a classifier is shown.

As can be seen for the newsgroups results in Table 4.9, the number of dimensions now has a bigger impact on the results, in particular for the depth-three and two trees. This difference was smaller for the depth-one trees. This is likely because despite the size of the document embedding model, they all modelled a similar most-important single direction relevant to the task, e.g. a direction that can classify if the entity is religious like the word-direction for "celestial". However, variants of the space size change the representation of auxiliary directions that are not as relevant to the task but are useful when the depth is higher for the tree.

The best space type also varied across domains, e.g. the classifiers performed best for reuters with a disentangled single-term feature representation obtained from MDS used as input, but a feature representation obtained from Doc2Vec when used as input performed best for newsgroups. Loosely, it is possible to attribute the performance increase for a space-type to an increased score (e.g. NDCG score) for relevant directions to the class. When looking at the qualitative results, generally the words common to all space-types are the most salient, as can

Newsgroups	D1				D2				D3				DN				SVM			
	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec
PCA 200	0.701	0.251	0.148	0.811	0.843	0.366	0.245	0.719	0.956	0.355	0.54	0.265	0.946	0.44	0.45	0.432	0.969	0.612	0.746	0.519
PCA 100	0.698	0.247	0.146	0.813	0.835	0.362	0.241	0.731	0.957	0.356	0.576	0.257	0.948	0.451	0.465	0.438	0.969	0.586	0.768	0.474
PCA 50	0.68	0.24	0.141	0.829	0.834	0.355	0.234	0.735	0.957	0.329	0.472	0.253	0.947	0.45	0.462	0.438	0.966	0.52	0.745	0.399
AWV 200	0.687	0.217	0.126	0.781	0.758	0.256	0.156	0.718	0.764	0.26	0.157	0.751	0.937	0.339	0.352	0.328	0.961	0.468	0.641	0.369
AWV 100	0.677	0.21	0.122	0.775	0.78	0.275	0.173	0.683	0.746	0.25	0.149	0.769	0.934	0.324	0.332	0.317	0.865	0.4	0.265	0.812
AWV 50	0.696	0.219	0.127	0.772	0.777	0.272	0.168	0.71	0.743	0.25	0.149	0.786	0.935	0.325	0.335	0.316	0.842	0.362	0.233	0.819
MDS 200	0.581	0.184	0.103	**0.837**	0.742	0.262	0.16	0.729	0.719	0.236	0.139	0.785	0.935	0.327	0.332	0.323	0.965	0.501	**0.802**	0.364
MDS 100	0.586	0.187	0.105	0.833	0.754	0.261	0.159	0.727	0.705	0.236	0.138	**0.808**	0.935	0.33	0.338	0.321	0.878	0.439	0.308	0.765
MDS 50	0.593	0.153	0.087	0.647	0.716	0.25	0.15	**0.756**	0.736	0.243	0.144	0.774	0.935	0.324	0.335	0.313	0.854	0.394	0.259	0.821
D2V 200	0.682	0.205	0.119	0.746	0.802	0.268	0.169	0.646	0.77	0.269	0.164	0.75	0.94	0.366	0.389	0.346	0.961	0.468	0.641	0.369
D2V 100	0.682	0.208	0.12	0.762	0.792	0.268	0.168	0.662	0.786	0.268	0.164	0.727	0.94	0.376	0.392	0.361	**0.971**	**0.628**	0.761	0.535
D2V 50	0.683	0.207	0.12	0.764	0.809	0.294	0.187	0.694	0.782	0.28	0.172	0.761	0.943	0.394	0.415	0.376	0.97	0.601	0.758	0.497
PPMI	**0.948**	0.33	**0.532**	0.239	0.947	0.407	0.511	0.338	0.944	**0.444**	0.506	0.396	**0.951**	**0.494**	**0.496**	**0.492**	0.962	0.613	0.627	0.599
Topic	0.852	**0.431**	0.304	0.743	**0.96**	**0.423**	**0.604**	0.326	**0.961**	0.444	**0.606**	0.35	0.944	0.432	0.434	0.429	0.879	0.46	0.318	**0.835**

Table 4.7: Best results for the newsgroups for a variety of classifiers when using the document embeddings as input.

Reuters	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	0.847	0.328	0.917	0.413	0.978	0.501	0.978	0.565	0.989	0.761
AWV	0.782	0.252	0.971	0.328	0.974	0.417	0.973	0.495	0.987	0.719
MDS	0.791	0.263	0.9	0.357	0.979	0.489	0.976	0.522	0.988	0.67
D2V	0.818	0.268	0.867	0.298	0.974	0.445	0.971	0.482	0.986	0.724
PPMI	**0.975**	**0.616**	**0.978**	**0.699**	**0.98**	**0.723**	**0.984**	**0.746**	**0.99**	**0.8**
Topic	0.92	0.411	0.977	0.527	0.977	0.536	0.977	0.56	0.95	0.513

Sentiment	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	0.745	0.705	0.755	0.77	0.778	0.773	**0.781**	**0.779**	**0.891**	**0.893**
AWV	0.642	0.652	0.643	0.694	0.695	0.717	0.66	0.663	0.827	0.829
D2V	0.642	0.664	0.66	0.707	0.702	0.7	0.711	0.708	0.878	0.878
PPMI	0.616	0.7	0.655	0.719	0.675	0.73	0.712	0.71	0.887	0.888
Topic	**0.793**	**0.79**	**0.794**	**0.791**	**0.81**	**0.811**	0.733	0.73	0.815	0.822

Placetypes OpenCYC	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	0.586	0.346	0.708	0.343	0.695	0.342	0.832	0.309	0.847	0.474
AWV	0.625	**0.383**	0.651	0.376	0.728	**0.396**	**0.844**	**0.362**	0.85	0.466
MDS	0.624	0.364	0.7	**0.397**	0.731	0.374	0.843	0.305	0.861	**0.476**
PPMI	**0.728**	0.371	0.75	0.351	0.739	0.352	0.843	0.323	**0.9**	0.366
Topic	0.708	0.365	**0.87**	0.271	**0.87**	0.313	0.831	0.313	0.808	0.407

Movies Genres	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	0.722	0.301	0.755	0.339	0.717	0.321	0.884	0.372	**0.925**	0.518
AWV	0.679	0.29	0.774	0.321	0.756	0.343	0.873	0.312	0.922	0.496
MDS	0.679	0.298	0.79	0.358	0.773	0.354	0.887	0.385	0.875	**0.532**
PPMI	**0.852**	**0.429**	**0.91**	0.443	**0.912**	**0.483**	0.882	**0.416**	0.923	0.526
Topic	0.767	0.415	0.905	**0.472**	0.912	0.455	**0.889**	0.415	0.843	0.491

Placetypes Foursquare	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	0.731	0.342	0.823	0.393	0.86	0.388	0.887	0.398	0.896	0.568
AWV	0.767	0.401	0.828	0.478	0.85	0.452	0.905	**0.505**	0.923	**0.622**
MDS	**0.915**	0.438	0.804	0.427	0.86	0.454	0.893	0.462	0.932	0.619
PPMI	0.889	0.473	0.915	**0.512**	0.904	0.491	0.881	0.31	**0.938**	0.567
Topic	0.864	**0.488**	**0.916**	0.433	**0.917**	**0.526**	**0.907**	0.464	0.916	0.569

Movies Keywords	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	0.647	0.185	0.644	0.193	0.677	0.199	0.846	0.161	0.787	0.272
AWV	0.5	0.16	0.641	0.179	0.595	0.174	0.853	0.141	0.717	0.23
MDS	0.633	0.179	0.69	0.198	0.674	0.201	0.84	0.163	0.788	**0.28**
PPMI	**0.818**	**0.243**	0.745	**0.224**	0.739	**0.224**	0.847	**0.17**	**0.921**	0.217
Topic	0.629	0.189	**0.932**	0.05	**0.93**	0.075	**0.857**	0.152	0.678	0.21

Placetypes Geonames	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	0.502	0.301	0.69	0.305	0.68	0.295	0.821	0.243	0.844	0.401
AWV	0.657	0.326	0.755	0.323	0.842	**0.367**	0.813	0.332	0.865	**0.514**
MDS	0.626	0.349	0.695	**0.34**	0.796	0.272	**0.845**	0.295	0.638	0.397
PPMI	**0.808**	0.361	0.732	0.301	0.76	0.242	0.83	0.283	**0.894**	0.312
Topic	0.771	**0.365**	**0.863**	0.3	**0.85**	0.219	0.828	**0.348**	0.819	0.349

Movies Ratings	D1		D2		D3		DN		SVM	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
PCA	**0.65**	0.463	0.681	**0.475**	0.684	**0.486**	0.744	0.408	0.771	0.58
AWV	0.601	0.423	0.618	0.433	0.596	0.448	0.736	0.372	0.73	0.532
MDS	0.592	0.437	0.635	0.449	0.631	0.452	**0.752**	**0.412**	0.773	**0.589**
PPMI	0.583	0.47	0.635	0.453	0.605	0.453	0.73	0.384	**0.825**	0.536
Topic	0.575	**0.473**	**0.789**	0.243	**0.789**	0.38	0.739	0.375	0.704	0.501

Table 4.8: Best results for all domains for a variety of classifiers when using the document embeddings as input.

be seen in Table 4.3, so in this case the increased results for different hyper-parameters likely comes down to either these common directions being better modelled in the representation or terms that were not modelled well before that were relevant to the task being modelled better.

We see that generally, the best document embedding model is the same across the same domain: specifically AWV is the best for the place-types but MDS is best for the movies (despite a marginal difference in the ratings). Generally, this means that these document embedding types in particular are good at modelling informative properties for the tasks.

Although some tasks on some classifiers did perform better using score-types like F1-score, in general, filtering directions by NDCG score performed the best. In particular it was found the best score-type for Sentiment, Newsgroups, Reuters, Movies Genres, Movies Keywords in depth-3 Decision Trees. In conclusion, the document embedding model is an important hyper-parameter to tune for each domain, as is the size of that document embedding. However, document embeddings that perform well on one task may be good at the others too. Finally, NDCG is a good scoring metric to use for finding features that are meaningful in the representation.

4.5.6 Clustered Directions

Clustering has two main goals: the first is that features that align with properties of entities in the domain are obtained by combining together single-word directions that reference phenomena, e.g. in a domain of movie reviews "Blood", "Scary", "Horror" can be clustered to obtain a cluster that better models the idea of horror in film, rather than modelling individual things that occur in film. The second is that it makes the features more interpretable by providing context to the words. In this section, we examine how these more abstract and complex cluster features perform quantitatively at key domain tasks.

K-means mostly outperforms Derrac. It does not in the case of Keywords, where Derrac performs better for every Decision Tree. Although the differences in absolute values are quite small in this case, it is still significant as it is quite difficult to achieve high performance on this task, making these relative changes important. This case can give us insight into how disentanglement affects performance on different classes and domains - and how our unsupervised method selects the best parameters. Specifically, when looking into the how individual classes fared

Newsgroups	D1				D2				D3			
	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec
PCA 200	0.955	0.348	0.521	0.261	0.959	0.424	0.678	0.309	0.96	0.454	0.674	0.343
PCA 100	0.957	0.382	0.491	0.313	0.961	0.474	0.679	0.364	**0.963**	0.512	0.694	0.406
PCA 50	0.957	0.373	0.417	0.337	**0.963**	0.478	0.621	0.388	0.963	0.506	0.7	0.396
AWV 200	0.832	0.35	0.226	0.777	0.957	0.383	0.517	0.305	0.958	0.445	0.598	0.354
AWV 100	0.83	0.343	0.219	0.785	0.823	0.36	0.233	**0.792**	0.956	0.387	0.563	0.295
AWV 50	0.807	0.341	0.215	0.816	0.833	0.361	0.236	0.762	0.954	0.392	0.511	0.318
MDS 200	**0.959**	**0.418**	**0.543**	0.339	0.962	0.465	0.669	0.357	0.962	0.493	**0.707**	0.379
MDS 100	0.857	0.365	0.244	0.725	0.959	0.428	0.624	0.326	0.96	0.453	0.644	0.349
MDS 50	0.821	0.324	0.206	0.762	0.842	0.386	0.258	0.77	0.957	0.398	0.596	0.299
D2V 200	0.831	0.343	0.22	0.784	0.96	0.47	**0.683**	0.358	0.962	0.494	0.69	0.385
D2V 100	0.844	0.374	0.243	0.803	0.961	**0.49**	0.642	0.396	0.962	0.517	0.67	0.421
D2V 50	0.845	0.388	0.252	**0.844**	0.962	0.488	0.639	0.395	0.963	**0.537**	0.673	**0.446**

Reuters	D1		D2		D3		**Sentiment**	D1		D2		D3	
	ACC	F1	ACC	F1	ACC	F1		ACC	F1	ACC	F1	ACC	F1
PCA	0.976	0.658	0.979	0.679	0.977	0.467	PCA	0.739	0.759	**0.797**	**0.814**	0.802	0.805
AWV	0.975	0.598	0.979	0.656	0.98	0.66	AWV	0.7	0.699	0.711	0.736	0.723	0.735
MDS	0.975	**0.678**	**0.98**	**0.706**	**0.982**	**0.72**	D2V	**0.776**	**0.784**	0.782	0.801	**0.822**	**0.821**
D2V	**0.977**	0.583	0.979	0.664	0.98	0.632							
Placetypes	D1		D2		D3		**Movies**	D1		D2		D3	
OpenCYC	ACC	F1	ACC	F1	ACC	F1	**Genres**	ACC	F1	ACC	F1	ACC	F1
PCA	0.632	0.371	0.704	0.381	0.735	0.365	PCA	0.824	0.412	0.82	0.441	0.913	0.463
AWV	**0.66**	**0.404**	**0.734**	**0.428**	**0.755**	**0.39**	AWV	0.81	0.421	0.837	0.436	0.912	0.457
MDS	0.658	0.374	0.711	0.385	0.746	0.35	MDS	**0.849**	**0.446**	**0.839**	**0.463**	**0.918**	**0.495**
Foursquare	ACC	F1	ACC	F1	ACC	F1	**Keywords**	ACC	F1	ACC	F1	ACC	F1
PCA	0.785	0.477	**0.907**	0.474	0.869	**0.531**	PCA	0.737	0.225	0.727	0.227	**0.709**	0.22
AWV	**0.918**	**0.541**	0.881	**0.498**	0.889	0.466	AWV	0.656	0.201	0.672	0.203	0.652	0.2
MDS	0.82	0.416	0.879	0.482	**0.897**	0.485	MDS	**0.745**	**0.23**	**0.74**	**0.233**	0.708	**0.224**
Geonames	ACC	F1	ACC	F1	ACC	F1	**Ratings**	ACC	F1	ACC	F1	ACC	F1
PCA	0.665	0.348	0.754	0.342	0.743	0.306	PCA	**0.647**	**0.466**	**0.721**	**0.499**	0.681	0.492
AWV	**0.711**	**0.444**	**0.795**	**0.533**	**0.802**	**0.473**	AWV	0.646	0.463	0.692	0.474	0.677	0.483
MDS	0.591	0.289	0.772	0.333	0.764	0.352	MDS	0.62	0.463	0.692	0.489	**0.686**	**0.498**

Table 4.9: The best results when the single-term disentangled feature representation is used as input to low-depth decision trees

when using 100 clusters, the Derrac variant clusters performed better at the keywords "shot-in-the-chest" and "machine-gun" and sacrificed performance in the "sequel" class. With the Derrac variant of k-means, there was the following cluster ("soldiers combat fighting military battle ... weapons rambo gunfights spaghetti guns ...") while for the results of the k-means clustering method when restricted to 200 clusters these words were split into two separate clusters, one for guns ("gun explosions shoot shooting weapons ... rambo") and one for military ("war soldiers combat military ... platoon infantry"). It is possible that as the Derrac method combined these together into a single cluster it was able to better capture the classes for "shot-in-the-chest" and "machine-guns" as these keywords refer to war films where people were shot or shooting. So in this case, the clusters chosen by the Derrac variant supported the classification of the documents.

This idea is supported when looking at the depth-three tree for this class, which uses this cluster as its first node as well as a node in the depth-two layer. This is an instance where averaging the direction of a heavily populated cluster performs better than obtaining features that are more dense and less disentangled.

Meanwhile, this same lack of separation caused the clusters to lose performance in the "sequel" class. With the k-means method, the cluster was found for ("franchise sequels sequel installments") with the Derrac variant, the corresponding cluster was ("franchise sequels sequel instalments entry returns"). This cluster was also chosen for the Derrac variant as the first node of its Decision Tree, but this caused it to perform worse than k-means. This is likely because although the words "entry" and "returns" were most similar to this cluster, they disrupted the direction too much. Indeed, when looking at the k-means clusters, the "returns" direction is clustered with "events situation conclusion spoiler ... protagonists exscapes break scenario ...", seemingly referring to a character or thing "returning" in a conclusive part of the movie, and the word "entry" is clustered with the words "effective genuine ... hits build surprisingly ... succeeds essentially finale entry ..." seemingly relating to a more sentiment related cluster about how a movie performed. So in this case k-means found more disentangled clusters than Derrac gave it a performance advantage.

This could be due to the best-performing variants of the Derrac clustering method obtaining 100 clusters (meaning the clusters would contain more terms) and the best performing variants of the k-means obtaining 200 clusters. However, in the 100-size k-means clusters, "gun" and "explosions" ended up being in a cluster with ("western outlaw heist shootout west"), making

it a more western oriented cluster, and the idea of a war was even more separated with a single cluster corresponding to ("war soldiers military solider army sergeant sgt platoon infantry"). In conclusion, Derrac for the Keywords task captured certain properties better than k-means, in particular by clustering together the idea of "war" and "guns" to achieve high performance on the keywords "shot-in-the-chest" and "machine-guns". k-means favoured a more separated approach to these ideas, which meant that although it captured the idea of "war" well, it was not able to capture the classes inbetween the idea of "war" and "guns".

4.5.7 Conclusion

In conclusion, this chapter introduced a methodology to go from an initial document embedding and an associated bag-of-words to a disentangled feature representation, and these representations are shown to be disentangled by enabling low-depth decision trees to perform well. The features of this disentangled representation are labelled, potentially enabling use as input to simple interpretable classifiers e.g. decision trees, as its components correspond to semantic features. This methodology could be used as an alternative to methods like Topic Models, and give insight into the parameters required and qualitative results that can be obtained. This method can be applied to many vector spaces and domains as a post-processing step, and is extensively tested qualitatively and quantitatively, in particular qualitatively finding that the variants which perform best in the quantitative evaluation also have semantic features that make good intuitive sense. We find that our method greatly outperforms the original representations on low-depth Decision Trees, giving good evidence that the representation is disentangled i.e. strongly suggesting that the features that are obtained with this method are indeed semantic. The results of these simple classifiers are competitive with topic models and a bag-of-words of PPMI values in most cases. In this chapter, we introduced several variants to the original model from Derrac [19], and these variants are found to generally perform well, in particular the NDCG scoring method performs better than Kappa in most cases, and that standard K-means performs better than the original clustering method. A variety of space-types and domains were experimented with, verifying that the methodology can be applied more generally than shown in [19]. The main experiments that would be interesting to expand on for this chapter would be more recent state-of-the-art representations, specific investigations of how those representations are able to achieve such strong results, and interpretability experiments to see how the cluster labels fare in

real-world situations.

Newsgroups	D1				D2				D3			
	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec	ACC	F1	Prec	Rec
K-means 200	**0.852**	**0.394**	**0.261**	0.795	**0.958**	**0.433**	**0.58**	0.345	**0.963**	**0.513**	**0.704**	0.403
K-means 100	0.842	0.388	0.257	0.791	0.958	0.366	0.516	0.284	0.962	0.5	0.635	**0.412**
K-means 50	0.834	0.381	0.248	**0.819**	0.815	0.336	0.212	**0.81**	0.961	0.485	0.612	0.402
Derrac 200	0.803	0.313	0.202	0.693	0.797	0.306	0.191	0.781	0.958	0.409	0.605	0.309
Derrac 100	0.792	0.305	0.197	0.667	0.791	0.287	0.179	0.721	0.957	0.374	0.56	0.281
Derrac 50	0.769	0.26	0.162	0.661	0.768	0.237	0.143	0.693	0.955	0.315	0.47	0.237

Table 4.10: All clustering size results for the newsgroups

Reuters	D1		D2		D3		**Sentiment**	D1		D2		D3	
	ACC	F1	ACC	F1	ACC	F1		ACC	F1	ACC	F1	ACC	F1
K-means	**0.875**	**0.338**	**0.975**	**0.54**	0.973	**0.58**	K-means	0.623	0.674	**0.837**	**0.844**	0.658	0.707
Derrac	0.797	0.291	0.973	0.402	**0.974**	0.485	Derrac	**0.712**	**0.735**	0.802	0.82	**0.803**	**0.813**
Placetypes	D1		D2		D3		Movies	D1		D2		D3	
OpenCYC	ACC	F1	ACC	F1	ACC	F1	Genres	ACC	F1	ACC	F1	ACC	F1
K-means	**0.641**	**0.413**	**0.735**	**0.405**	0.75	**0.43**	K-means	**0.813**	**0.431**	**0.913**	**0.513**	**0.913**	**0.506**
Derrac	0.605	0.39	0.672	0.392	**0.755**	0.391	Derrac	0.759	0.341	0.789	0.431	0.911	0.432
Foursquare	ACC	F1	ACC	F1	ACC	F1	Keywords	ACC	F1	ACC	F1	ACC	F1
K-means	**0.913**	**0.462**	**0.911**	**0.5**	**0.891**	**0.511**	K-means	0.667	0.208	0.648	0.202	0.678	0.213
Derrac	0.768	0.392	0.835	0.445	0.805	0.425	Derrac	**0.726**	**0.215**	**0.745**	**0.22**	**0.707**	**0.219**
Geonames	ACC	F1	ACC	F1	ACC	F1	Ratings	ACC	F1	ACC	F1	ACC	F1
K-means	**0.772**	0.43	**0.774**	0.407	**0.819**	**0.472**	K-means	**0.671**	**0.504**	0.638	**0.507**	**0.686**	**0.513**
Derrac	0.678	**0.449**	0.74	**0.411**	0.807	0.415	Derrac	0.651	0.445	**0.669**	0.463	0.627	0.479

Table 4.11: The best results when using the entities ranked on the directions output by the clustering method for the optimal number of clusters for each domain and task..

Chapter 5

Disentangling neural network layers

5.1 Introduction

The previous chapter showed how in a vector space of entities (e.g. different movies) fine-grained semantic relationships can be identified with directions (e.g. more violent than) and used as features in a document classification task. This chapter explores how these fine-grained semantic relationships can be used to gain insights into the characteristics of learned neural network models. Two kinds of neural networks are investigated, feed-forward networks and auto-encoders. The hidden layers of these networks are viewed as vector spaces, and are re-organized into interpretable feature representations using the method in Chapter 4. Additionally, the use of these semantic features derived from neural network embeddings as disentangled feature representations is qualitatively and quantitatively investigated. In the case of the feed-forward networks, these feature representations are quantitatively verified to be meaningful using low-depth decision trees, and are qualitatively investigated using examples. It is found that semantic features can indeed be obtained from these neural network representations, and that the rankings of entities on these feature-directions make sense. Interestingly, the results of the low-depth decision tree obtained for the 20-newsgroups that uses a disentangled feature representation of cluster-features as input even outperforms the original neural network.

Auto-encoders are also investigated by obtaining semantic features from their layers, and their use is investigated in the context of learning about relationships between properties in the domain. Specifically, auto-encoders are used to obtain a sequence of increasingly low-dimensional entity embeddings that are expected to model increasingly abstract relationships. Features from each auto-encoder vector space are then used to induce symbolic rules that relate specific properties to more general ones. Illustrative examples are provided of the hidden layers and these

rules. As an example, below is one of the rules derived with this method, where the first two terms are from one entity embedding, and the final term is from a more abstract entity embedding.

```
IF Emotions AND Journey THEN Adventure          (5.1)
```

This method generally has two goals, the first is to qualitatively investigate if neural networks can help identify relationships between features, and the second is to investigate the kind of features encoded in auto-encoders. One potential application of this work in recommendations is in the robustness of explanations using these features. In the domain of movies, we may have a situation where the synopsis or reviews mention the words “Emotions" and “Journey", from which the system could derive that it is probably an “Adventure" movie and use that term as a part of its supporting explanation e.g. "You liked how this movie was an emotional and a journey, so you may like this adventure movie." Of-course if these features are indeed salient and can be linked in this way, these features are also be useful for investigating how neural networks represent information.

In conclusion, feed-forward networks and auto-encoders are investigated using the methods described in Chapter 4. In Section 5.2 some neural-network specific methods for interpretability are discussed. Following that, in Section 5.3 the detail of how the method from Chapter 4 is used to investigate neural networks is made clear, as well as how rules are induced that explain how semantic features relate to each other across the different layers of an auto-encoder. Next in Section 5.4 the results of the qualitative and quantitative investigation of the feed-forward networks is discussed followed by the qualitative results for the auto-encoders. Specifically, it is found that the semantic features derived from neural network embeddings are meaningful, low-depth decision trees perform well when using these semantic features as input, and various characteristics of neural network embeddings and their associated semantic features are observed in a qualitative analysis. Finally in 5.6 conclusions are drawn as to the usefulness of the method, the results, their limitations and how the work in Chapter 5 solves these limitations.

5.2 Background

Neural network rule extraction algorithms can be categorized as either decompositional, pedagogical or eclectic [3]. Decompositional approaches derive rules by analysing the units of the network, while pedagogical approaches treat the network as a black box, and examine the global relationships between inputs and outputs. Eclectic approaches use elements of both decompositional and pedagogical approaches. Our method could be classified as decompositional, as we re-organize the hidden layer of the neural network. We will now describe some similar approaches and explain how our methods differs.

The algorithm in [36] is a decompositional approach that applies to a neural network with two hidden layers. It uses hyperplanes based on the weight parameters of the first layer, and then combines them into a decision tree. NeuroLinear [65] is a decompositional approach applied to a neural network with a single hidden layer that discretizes hidden unit activation values and uses a hyperplane rule to represent the relationship between the discretized values and the first layer's weights. HYPINV [62] is a pedagogical approach that calculates changes to the input of the network to find hyperplane rules that explain how the network functions.

The main difference in our work is that our method induces rules from properties derived from the re-organized layers of a network, rather than learning rules that describe the relationships between units in the network itself. Additionally, the focus is on a qualitative investigation of the network and its potential to learn increasingly general entity embeddings from hidden representations rather than tuning network parameters such that weights directly relate to good rules.

There are many methods for obtaining an explanation of a black-box classifier after it has been learned. One typical approach is to learn a 'proxy model', a more interpretable classifier that approximates the decisions made by the network. This is a pedagogical approach, that focuses on mapping inputs to outputs. The most popular approach is LIME which produces a linear model to explain a single prediction in terms of the input features [59]. By approximating many difference instances, it can thus give users an impression of how the model is behaving, despite these local predictions sometimes being contradictory in how features are used [58]. LIME differs from the work in this thesis as it focuses on the relationship between input and output, rather than explaining the internal layers of the network. Essentially, the work on this thesis

focuses on layers, while their work focuses on explaining a black-box in terms of its input.

Another method that can be used to explain a neural network is DeepRed. DeepRed explains a neural network by creating a proxy model decision tree that is faithful to the original network [80]. However, in practice when producing a decision tree that approximates a neural network with multiple layers, the tree can be large, despite pruning methods. Although this method may seem similar on the surface, as this work uses decision trees, it differs from the work here as the goal of this work is not to be faithful to the predictions of the neural network, but rather to investigate the representation in a neural network's hidden layer by re-organizing it into a disentangled feature representation. The decision tree in this work is used as a simple model that can verify the interpretable feature representaiton is disentangled. Hypothetically, a variety of other simple models could be used.

Another recent topic that relates to our work is improving neural networks and entity embeddings using symbolic rules [29]. In [60] a combination of first-order logic formulae and matrix factorization is used to capture semantic relationships between concepts that were not in the original text. This results in relations that are able to generalize well from input data.

Section 5.5 considers the opposite task: using semantic features derived from neural network embeddings to learn better rules. The rules that are derived are not intended to explain how the network functions but rather to attempt to describe the semantic relationships that hold in the considered domain. In other words, the aim of the work on auto-encoders to learn rules in this Chapter is to investigate the use of neural network representations in the hidden layer as a tool for learning logical domain theories, where the focus is on producing rules that capture meaningful semantic relationships.

5.3 Method

5.3.1 Feed-forward networks

We train two neural networks with non-linear activation functions. The first is NNET-U, a neural network with a single hidden layer that uses a pre-trained document embedding as input. The second is NNET-B, a neural network that starts with the PPMI BOW as input. The output

layer is the of size C_n where C_n is the number of classes. Each network is trained on one task from each domain. As the goal is a qualitative analysis, these tasks were chosen as they have a clear semantic meaning in the domain. In particular, for the movies the "Genres" task was chosen, and for the place-types the "Foursquare" task was chosen as its classes are the easier to understand without expert knowledge. The newsgroups domain is also included in the results for comparison. To obtain a representation from this neural network, the activation values from a layer of the trained neural network are treated as the coordinates of a vector space.

The representations derived from these networks should differ from the neural network as it is trained on a multi-label classification problem, meaning that all of the classification objectives are predicted in the output layer simultaneously. In the case of a binary task like sentiment (where only one class is classified and it is either 1 or 0), the neural network may filter out a large amount of information if it is performing well, as a large portion of it is irrelevant to the task. However, as multi-label objectives like those of the newsgroups or genres tasks are used, it can be expected that a large portion of information will be retained as much of the data is highly relevant to these classes.

Following results showing that simple neural networks can still perform well on text classification without needing a complex architecture [40] [53], in this work, the following parameters were recommended for the feed-forward network: Cross-entropy loss-function, Adagrad trainer with default parameters, Dropout, Sigmoid activation on the output layer, and Relu hidden activation units. However, in this work the tanh activation function is used instead of Relu, as the binary cut-off did not result in high-quality directions, and did not improve performance remarkably in early tests. The reason tanh was chosen is because in initial tests the directions were meaningful. Alternative activation functions like sigmoid were not tested.

NNET-B, the network that uses a bag-of-words as input has two hidden-layers, the first has 1,000 nodes and the second has 100. The reason multiple layers is used is because in preliminary experiments, the network with two layers outperformed the single layer network consistently, and in the case of newsgroups outperformed the top-performance linear SVM used an unsupervised document embedding as input. The representation that the interpretable feature representation is derived from is the output of the hidden layer of size 100 in the learned network. For the qualitative examples derived from this network, the final hidden layer of size 100 was used.

To learn NNET-U, the input is the unsupervised document embedding associated with the single-term disentangled feature representation that performed highest in F1-score when used as input to a decision tree limited to a depth of 3, found in Table 4.5.4. For example, in the newsgroups domain, this is the Doc2Vec embedding size 100 space. The reason this space is used is because it is assumed that a representation that performs well on decision trees limited to a depth of 3 contains semantic features that can be meaningfully adjusted by the network. Only a single hidden layer is used, as there was not a significant performance difference in initial experiments when using multiple hidden layers. The amount of nodes in the hidden-layer is set to be the same size as the input representation. This enables easy comparison between the initial unsupervised embedding and the embedding derived from the neural network.

5.3.2 Inducing Rules from Auto-Encoder Entity Embeddings

In this section, the method to obtain a series of increasingly general entity embeddings is explained, as well as the methodology of how symbolic rules can be learned that link properties from subsequent spaces together.

For this method with auto-encoders, the MDS embedding on the movies dataset is used, as it performed the best on low-depth Decision Trees in Chapter 4 and its features were used in previous qualitative analysis and found to be intuitive. To construct more general embeddings from the initial embedding provided by the MDS method, stacked denoising auto-encoders [76] are used. Standard auto-encoders are composed of an “encoder" that maps the input representation into a hidden layer, and a “decoder" that aims to recreate the input from the hidden layer. Auto-encoders are normally trained using an objective function that minimizes information loss (e.g. Mean Squared Error) between the input and output layer [6]. The task of recreating the input is made non-trivial by constraining the size of the hidden layer to be smaller than the input layer, forcing the information to be represented using fewer dimensions, or in denoising auto-encoders by corrupting the input with random noise, forcing the auto-encoder to use more general commonalities between the input features. By repeatedly using the hidden layer as input to another auto-encoder, we can obtain increasingly general representations. To obtain the entity representations from our auto-encoders, we use the activations of the neurons in a hidden layer as the coordinates of entities in a new vector space.

One interesting novelty of the approach is that it introduces a method to characterize semantic features (i.e. clusters of directions) modelled in one space in terms of salient properties that are modelled in another space. To do so, the off-the-shelf rule learner JRip (Using the "RIPPER" algorithm [16]) is used to predict which entities will be highly ranked, according to a given cluster direction, using the rankings induced by the clusters of the preceding space as features. To improve the readability of the resulting rules, rather than using the precise ranks as input, the ranks are aggregated by percentile, i.e $1\%, 2\%, ..., 100\%$, where an entity has a 1% label if it is among the 1% highest ranked entities, for a given cluster direction. For the class labels, a movie is defined as a positive instance if it is among the highest ranked entities (e.g. top 2%) of the considered cluster direction. Using the input features of each layer and the class labels from the subsequent layer, these rules can be used to explain the semantic relationships between properties modelled by different vector spaces. One drawback of discretizing continuous attributes is that the accuracy of the rules extracted from the network may decrease [64]. However, in our setting, understanding what is meant by the rules is more important than accuracy, as these rules are not intended for making predictions, but only for getting insight into data and potentially generating explanations of domain knowledge.

5.4 Investigating Feed-forward Neural Networks

5.4.1 Parameters

Three different domains are investigated: Newsgroups, Placetypes and Movie reviews. These domains are chosen as their associated classification tasks are relevant to a large number of documents, meaning that the network must learn general domain concepts, e.g. in order to classify the 20-newsgroups a variety of information is required. Although the place-types domain is included, due to the low number of entities a high accuracy score is not expected. This domain is included because their entities are easy to understand in qualitative examples. In fact, as the positive instances for the Foursquare task, the task chosen, has such a low number of positive instances it would be unreasonable to expect good results with neural networks which typically perform well with a large number of entities and a more balanced task.

As in Chapter 3 4.3.1 the words used when obtaining candidate directions are limited by a fre-

quency cut-off. In prior experiments, this was used as a hyper-parameter. However, in this chapter the frequency threshold is set for all experiments so that only the top 10,000 most frequent words are considered. Additionally, the cut-off to decide how many top-scoring directions are used as input to the clustering algorithm and decision trees is set to the top 1,000 highest-scoring directions. In the previous experiment, both of these cut-offs were tuned for the specific space-type and task in-order to achieve stronger results. The reason that these parameters are not tuned for this experiment is because the main focus of these experiments is a qualitative investigation of the directions, rather than attempting to maximize disentaglement or performance on the text classification task. Essentially, standardizing these cut-offs makes it easier to come to conclusions when comparing between them qualitatively, despite potential performance losses. Similarly, as Normalized Discounted Cumulative Gain (See Section 4.3.2) was shown to perform well for a variety of embedding types and domains, it was the only scoring method used.

The network is hyper-parameter tuned with the following values:

- *epoch* $= [100, 200, 300]$
- *dropout* $= [0.1, 0.25, 0.5, 0.75]$
- *hidden layer size* $= [1, 2, 3, 4]$

The batch size is 100 for all experiments, excluding the place-types which used a batch-size of 10 as there were so few entities. For the linear SVM and the Decision Trees of depth-3, hyper-parameter optimization for these models was used with the same parameters as described in Section 4.5.2.

5.4.2 Quantitative Results for Feedforward Network

This section shows the performance of NNET-U and NNET-B, and the results for the best performing linear SVM that uses an unsupervised document embedding as input are shown. Additionally, results for decision trees limited to a depth of 3 that use disentangled feature representations derived from NNET-U and NNET-B as input are shown. Essentially, the results

	Movies	Placetypes	Newsgroups
U-Embedding (Linear SVM)	0.532082959	0.630040432	0.628171051
NNET-U	0.559090895	0.563313632	0.627577055
NNET-B	0.435345945	0.597008771	0.673618458
Term-features U-Embedding DT3	0.493072458	0.508163669	0.536686468
Term-features NNET-U DT3	0.505155338	0.460648739	0.50988517
Term-features NNET-B DT3	0.501545907	0.521526974	0.664793407
Cluster-features U-Embedding DT3	0.506096231	0.544001448	0.513367341
Cluster-features NNET-U DT3	0.472297312	0.541887565	0.478077988
Cluster-features NNET-B DT3	0.501510396	0.5967867	0.682928611

Table 5.1: F1-scores for each embedding type and domain

of NNET-U, NNET-B and the linear SVM that uses the best performing unsupervised document embedding as input serve as a reference point for the decision tree results. If the decision tree results do indeed outperform or match the neural networks, then they are performing well. The first intention of these results is to validate that disentangled feature representations are composed of semantic features, and this is measured as in Chapter 4 by the performance of low-depth decision trees when they are used as input. The second but more primary intention is to try and determine what can be expected qualitatively from these disentangled feature representations, and this is determined by comparison between different models. The main comparison, informed by the results and investigation in Chapter 4, is by comparing the results for the depth-3 limited decision trees that use disentangled feature representations derived from neural networks as input to the performance of the decision trees that used disentangled feature representations derived from unsupervised document embeddings as input. As a general overview of the results, the disentangled feature embeddings derived from neural networks perform as well as or better than the unsupervised representations on the task.

In particular, the quantitative results in Table 5.1 are F1-scores, where "U-Embedding" is the unsupervised embedding. The various methods used to obtain the results are as follows:

- **U-Embedding Linear SVM:** The best performing linear SVM for the task that used an unsupervised document embedding is used as input from Chapter 4.

- **DT3** This refers to a decision tree limited to a depth of 3.
- **NNET-U and NNET-B:** These are the scores when the output layer of the neural networks are used to predict the class of documents.
- **Term-features:** This refers to the results for decision trees limited to a depth of 3 when using disentangled feature representations as input, composed of rankings of entities on single-term directions.
- **Cluster-features:** This refers to the results for decision trees limited to a depth of 3 when using disentangled feature representations as input, composed of rankings of entities on cluster directions.

The assumption that interpretable features can be obtained from the neural networks using the method in Chapter 4 is validated, as not much predictive power is lost over the original neural network in most cases. In particular, for the domain of place-types and the domain of newsgroups the Decision Trees of depth 3 that used single-term features and cluster-features derived from NNET-B outperformed the other approaches by a significant margin. In the newsgroups domain, the low-depth decision trees that used the disentangled feature representation as input outperformed the neural network that it was derived from. This verifies the idea that decision trees of depth 3 that use disentangled feature representations as input can outperform more complex classifiers. Interestingly however, in the movies domain the neural network that used an unsupervised document embedding as input (NNET-U) performed the best, and NNET-B had a relatively low performance. This is likely because the vocabulary for the movies is large (limited to 100,000 unique terms for movies from the original vocabulary size of 551,080 versus 51,064 for newsgroups, see Section 3.3), and this resulted in overfitting.

These results give an expectation that well-represented (i.e. have a high NDCG score) features of the disentangled representation derived from NNET-U will be relevant to the task. Similarly, as in the movies domain the neural network that used a bag-of-words as input (NNET-B) performed poorly, it can be expected that properties that are well represented in this representation are less relevant to the task. However, all of these representations performed similarly in the depth-3 decision trees. This likely means that they have all been able to identify features that are particularly relevant to the class, e.g. a feature that corresponds to "Horror" when classifying the genres. This brings up the question, why does NNET-U perform better than the others,

and NNET-B significantly worse, but the low-depth decision trees that use disentangled feature representations derived from NNET-U and NNET-B perform similarly to each other? The explanation assumed in this section is that the features that are well-represented in NNET-U and NNET-B are not used in the decision trees, as features that are common to all representations are sufficient for good performance. This hypothesis is further investigated in the qualitative analysis (Section 5.4.3).

Direction results

Generally, the score of a depth-3 decision tree classifier when using a disentangled feature representation obtained from a neural network embedding is close to the performance of the neural network it was derived from. In the following, potential reasons why disentangled feature representations derived from neural network embeddings might perform well are described:

- Metadata that is not relevant to the task is not well-represented, meaning they are not disentangled from semantic features relevant to the task. (e.g. metadata like e-mail list names "listinfo" "robomod" are no longer highly separable, meaning that the only features found are meaningful)
- Features that are beneficial to classifying the task that were previously entangled are disentangled (e.g. "Blood" may be a feature that is linearly separable in the unsupervised representation, and that was beneficial for classifying the "Horror task, but in the neural network representation a feature for "Zombie" is also linearly separable which enables the classifier to better identify if a movie is classified as being in the "Horror" genre)
- The rankings for properties that are also encoded in the unsupervised document embedding more faithfully capture the corresponding semantic property (e.g. The "Comedy" direction more correctly ranks movies based on how funny they are) If the directions perform well and use similar features it is reasonable to assume that the rankings are better than in the unsupervised representation.

In conclusion, disentangled properties that can be used to classify entities can be identified in the document embeddings derived from supervised neural networks, and in the newsgroups domain

low-depth decision trees that use the disentangled feature representations derived from these neural network embeddings perform better than the original neural network. Assumptions are made about what kind-of features these disentangled feature representations will contain, in particular that disentangled feature representations derived from neural networks that perform well on the task will likely contain unique features and potentially improved rankings for features that are particularly relevant to the task. The next section qualitatively investigates disentangled feature representations derived from neural network embeddings.

5.4.3 Qualitative Investigation of Feedforward Networks

This section focuses on qualitatively investigating the characteristics of feed-forward neural network embeddings, as well as the disentangled feature representations derived from them. In particular, there are three main goals of this qualitative investigation section:

- Verify that semantic features derived from the feed-forward neural network embeddings are meaningful.
- Gain a better understanding of the disentangled feature representations derived from feed-forward neural network embeddings.
- Identify distinguishing characteristics of feed-forward neural networks.

If the semantic features obtained from the neural network makes intuitive sense, then they are verified to be meaningful. Disentangled feature representations derived from feed-forward neural network embeddings are explained using examples, and distinguishing characteristics of feed-forward neural networks are determined by comparisons between the semantic features derived from neural network embeddings and those derived from unsupervised embeddings.

When showing results for single directions, the words that these directions are labelled with e.g. "Blood" will be accompanied by the two terms with the most similar directions e.g. "Blood (Gore, Horror)". This is so the meaning of the direction can be understood more easily. Unlike the clustered features, the direction is not changed. The unsupervised embedding, NNET-U and NNET-B are used in this section for each domain.

Terms are scored using NDCG 4.3.2 as it was found to perform well on a variety of tasks and spaces in Chapter 4. The frequency threshold was set to the top 10,000 terms and the score threshold was set to the top 2,000 scoring terms in NDCG.

Top Scoring Terms

In this section, we look at the top-scoring terms for three different domains. These initial results are intended to give a general impression of what these terms and their associated similar directions are like, as well as to gain some initial insight into what the embeddings are representing.

In Table 5.2 the top NDCG scoring terms are listed for each domain and embedding type. One immediate observation is that the unsupervised embedding and the NNET with the unsupervised embedding as input ("NNET-U") have similar top scoring terms, e.g. "listinfo" "mailing". Extremely separable metadata directions that are not relevant to the task retained this separability despite the neural network being trained to optimize the task. This seems to indicate that directions that are separable in an input embedding will not be made less separable in the neural network embedding. In the case of the neural network using BOW as input (NNET-B), however, these terms are not present. This brings up the question, what exactly has the neural network learned such that it performs higher in F1-score in the case of the unsupervised embedding as input?

Common and Unique Terms to Each Embedding

In order to further investigate the question posed in the previous section, this section evaluates whether terms from the movies domain are unique to the embedding or common to all three embeddings. The intention is to see if NNET-U and NNET-B embeddings have introduced any new properties to the top 2,000. To this end, a basic procedure is used where if terms did not occur in either of the top 2,000 highest scoring terms for the other embeddings then they were considered not unique, and if a term occurs in the 2,000 top scoring terms of both the other two embeddings it is considered common.

Interestingly, there were only 14 unique terms for the unsupervised embedding, and only 19 for the neural network that used the unsupervised embedding as input. Meanwhile, there were 770

Movies		
Unsupervised Embedding	NNET BOW Input	NNET Embedding Input
listinfo (mailman, rec)	horror (pacing, dialog)	listinfo (mailman, rec)
robomod (mailman, rec)	westerns (russian, digitally)	robomod (mailman, rec)
mailing (mailman, listinfo)	documentary (joke, ultimate)	mailing (mailman, listinfo)
noir (fatale, femme)	comedies (actress, entertain)	noir (fatale, femme)
martial (kung, fight)	hilarious (disappointment, dozen)	horror (scary, horrific)
gay (homosexual, homosexuality)	laughs (woods, rights)	martial (kung, arts)
horror (scary, scares)	sci (equivalent, intent)	gay (homosexual, homosexuality)
prison (jail, prisoners)	adults (grown, rushed)	prison (jail, convicted)
arts (rec, listinfo)	songs (battle, speak)	animation (animated, cartoon)
musicals (musical, singing)	war (military, james)	arts (rec, listinfo)
mailman (listinfo, robomod)	western (don, stick)	musicals (musical, numbers)
Placetypes		
southcoast (filters, reala)	leafs (botanic, f100)	interchange (midday, elevated)
interchange (underpass, hk)	aerialphotography (beaver, kiteaerialphotography)	canonrebel (controluce, pigeons)
canonrebel (1855mm, nikond300s)	irvine (jay, meet)	rave (dj, erin)
statua (stern, 1st)	centralcoast (published, aloha)	winnipeg (konicaminolta, twincities)
municipal (citizen, farbe)	swell (fin, polar)	windmills (goldenhour, puffy)
madrid (noir, df)	pacificocean (puerto, waves)	statua (palacio, estatua)
reizen (seventies, canonef2470mmf28lusm)	trunks (birch, pair)	reizen (1022mm, t3)
commuter (underpass, muni)	southflorida (fla, hawaiian)	f456 (300mm, a550)
crime (illegal, violence)	lapland (alberi, topv333)	gibraltar (iow, upon)
leafs (iris, cyan)	sunbathing (underwater, relaxed)	song (singing, juni)
Newsgroups		
solid (design, single)	vol (vast, foot)	temperature (discoveries, surveys)
struggle (grew, landed)	volt (amplifier, soldered)	testify (ali, concentrated)
spreading (pursued, tolerant)	voltage (cautious, scanned)	solid (company, sides)
salt (drinking, combinations)	volume (fairly, distinguish)	tea (tech, buck)
random (attacks, described)	volumes (expressed, scenario)	widely (recently, 1982)
widely (developed, tend)	voluntary (demonstrating, explore)	denial (judaism, kurds)
temperature (layers, consumption)	volvo (aftermarket, horsepower)	detecting (skeptical, signals)
viable (motivation, emphasized)	tigers (carter, brady)	dick (quoted, seen)
vol (published, journal)	postage (straightforward, engaged)	pitches (screw, headers)
volt (garage, voltage)	povray (animations, pbmplus)	random (dropped, atheism)
voltage (amps, resistor)	starters (worthwhile, fame)	salt (stations, alike)
volume (hundred, frequently)	secular (descendants, fundamentalists)	vol (contents, students)
volumes (historians, turkish)	ring (increasing, behind)	volt (disagreement, clouds)
voluntary (heterosexual, posed)	utterly (des, retain)	voltage (criteria, disclosed)
volvo (saab, chrysler)	occurring (communicate, contacted)	volume (although, quickly)
tea (nagornokarabakh, mothers)	single (entire, prove)	volumes (raster, josephus)
quick (careful, usual)	oxygen (speeding, extraordinary)	voluntary (transmitted, satan)
utterly (violates, sinners)	widgets (shells, experimentation)	volvo (outlet, 302)

Table 5.2: Top-scoring directions measured using NDCG score

Movies			
Unique Unsupervised	Unique NNET-B	Unique NNET-U	Common
immigrants (immigrant, america)	unlike (terrific, efforts)	carry (recent, hoping)	noir (fatale, femme)
flashback (flashbacks, present)	efforts (detail, directors)	federal (fbi, agents)	martial (kung, fight)
bloated (spectacle, overlong)	impression (throw, truth)	possessed (demonic, forces)	horror (scary, scares)
chapter (previous, installment)	viewed (catch, escape)	faustus (geocities, html)	arts (rec, robomod)
rebel (rebellious, freedom)	suggest (wondering, trip)	spike (african, lees)	musicals (musical, singing)
assault (attack, violent)	focuses (sadly, thoughts)	dashing (handsome, excitement)	hilarious (funniest, laughing)
client (lawyer, attorney)	terms (theatre, marvelous)	wartime (wwii, bombing)	westerns (western, cowboy)
predecessor (sequel, sequels)	suppose (heres, greatly)	phantom (sees, opera)	jokes (laughs, joke)
competitive (competition, sport)	credit (clever, sequence)	theories (theory, conspiracy)	romantic (romance, romances)
jealous (attraction, crush)	exception (beat, passed)	robots (sci, princess)	animation (animated, cartoon)
betrayal (loyalty, affair)	heres (suppose, negative)	abused (abuse, abusive)	western (westerns, west)
artsy (pretentious, artistic)	fare (twenty, concerned)	fiance (fianc, engaged)	songs (song, lyrics)
hotel (manager, vacation)	unable (convey, accept)	hatred (hate, racism)	comedies (comedic, laughs)
stereotypical (stereotypes, clich)	deliver (depth, limited)	mysteries (mystery, clues)	war (soldiers, military)

Table 5.3: Terms from three different document embeddings, the unsupervised embedding, the neural network that used a bag-of-words as ioput and the neural network that used the unsupervised vector space as input. Arranged by NDCG, from highest to lowest.

unique terms for the neural network with bag-of-words as input. This shows that representing new directions was not the reason for superior performance for NNET-U. Rather, it is likely that terms that are relevant to the genre that were already in the top 2,000 top-scoring terms are more separable. In particular, we see that terms that are highly relevant to the task of genres are common to all embeddings, e.g. "horror (scary, scares)", "hilarious (funniest, laughing)", "jokes (laughs, joke)". This gives some indication of why in the Quantitative Investigation it was seen in Table 5.1 that although NNET-U outperformed NNET-B and a linear SVM that used the unsupervised embedding as input, the disentangled feature representations derived from these embeddings performed similarly. The explanation for this is that each embedding contains the same features that are most suitable to use in a low-depth decision tree for the task, but they differ in those that are less relevant to the task. This idea is further validated in Section 5.4.3.

The Difference Between the Term Scores in the Embeddings

In this section, the differences between the scores of terms in the movies domain are qualitatively analysed. The intention is to determine what some differences are between the unsupervised embeddings and the neural network embeddings, and evaluate previous assumptions

regarding the behaviour of the network. These results are for all 10,000 terms term directions obtained from each embedding.

In the first column of Table 5.4.3 terms that have a higher associated NDCG score in the case of the unsupervised embedding than the in the case of NNET-U are not clearly related to genres. For example, "yup", "wright", "wimpy" and "zoom", which are the top four highest score differences, do not seem related to genres at all. Meanwhile, the terms that have gained in associated NDCG score derived from in NNET-U are clearly relevant to genres, e.g. "animation", "adventure", "comedies". Although, it has also increased the separability of some metadata e.g. the term "listinfo", which is a problem already identified when using the unsupervised embedding as input to the neural network. This gives us some indication of what the neural network is doing to solve the task. This behaviour of the neural network decreasing separability of noisy terms and increasing separability of terms relevant to the task could contribute to the difference in score where the neural network outperformed a linear SVM on the unsupervised embedding.

Meanwhile NNET-B seems to have elevated terms that are not relevant to the task. As the neural network seems to have overfit due to the large vocabulary, it has made terms separable that are not immediately meaningful, e.g. "attends" "adds" "foray". However, as previously mentioned it did not have some of the same metadata as the unsupervised embedding. However, it seems to have made metadata unique to it separable.

Unsupervised - NNET-U	Unsupervised - NNET-B	NNET-U - Unsupervised	NNET-B - Unsupervised
zoom (cuts, editing)	zoom (cuts, editing)	allens (woody, allen)	atmosphere (mood, atmospheric)
wimpy (villian, instance)	zombies (zombie, undead)	animation (animated, cartoon)	biased (agree, showed)
wright (forrest, buyer)	williams (jim, includes)	adults (children, adult)	abomination (insult, horrible)
yards (shoot, yard)	willingly (explicitly, risk)	arts (rec, listinfo)	absent (absence, previous)
yup (junk, downright)	wrecked (crashes, crashed)	allen (woody, allens)	car (cars, driving)
watchers (underlying, holding)	welles (orson, kane)	adaptation (adaptations, adapted)	abandons (leaving, decides)
valiant (merit, admirable)	writers (write, finish)	animated (animation, cartoon)	attends (attending, attend)
wee (pee, pick)	willingness (conventional, goal)	aboard (ship, board)	adds (added, addition)
winding (road, wheel)	wars (enemy, war)	adventure (adventures, adventurous)	broader (greater, balanced)
yea (hey, wanna)	winner (winning, won)	australian (australia, sydney)	artwork (painting, animation)
wax (inclined, usage)	walt (disneys, disney)	addiction (addict, addicted)	cheerfully (gleefully, bursts)
woke (yeah, wake)	unstable (psychotic, mentally)	berardinelli (employers, distributor)	foray (marks, venture)
wrecked (crashes, crashed)	walking (walk, walks)	abuse (abused, abusive)	aboard (ship, board)
winded (significant, deliver)	wrath (gods, angry)	actress (actresses, lady)	absurd (ridiculous, absurdity)
wine (bottle, drink)	widely (recently, equally)	apes (monkey, monkeys)	civilization (civilized, survival)
wrapping (product, needed)	wondering (figured, wanting)	comedies (comedic, laughs)	abrupt (lacked, ended)
yahoo (faustus, faust)	yawn (dull, tedious)	listinfo (mailman, rec)	active (helped, allowed)
whip (beat, kick)	zeal (enthusiasm, celluloid)	africa (african, countries)	admiration (respect, remarkable)
zeal (enthusiasm, celluloid)	wolf (wolves, pack)	bogart (casablanca, noir)	consciousness (profound, meaning)
zone (twilight, concept)	wire (wires, walls)	animals (animal, humans)	abandoned (deserted, forced)
wayward (cross, join)	undead (zombie, zombies)	anime (animation, japan)	attitude (respect, reasons)
visitor (arrival, visiting)	winter (snow, frozen)	african (racial, racist)	graphically (graphic, violent)
weaving (weaves, crafted)	yesteryear (throwback, nostalgic)	band (bands, rock)	acceptable (rhodes, internetreviews)
yearns (desires, affection)	voted (vote, awards)	accurate (accuracy, accurately)	advances (developed, attractive)
youngster (kid, childhood)	unquestionably (arguably, closing)	accuracy (accurate, inaccuracies)	depths (beneath, surface)
yell (angry, yelling)	wifes (husband, marriage)	artist (artists, artistic)	advice (decided, spent)
walker (damage, bill)	worthwhile (skip, include)	catholic (priest, church)	explanation (explained, explain)

Table 5.4: The largest differences in term scores between embeddings for the movies. The first two columns are the terms that were most reduced in score, while the third and fourth columns are the terms that increased most in score.

Newsgroups Decision Tree Cluster-Features

In the newsgroups domain, low-depth decision trees when disentangled cluster features were used as input outperformed the neural network they were derived from. In this section we qualitatively investigate the clusters for the newsgroups, with particular interest in why the NNET-B clusters performed as well as the neural network and better than the unsupervised representation by a wide margin.

In Table 5.4.3, we examine the clustered features used in the decision trees for the newsgroups. In particular, we obtain the top decision tree node for each class. This is to give an overview of the different kinds of clusters and how they perform for each task. When examining the difference between the cluster-features for the unsupervised embedding and NNET-B, we can see that the features chosen as the top nodes of the decision tree seem more relevant to the task. Rather than refining the rankings of the previous properties, new cluster-directions have been found that more closely correspond to the task. For example, the cluster-feature used for comp.graphics seems to capture a more relevant and complex property "(pbmplus, sgis, sunview, phigs, pex, colormap, pixmap)" compared to the unsupervised embedding "(povray, raytracing)" as does the cluster-feature used for rec.autos "(sedan, camry, saab, ..., diesel, coupe)" versus "(sedan, camry)". Following this the conclusion is that having access to the BOW representation allows the neural network to better arrange the entities such that these natural cluster directions are meaningful to the class. However, the reason this did not occur for the movies domain was because the neural network overfit.

The best-performing cluster parameters for NNET-U resulted in directions that are clusters of terms not relevant to the task, and single-term features that are relevant to the class. In the examples in 5.4.3, only those single-term features that are relevant to the class are shown, as these cluster features were not used in the depth-3 decision tree. This means that the clustering process did not capture features that represent more abstract but relevant features despite separating relevant term directions, e.g. "rsa" is particularly relevant to the class sci.crypt, but the NNET-B cluster-feature of "(authentication, diffiehellman, publickey, 80bit, vesselin, skipjack)" captures a more abstract but relevant feature.

Newsgroups	Unsupervised Embedding	NNET-B	NNET-U
alt.atheism	(celestial, creationist, psalm, ..., fallacy, baptism)	(messenger, faiths)	(homosexuality)
comp.graphics	(povray, raytracing)	(pbmplus, sgis, sunview, phigs, pex, colormap, pixmap)	(tiff)
comp.os.ms-windows.misc	(winini, systemini, cica)	(cirrus, workgroups, hicolor, ..., keystrokes, sdk)	(xlib)
comp.sys.ibm.pc.hardware	(lpt1, irq, chipset, mfm, logitech)	(qemm, emm386, hernandez, ..., com2, brewers)	(motherboards)
comp.sys.mac.hardware	(centris, lciii)	(democrats, emissions, helicopters, enforced, tribes, reign, clintons)	(quadra)
comp.windows.x	(xdm, makefile, r4, ..., rainer, ow)	(xmotif, tvtwm, xfree86, ..., xsun, polygon)	(xlib)
misc.forsale	(diffiehellman, exterior, strips, ..., uucp, slots)	(267, saga, warriors, ..., pov, carnage)	(obo)
rec.autos	(sedan, camry)	(sedan, camry, saab, ..., diesel, coupe)	(toyota)
rec.motorcycles	(porsche, countersteering, msf)	(carb, bikes, reed)	(bikes)
rec.sport.baseball	(gant, duke, padres, ..., marlins, slg)	(larkin, marlins, pennant, ..., platoon, coaching)	(pitching)
rec.sport.hockey	(providence, keenan, hawks, ..., champs, ahl)	(nyr, messier, motto, ..., goaltending, keenan)	(stanley)
sci.crypt	(playback, ciphertext, cryptanalysis, ..., escrow, nist)	(authentication, diffiehellman, publickey, 80bit, vesselin, skipjack)	(rsa)
sci.electronics	(ham, amp, reactor, watts, amplifier, amps)	(omega, diode, charger, ..., antennas, joystick)	(voltage)
sci.med	(intake, calcium, antibiotic, ..., kidney, quack)	(bethesda, sensation, defects, ..., therapies, tablets)	(patients)
sci.space	(reboost, fusion, astronomers, ..., pasadena, galaxy)	(shafer, reusable, pluto, ..., orbiter, billboards)	(lunar)
soc.religion.christian	(divorce, sinless, moses, ..., corinthians, baptized)	(pastor, baptized, congregation, corinthians, repent, pagan)	(scripture)
talk.politics.guns	(federalist, unregistered, tyranny, ..., wright, shotguns)	(tpg, progun, 9mm, ..., rkba, shotguns)	(batf)
talk.politics.mideast	(arab, rebellion, syrian, ..., zionist, gaza)	(asalasdpaarf, revisionism, zionism, ..., hamas, grandparents)	(palestinian)
talk.politics.misc	(croats, redraw, shelling, ..., bosnians, yugoslavia)	(wiretapping, safeguards, vlsi)	(homosexual)
talk.religion.misc	(divorce, sinless, moses, ..., corinthians, baptized)	(thy, forgiveness, churchs, ..., protestant, ephesians)	(scripture)

Table 5.5: The top node of the Decision Tree for newsgroups

Movies Decision Tree Cluster-Features

In this section, we qualitatively investigate the cluster-features used in the top decision tree nodes for the movies domain, with a particular interest in why all embeddings have similar overall predictive performance despite containing very different properties (as talked about in Section 5.4.3).

One observation that follows the discussion in Section 5.4.3 is that the Adventure genre is classified using a cluster-feature that contains the word "adventure" in the case of NNET-U (adventure, spectacle, exciting, ..., seat, boyfriend), rather than the "(animation, animated, anime, voiced, cartoon)" cluster-feature that is re-used in the unsupervised embedding. This follows the previous observation as the "adventure" direction has a higher associated NDCG score in the NNET-U embedding versus the unsupervised embedding, as seen in . Despite this, the performance overall is the same.

The reason for the similar performance is likely that at the cost of re-organizing the representation such that the adventure cluster-direction existed, other cluster-directions were disrupted. The most prominent example of this is that the a good cluster for classifying animation is no longer present, instead this cluster feature "(voice, voiced, recording, voices, vocal, listening)". Another example is that in the case of NNET-U the history class is classified in the unsupervised representation by the seemingly relevant "(events, accuracy, accurate, facts, confusing)", but in the case of NNET-U this cluster is disrupted by seemingly irrelevant terms "(western, historical, musicians, ..., biography, propaganda)". Note that we do not really see this in the newsgroups Table for NNET-B, where instead new and meaningful cluster-features are found that are relevant to the task.

Movies	Unsupervised Embedding	NNET-B	NNET-U
Action	(fight, fighting, epic, ..., battle, weapons)	(martial)	(martial, fight, fighting, fights, choreography, choreographed, fighter)
Adventure	(animation, animated, anime, voiced, cartoon)	(adventure, adventures)	(adventure, spectacle, exciting, ..., seat, boyfriend)
Animation	(animation, animated, anime, voiced, cartoon)	(voiced)	(voice, voiced, recording, voices, vocal, listening)
Biography	(biography, biopic)	(biopic)	(gritty, historically, fiction, ..., accurate, accuracy)
Comedy	(hilarious, jokes, comedies, ..., funniest, funnier)	(comedies, funniest, funnier, ..., laughed, slapstick)	(hilarious, jokes, comedies, ..., funniest, funnier)
Crime	(noir, crime, caper, criminal, criminals, crimes)	(crime, police, criminal, cop, cops)	(gangster, gangsters)
Documentary	(documentary, footage, documentaries, interviews, interviewed, informative)	(documentary, footage, documentaries)	(documentary, footage, documentaries, ..., extras, interviewed)
Drama	(emotional, magical, waves, ..., silent, evidence)	(emotional, emotions, emotionally, ..., relationships, study)	(bond, emotional, families, ..., powerful, gripping)
Family	(adults, children, childrens, ..., parents, daughter)	(adult, ages, children, parents, teenager)	(disneys, walt)
Fantasy	(fantasy, fairy, fairytale)	(magic, colorful, comical, lavish, delightfully, dazzling, colors)	(fantasy, surreal, elm, fairy, fairytale, dreams, dream)
Film-Noir	(femme, fatale)	(noir, vintage)	(femme, fatale)
History	(events, accuracy, accurate, facts, confusing)	(historically)	(western, historical, musicians, ..., biography, propaganda)
Horror	(horror, creepy, slasher, ..., scares, scare)	(creepy, scare, spooky, scary, monster, menacing, nightmares)	(horror, creepy, scares, ..., spooky, chills)
Music	(musicals, songs, singing, ..., song, numbers)	(songs, musical, song, ..., sing, dancing)	(songs, singing, song, ..., sings, singer)
Musical	(musicals, songs, singing, ..., song, numbers)	(musicals, broadway)	(songs, singing, song, ..., sings, singer)
Mystery	(detective, investigation, mystery, clues, mysterious, investigating)	(mystery, dramatic, fiction, ..., psychological, slowly)	(thriller, suspense, hitchcock, ..., thrillers, suspenseful)
Romance	(romantic, charming, romance, ..., charm, chick)	(romance, romantic, chemistry, ..., handsome, scenery)	(wedding, marry, marries, marrying, bride)
Sci-Fi	(sci, alien, space, aliens, outer, invasion)	(sci, science, futuristic)	(science, scientific, scientist, ..., scientists, investigating)
Short	(episodes, episode, seasons, aired, television, storylines) (python, monty)	(agree, soundtrack, product, ..., happening, inside)	(geocities, html, aol, faustus)
Sport	(coach, sports, team, sport, football)	(sports, sport, coach)	(boxing)
Thriller	(thriller, suspense, adventure, ..., suspenseful, tension)	(thriller, suspense, thrillers, suspenseful, thrills, thrilling)	(thriller, suspense, hitchcock, ..., thrillers, suspenseful)
War	(war, soldiers, vietnam, ..., military, troops)	(wwii, german)	(war, soldiers, vietnam, ..., military, troops)
Western	(westerns, western)	(westerns)	(outlaw)

Table 5.6: The top node of the Decision Tree for Movies

Top-Ranking Entities on Decision Tree Cluster-Features in the Place-types Domain

The intention of this section is to investigate the rankings of entities in the embeddings and see if they make sense. The intuition is that despite neural network embeddings obtaining directions that make reasonable sense, their entities may contain irrelevant entities or be inaccurate. This experiment is used as further verification that features derived from neural networks from are indeed semantic. In Table 5.7 the five entities with the highest dot products, the five highest ranked entities, are shown alongside directions used as the top node of the best performing decision tree. Note that it may not seem meaningful to take only the top five entities, but as there are only 391 entities overall for the class, and very few of those are positive instances of the class, these top five entities are the most relevant for classifying the associated class correctly. Essentially, if these top five instances seem semantic, we can reasonably assume that the class will be classified correctly if the direction used is relevant to the class.

To begin, we look at the clusters used to classify "CollegeAndUniversity", the unsupervised representation uses the cluster "(annarbor, graduation, eugene, institute, highschool, cal)" that seems particularly relevant to educational institutions, and its associated entities also seem very meaningful and relevant "(college, campus, college campus, university, school)". For NNET-B, the cluster itself seems absurd "(investment)", but the associated entities are relevant to that cluster "(commercial real estate, large construction, retirement home, ..., rental property)". Similarly, the cluster for NNET-U does not seem relevant to the class and neither do its top entities "(snowboard, turism, amusementpark, ..., waltdisney, disneyworld)" "(video game store, theme park, space shuttle, launch pad, speedway)". So in this case, the entities seem reasonable and meaningful for their associated directions.

In general, the results for this table follow a similar trend. Entities align with the meaning of cluster-features, even in neural network embeddings. The problems of classification come from poor selection of these cluster-features, rather than the cluster-features of those entities being incorrect. There is an interesting difference in the case of classifying NightLifeSpot, where three distinct but relevant clusters and associated entities are found and used in each of the embeddings. In the unsupervised embedding, a cluster-feature related to music was used "(audience, instrument, musicians, ..., dancers, bands)" with top-ranked entities "(stage, bar, rock, sound, music venue)". In the case of NNET-B, a cluster for drinking alcohol was used

instead "(booze, vodka, whiskey, liquor)" with the top-ranked entities corresponding to places where alcohol can be bought "(dive bar, beer garden, karaoke bar, hotel bar, cocktail bar)", finally the NNET-U identifies a cluster more related to sex but also including tattoo and dance studios (shoulder, darkroom, topless, ..., boobs, lips) with associated entities "(dance studio, tattoo studio, shoulder, topless beach, strip club)".

In conclusion for this section, entities seem to make sense in the case of place-types where they align well with their cluster-feature. Additionally, combining the use of a Decision Tree, the associated cluster terms with the cluster-feature, as well as the top entities gives great insight into the representation and what it is doing, as each provides valuable context for each other.

Class	Decision Tree Cluster-Feature	Top 5 Entities
ArtsAndEntertainment	(snack, vegetable, lemon, pepper, vegetables)	(restaurant, market, asian restaurant, seafood restaurant, japanese restaurant)
CollegeAndUniversity	(annarbor, graduation, eugene, institute, highschool, cal)	(college, campus, college campus, university, school)
Food	(cuisine)	(restaurant, asian restaurant, japanese restaurant, seafood restaurant, chinese restaurant)
NightlifeSpot	(audience, instrument, musicians, ..., dancers, bands)	(stage, bar, rock, sound, music venue)
ParksAndOutdoors	(cloudscape, solitary, hazy, ..., fluffy, intense)	(coast, shore, coastline, shoreline, beach)
ProfessionalAndOtherPlaces	(nationalhistoriclandmark, register, revival, jefferson, independence, pioneer, civilwar)	(court house, courthouse, government building, state, cemetery)
Residence	(laundry, dwelling)	(house, home, residential building, apartment, historic building)
ShopsAndService	(mannequin, etsy, crafts, boutique, bags, jewelry, necklace)	(market, boutique, store, clothing store, vintage store)
TravelAndTransport	(donkey, costarica, tanzania, ..., reisen, andes)	(market, desert, national park, crater, volcano)
	NNET-B	
ArtsAndEntertainment	(ruraldecay)	(sugar mill, silo, corral, abandoned farm, ranch)
CollegeAndUniversity	(investment)	(commercial real estate, large construction, retirement home, ..., rental property)
Food	(dolphins)	(football field, training camp, strait, baseball stadium, continent)
NightlifeSpot	(booze, vodka, whiskey, liquor)	(dive bar, beer garden, karaoke bar, hotel bar, cocktail bar)
ParksAndOutdoors	(condos, venice, entertainment, ..., socks, americanflag)	(piano bar, public housing, waiting room, residential street, dining room)
ProfessionalAndOtherPlaces	(gardening, tiki, cocktail, ..., candle, horticulture)	(sand bar, vineyard, monsoon forest, ski lodge, topless beach)
Residence	(tribute, plains, des, acropolis, aa)	(battlefield, national monument, battlefield park, cliff dwelling, space shuttle)
ShopsAndService	(cloister)	(diocese, convent, tomb, college theater, study)
TravelAndTransport	(dmctz3, panasonicdmctz3, panasonictz3)	(railroad tunnel, terminal, airport tram, railroad signal, railroad yard)
	NNET-U	
ArtsAndEntertainment	(messy, schoolhouse, beatles, ..., halftimbered, publictransport)	(cubicle, newsroom, workroom, bedroom closet, detached house)
CollegeAndUniversity	(snowboard, turism, amusementpark, ..., waltdisney, disneyworld)	(video game store, theme park, space shuttle, launch pad, speedway)
Food	(boutique, liquor, olive, ..., counter, dessert)	(molecular gastronomy restaurant, whisky bar, gourmet shop, cocktail bar, juice bar)
NightlifeSpot	(shoulder, darkroom, topless, ..., boobs, lips)	(dance studio, tattoo studio, shoulder, topless beach, strip club)
ParksAndOutdoors	(nesting, tits, mouth, ..., gulls, avian)	(rookery, hunting reserve, wetland, wildlife reserve, nest)
ProfessionalAndOtherPlaces	(islam, muslim, supermarket, ..., sales, marruecos)	(retail outlet, electronics store, tourist information center, jewelry store, souk)
Residence	(heath, ward, lock, ..., stained, argyll)	(ventilation shaft, abandoned prison, sanatorium, abandoned complex, reformatory)
ShopsAndService	(video, candy, clothing, ..., skin, blonde)	(video game store, jewelry store, thrift store, workroom, nail salon)
TravelAndTransport	(survey, mammal, antenna, ..., paws, fur)	(crop farm, flowerbed, tongue, nest, crop circle)

Table 5.7: The placetypes clusters used at the top of the decision tree and the associated top-ranked entities

Top Entities for Common Terms

The previous section gave insight into the top five ranked entities, but did not illustrate the difference between the top five ranked entities of common directions derived from the unsupervised embedding and the neural embeddings. In this section, we investigate the differences between top entities for common terms. The method is the same as Section 5.4.3 to obtain these common terms, but instead of taking the top 2,000 terms we take the top 500, so as not to include low-scoring terms that will have poor entity representations across the embeddings.

The results show that in most cases, entities are semantically similar, meaning that common directions across both neural embeddings and the best-performing unsupervised embeddings have common directions that also have common top entities. For example, the term "arrival (ticket, departure)" always has entities related to airports "(taxiway, airport tram, airport, aircraft cabin, airport gate)" even in the case of NNET-U, where the entity "toll booth" is included "(airport tram, airport control tower, airport, airport lounge, toll booth)". This further validates the idea given earlier that if different disentangled representations are able to achieve similar performance when used as input to low-depth decision trees, then they likely have found common but meaningful directions that are relevant to the task. This is particularly interesting in the case of NNET-B, as it started from a bag-of-words, but remained very similar to an unsupervised entity embedding.

5.4.4 Summary of Results

First, directions in the movie domain were examined. When looking at top scoring terms in Section 5.4.3, it was found that NNET-U retained highly separable metadata directions. This seems to indicate that despite directions being uninformative, the neural network will not change the representation enough to make these directions less separable despite adjusting the embedding to better suit the task. In Section 5.4.3 it was found that NNET-U did not contain many unique directions in its top 2,000 scoring terms, but NNET-B did. The directions that were common to all embeddings seemed most relevant to the task. This gave an explanation for the reason that the disentangled feature representations derived from NNET-U, NNET-B and the unsupervised embedding all performed similarly: they all contained the directions that are most relevant to the task.

Cluster Features	**Top 5 Ranked Entities**
Unsupervised Embedding	
bass (drums, tom)	(indie theater, jazz club, music studio, music venue, rock club)
arrival (ticket, departure)	(taxiway, airport tram, airport, aircraft cabin, airport gate)
condo (condominium, apartments)	(rental property, condominium, condo, villa, plaza)
palazzo (ff, notte)	(campanile, palace, villa, triumphal arch, cuesta)
fans (ball, player)	(basketball stadium, hockey arena, soccer stadium, baseball stadium, football stadium)
animalplanet (feeder, natureselegantshots)	(zoo, zoological garden, nest, moais, animal shelter)
rent (homes, villas)	(rental property, real estate offices, cuesta, condo, condominium)
agriculture (agricultural, fields)	(cropland, conifer forest, olive grove, arboretum, sugar plantation)
champions (win, league)	(football stadium, soccer stadium, basketball stadium, college stadium, cuesta)
cielo (azul, nubes)	(cuesta, campanile, plaza, villa, playa)
drummer (drums, who)	(jazz club, cuesta, moais, music venue, indie theater)
accommodation (accomodation, guest)	(rental property, amenities, hotel pool, suite, ski chalet)
decayed (forgotten, exploring)	(abandoned airfield, sanatorium, concentration camp, abandoned prison, hospital ward)
yankees (louis, victory)	(baseball stadium, basketball stadium, baseball field, hockey arena, stadium)
NNET-B	
bass (drums, tom)	(indie theater, jazz club, music studio, music venue, rock club)
arrival (ticket, departure)	(taxiway, airport tram, airport, aircraft cabin, airport gate)
condo (condominium, apartments)	(rental property, condominium, condo, villa, plaza)
palazzo (ff, notte)	(campanile, palace, villa, triumphal arch, cuesta)
fans (ball, player)	(basketball stadium, hockey arena, soccer stadium, baseball stadium, football stadium)
animalplanet (feeder, natureselegantshots)	(zoo, zoological garden, nest, moais, animal shelter)
rent (homes, villas)	(rental property, real estate offices, cuesta, condo, condominium)
agriculture (agricultural, fields)	(cropland, conifer forest, olive grove, arboretum, sugar plantation)
champions (win, league)	(football stadium, soccer stadium, basketball stadium, college stadium, cuesta)
cielo (azul, nubes)	(cuesta, campanile, plaza, villa, playa)
drummer (drums, who)	(jazz club, cuesta, moais, music venue, indie theater)
accommodation (accomodation, guest)	(rental property, amenities, hotel pool, suite, ski chalet)
decayed (forgotten, exploring)	(abandoned airfield, sanatorium, concentration camp, abandoned prison, hospital ward)
yankees (louis, victory)	(baseball stadium, basketball stadium, baseball field, hockey arena, stadium)
NNET-U	
bass (tom, drums)	(music studio, jazz club, rock club, music school, piano bar)
arrival (departure, journey)	(airport tram, airport control tower, airport, airport lounge, toll booth)
condo (condominium, condominiums)	(condo, condominium, rental property, suite, hotel pool)
palazzo (palais, efs)	(campanile, palace, cathedral, villa, plaza)
fans (arena, name)	(football stadium, basketball stadium, soccer stadium, hockey arena, stadium)
animalplanet (tiere, vosplusbellesphotos)	(zoo, zoological garden, nest, animal shelter, rookery)
rent (rental, animalplanet)	(rental property, condo, suite, condominium, real estate offices)
agriculture (farming, petrol)	(crop farm, olive grove, arboretum, cropland, wheatfield)
champions (win, victory)	(football stadium, soccer stadium, stadium, basketball stadium, hockey arena)
cielo (ciel, nubes)	(campanile, cuesta, cathedral, plaza, villa)
drummer (guitarist, drums)	(jazz club, rock club, stage, piano bar, music venue)
accommodation (condominium, hostel)	(amenities, rental property, suite, hotel pool, resort)
decayed (forgotten, decaying)	(sanatorium, abandoned airfield, barracks, abbey, military barracks)
yankees (mlb, league)	(baseball stadium, baseball field, avenue, basketball stadium, hockey arena)

Table 5.8: Terms common for all embeddings and the associated ranking of entities on those term directions for each embedding. Arranged by NDCG-score in the unsupervised embedding..

In 5.4.3 the largest differences in score between directions was examined. It was found that terms relevant to the task had an increased associated NDCG score in the NNET-U representation, and NNET-B had the highest score increases for directions that were not relevant to the task. This gives some explanation for why NNET-B performed worse than NNET-U, NNET-B likely represented terms that were not relevant to the task. In 5.4.3 it was hypothesized that the reason that this did not result in overall higher performance in a low-depth decision tree when using a disentangled feature representation derived from NNET-B as input is because despite it improving performance when classifying the "Adventure" genre, it sacrificed performance on another genre, because a cluster-feature relevant to classifying the "Animation" genre was not present.

In Section 5.4.3 cluster-features derived from NNET-B in the newsgroups domain were examined, as they performed as well as NNET-B. It was found that for NNET-B, the features used as the top decision tree nodes seemed to capture abstract concepts more relevant to the task than in the case of NNET-U or the unsupervised embedding. This simply explains the increase in performance.

In Section 5.4.3 it was found that entities for features are meaningful in almost all cases, and across all embedding models. This further verifies that features are semantic across domains even when derived from neural network embeddings. It is speculated that using meaningful entities could make the features more understandable. This is further investigated in section 5.4.3, where it is found that single-term directions common to all embeddings have similar entities across all embeddings. This further validates the idea in section 5.4.3 that each embedding model in the movies domain found similar relevant directions and associated entities that are most relevant to the task, as it shows that common directions are likely common in entity rankings for the domain as well.

5.5 Qualitative Evaluation of Auto-encoders

We base our experiments on the movie reviews domain. To collect the terms that are likely to correspond to property names, we collect adjectives and nouns that occur at least 200 times in the movie review data set, collecting 17,840 terms overall. These parameters are used for all entity embeddings induced by the auto-encoder.

5.5.1 Software, Architecture and Settings

To implement the denoising auto-encoders, the Keras[1] library is used. As in Chapter 4, scikit-learn is used for the SVM implementation. All of the code and data for this Chapter is available on GitHub[2]. A 200 dimensional MDS space is used as the input to our stack of auto-encoders, as this performed the best in Chapter 4. The network is trained using stochastic gradient descent and the mean squared error loss function. For the encoders and decoders, the tanh activation function is used. For the first auto-encoder, the same size layer as the input is maintained. Afterwards, the hidden representation size is halved each time it is used as input to another auto-encoder, and this process is repeated three times, resulting in four new hidden representations $\{Input : 200, Hidden : 200, 100, 50, 25\}$. The input space is corrupted using Gaussian noise with a standard deviation of 0.6 each time. This was chosen as higher noise values seemed to result in meaningless directions in initial results, but lesser noise values did not result in much change. As the lower layers are closer to the bag-of-words representation and are higher dimensional, the Kappa scores are higher in earlier spaces, as it is easier to separate entities. We address this in the clusters by setting the score threshold T using Kappa score such that the number of terms we choose from is twice the number of dimensions in the space. Similarly, we set the threshold for the frequency of the words such that 12,000 directions are available to assign to the cluster centres in every space.

5.5.2 Qualitative Evaluation of Induced Clusters

In Table 5.9, we illustrate the differences between clusters obtained using standard auto-encoders and denoising auto-encoders. Layer 1 refers to the hidden representation of the first auto-encoder, and Layer 4 refers to the hidden representation of the final auto-encoder. As single labels can lead to ambiguity, in Table 1 we label clusters using the top three highest scoring terms in the cluster. Clusters are arranged from highest to lowest Kappa score.

Both auto-encoders model increasingly general properties, but the properties obtained when using denoising auto-encoder properties are more general. For example, the normal auto-encoder captures properties like “Horror" and “Thriller", but does not capture more general properties

[1] Keras

[2] `https://github.com/eygrr/RulesFromAuto-encoders`

like “Society" and “Relationship". Further, “Gore" is the most similar to the directions “Zombie" and “Zombies" in Layer 1, the most similar directions of “Budget" and “Effects" in Layer 4. We can assume that this means that the feature in Layer 4 represents the more general idea of a "Horror" movie, rather than a "Zombie" movie, by being grouped with directions that are more relevant to all horror movies rather than only some horror movies.

5.5.3 Qualitative Evaluation of Induced Symbolic Rules

Our aim in this work is to derive symbolic rules that can be used to explain the semantic relationships between properties derived from increasingly general entity embeddings. We provide examples of such rules in this section. Since the number of all induced rules is large, here we only show high accuracy rules that cover 200 samples or more. Still, we naturally cannot list even all the accurate rules covering more than 200 samples. Therefore we focus here on the rules which are either interesting in their own right or exhibit interesting properties, strengths or limitations of the proposed approach. The complete list of induced rules is available in the appendix.

For easier readability, we post-process the induced rules. For instance, the following is a rule obtained for the property “Gore” in the third layer of the network shown in the original format produced by JRip:

```
IF scares-L2 <= 6 AND blood-L2 <= 8 AND funniest-L2 >= 22
=> classification=+ (391.0/61.0)
```

In this rule, `scares-L2 <= 6` denotes the condition that the movie is in the top 6% of rankings for the property “`scares`" derived from the hidden representation of the second auto-encoder. We will write such conditions simply as “$\texttt{Scares}_2$". Similarly, a condition such as `funniest-L2 >= 22`, which indicates that the property is not in the top 22%, will be written as "$\texttt{NOT Funniest}_2$". In this simpler notation the above rule will look as follows:

$\texttt{IF Scares}_2 \texttt{ AND Blood}_2 \texttt{ AND NOT Funniest}_2 \texttt{ THEN Gore}_3$

This rule demonstrates an interpretable relationship. However, we have observed that the meaning of a rule may not be clear from the property labels that are automatically selected. In such

Standard Auto-encoder		**Denoising Auto-encoder**	
Layer 1	Layer 4	Layer 1	Layer 4
horror: terror, horrific	horror: victims, nudity	gore: zombie, zombies	society: view, understand
thriller: thrillers, noir	documentary: perspective, insight	jokes: chuckle, fart	emotional: insight, portrays
comedies: comedy, timing	blood: killing, effects	horror: terror, horrific	stupid: flick, silly
adults: disney, childrens	suspense: mysterious, tense	emotionally: tragic, strength	gore: budget, effects
husband: wife, husbands	thriller: thrillers, cop	gags: zany, parodies	military: war, ship
relationships: intimate, angst	gory: gruesome, zombie	hindi: bollywood, indian	romance: younger, handsome
nudity: naked, gratuitous	beautifully: satisfying, brilliantly	touching: teach, relate	ridiculous: awful, worse
political: politics, nation	emotional: complex, struggle	scary: frightening, terrifying	government: technology, footage
smart: slick, sophisticated	laughed: laughing, loud	documentary: document, narration	awesome: chick, looked
creepy: sinister, atmospheric	charming: delightful, loves	adults: disney, teaches	political: country, documentary
laughed: humorous, offensive	hilarious: funny, parody	laughed: brow, laughter	relationship: relationships, sensitive
adventure: adventures, ship	scares: halloween, slasher	thriller: thrillers, procedural	horror: genre, dark
actions: reaction, innocent	funniest: funnier, gags	cgi: animated, animation	waste: concept, plain
cute: adorable, rom	emotions: respect, relationships	suspense: clues, atmospheric	army: disc, studio
british: england, accent	laugh: mom, crazy	dumb: mindless, car	combat: enemy, weapons
horrible: worse, cheap	filmmaker: approach, artist	political: propaganda, citizens	supporting: office, married
narrative: filmmaker, structure	drama: portrayed, portrayal	witty: delightfully, sarcastic	amazon: bought, copy
digital: dolby, definition	interviews: included, showed	laughing: outrageous, mouthed	study: details, detail
gory: graphic, gruesome	comedic: comedies, humorous	relationships: ensemble, interactions	land: water, super
romantic: handsome, attractive	emotionally: central, relationships	creepy: mysterious, eerie	chemistry, comedies, comedic

Table 5.9: A comparison between the first layers and the fourth layers of two different kinds of auto-encoders.

cases, it is beneficial to label them by including the most similar cluster terms. For example, using the cluster terms below we can see that “Flick" relates to “chick-flicks" and that “Amazon" relates to old movies:

IF Flick_2 AND Sexual_2 AND Cheesy_2 AND NOT Amazon_2 THEN Nudity_3

Flick_2: {Flicks, Chick, Hot}
Amazon_2: {Vhs, Copy, Ago}

Rules derived from later layers use properties described by rules from previous layers. By seeing rules from earlier layers that contain properties in later layers, we can better understand what the components of later rules mean. Below, we have provided rules to explain the origins of components in a later rule:

IF Emotions_2 AND Actions_2 THEN Emotions_3
IF Emotions_2 AND Emotion_2 AND Impact_2 THEN Journey_3
IF Emotions_3 AND Journey_3 THEN Adventure_4

We observe a general trend that as the size of the representations decreases and the entity embeddings become smaller, rules have fewer conditions, resulting in overall higher scoring and more interpretable rules. To illustrate this, we compare rules from an earlier layer to similar rules in a later layer:

IF Romance_1 AND Poignant_1 AND NOT English_1 AND NOT French_1
AND NOT Gags_1 AND NOT Disc_1 THEN Relationships_2

IF Relationships_2 AND Emotions_2 AND Chemistry_2 THEN Romantic_3
IF Emotions_2 AND Compelling_2 THEN Beautifully_3
IF Warm_2 AND Emotions_2 THEN Charming_3
IF Emotions_2 AND Compelling_2 THEN Emotional_3

Rules in later layers also made effective use of a NOT component. Below, we demonstrate some of those rules:

IF Touching_3 AND Emotions_3 AND NOT Unfunny_3 THEN Relationship_4
IF Laughs_3 AND Laugh_3 AND NOT Compelling_3 THEN Stupid_4

IF Touching_3 AND Social_3 AND NOT Slasher_3 THEN Touching_4

As the same terms were used to find new properties for each space, the obtained rules sometimes use duplicate property names in their components. As the properties from later layers are a combination of properties from earlier layers, the properties in later layers are refinements of the earlier properties, despite having the same term. Below, we provide some examples to illustrate this:

IF Emotions_2 AND Actions_2 THEN Emotions_3

Emotions_2: {Acted, Feelings, Mature}
Actions_2: {Control, Crime, Force}
Emotions_3: {Emotion, Issue, Choices}

IF Horror_2 AND Creepy_2 AND Scares_2 THEN Horror_3

Horror_2: {Terror, Horrific, Exploitation}
Creepy_2: {Mysterious, Twisted, Psycho}
Scares_2: {Slasher, Supernatural, Halloween}
Horror_3: {Creepy, Dark, Chilling}

IF Touching_2 AND Chemistry_2 THEN Touching_3
IF Touching_2 AND Emotions_2 THEN Touching_3
IF Compelling_2 AND Emotional_2 AND Suspense_2 THEN Compelling_3
If Romance_2 AND Touching_2 AND Chemistry_2 THEN Romance_3
IF Emotionally_2 AND Emotions_2 AND Compelling_2 THEN Emotionally_3

5.6 Conclusion

Feedforward neural networks and auto-encoders are qualitatively investigated using the methods outlined in Chapter 4. These properties are meaningful and interpretable for both types of neural network architecture. In the case of feed-forward networks, it is observed that associated entities

of features are meaningful, and similar features are found in all embeddings. These associated top entities are particularly relevant when understanding the semantic features. Further, the performance of a feed-forward network seems to rely on making particularly relevant properties to the task separable, as seen in the case of NNET-B in the newsgroups domain. If properties that are relevant to the task are common to all embedding models, low-depth decision trees that use these disentangled feature representations as input perform similarly, for example in the movies domain.

In the case of the auto-encoders as the space size reduces the representation becomes more abstract, and additionally through qualitative investigation rules show some promise in being used to link together properties in the layers and form some explanation of abstraction. However, it is found that the Kappa score of the directions derived from the embeddings is lower in more stacked auto-encoder embeddings, meaning that they are less separable. This is likely due to repeated non-linear transformations. This brings-up the question, is it possible to fine-tune the embedding such that features remain linearly separable but also become more abstract? This lead on to the work in Chapter 5, where a fine-tuning process is introduced to improve the quality of the disentangled feature representations. To expand on the work in this chapter, it would be interesting to see how the interpretability of these disentangled representations could be compared to alternative approaches, and they could be used to benefit those that work on those safety and fairness of neural network models, in particular to make black-box state-of-the-art models transparent by explaining layer-to-layer connections.

Chapter 6

Modelling Semantic Features in Fine-Tuned Vector Spaces

6.1 Introduction

Chapter 4 introduced a method to obtain feature-directions from vector-spaces, as well as methods to test the quality of these feature-directions and their associated entity rankings. Then, this method was applied in Chapter 6 to obtain feature-directions from the layers of neural networks. However, feature-directions obtained from either of these vector spaces can sometimes be sub-optimal. For example in the case of neural network auto-encoders, it was found that the quality of feature-directions in a auto-encoder representations degrade the more that the auto-encoder layers are stacked.

In figure 6.1, a problem that can occur with feature-directions from representations learned with a similarity-centred objective, e.g. Multi-Dimensional Scaling (see Section 2.5.2), is illustrated. This is an example problem in the toy domain of shapes, where basic geometric shapes are embedded in a two-dimensional space. In this example, directions have been identified which encode how light an object is and how closely its shape resembles a square. While most of the shapes embedded in this space are grey-scale circles and squares, one of the shapes embedded in this space is a red triangle, a clear outlier. When considering that the objective the space is learned with is similarity, the spatial representation for this triangle is correct, as it is far from all the other shapes. However, when ranking the shapes on the feature-directions for square and light, the outlier takes up an extreme position on the rankings. This means that the triangle is ranked incorrectly, as it is considered to be the shape that most exhibits the features “light” and “square”.

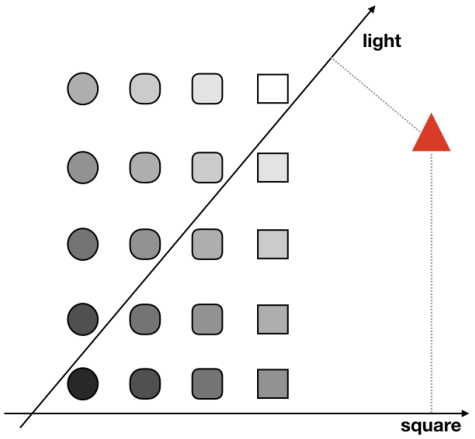

Figure 6.1: Toy example showing the effect of outliers in a two-dimensional embedding of geometric shapes..

Ideally, representations would be learned with knowledge of feature-directions. For example, the method to learn a representation of the toy domain would know that it should model the features "square" and "light" rather than a similarity objective, so that this triangle would end up closer to the bottom-left corner. However, as we cannot a-priori determine the features the space must be learned from, it is difficult to learn a representation in this way. This chapter instead introduces an unsupervised method, that given a representation and its associated feature-directions, can obtain a vector space and associated feature-directions where the quality of the feature-directions is prioritized over the similarity structure. The intention of this method is to resolve issues like those described in the previous two paragraphs.

To introduce the idea behind the method, we start with the assumption that each feature-direction is characterised by one feature-word, which describes the feature: if the feature-ranking of a document on a feature-direction is faithful to the bag-of-words score for the feature-word in the document, then the feature-ranking is good. To give an example of why this assumption is useful, in the IMDB movies domain 3.2 Multi-Dimensional Scaling (MDS) spaces there is the case of an Indian Bollywood movie that is very unlike other movies, as its reviews only use language specific to Bollywood films and the number of reviews it has is low overall. This movie occupies a top-ranking position in a variety of feature-directions, as a consequence of it being very dissimilar to other movies. The fine-tuning process solves this problem by attempting to match its ranking in the vector space that is very high, to its bag-of-words value, which is zero. This results in this obscure outlier movie being moved down drastically in the rankings.

Feature direction	Highest ranking objects	Highest fine-tuned ranking objects
{steep, climb, slope}	mountain, landscape, national park	ski slope, steep slope, slope
{illuminated, illumination, skyscraper}	building, city, skyscraper	tall building, office building, large building
{play, kid, kids}	school, field, fence	college classroom, classroom, school
{spooky, creepy, scary}	hallway, fence, building	hospital room, hospital ward, patient room
{amazing, dream, awesome}	fence, building, beach	hotel pool, resort, beach resort
{pavement, streetlight, streets}	sidewalk, fence, building	overpass road, overpass, road junction
{dead, hole, death}	fence, steps, park	grave, cemetery, graveyard
{spire, belltower, towers}	building, arch, house	bell tower, arch, religious site
{stones, moss, worldheritage}	landscape, fence, steps	ancient site, ancient wall, tomb
{mosaic, tile, bronze}	building, city, steps	cathedral, church, religious site

Table 6.1: Comparing the highest ranking place-type objects in the original and fine-tuned space.

To give some additional examples, in Table 6.1, names of documents are shown ranked on feature directions in the domain of place-types (See Section 3.2). In these examples, for the cluster-feature $\{steep, climb, slope\}$, the top ranked document *mountain* is clearly relevant. However, the next two documents — *landscape* and *national park* — are not directly related to this feature. Intuitively, they are ranked highly because of their similarity to *mountain* in the vector space. Similarly, for the second feature, *building* is ranked highly because of its similarity to *skyscraper*, despite intuitively not having this feature. Finally, *fence* received a high rank for several features, mostly because it is an outlier in the space.

Generally, the method that fine-tunes vector spaces and their associated feature-directions is as follows: First, a vector space is learned from bag-of-words representations of the considered documents, using a standard similarity-centric method or neural network. Next, the method from chapter 4 is used to obtain feature-directions and their associated words from a vector space. Then, following our assumption outlined in the previous paragraph, documents are ranked on the feature-direction's associated words using the bag-of-words. Finally, this ranking is used to fine-tune the vector space and feature-directions so that the resulting feature-rankings are more faithful to the ranking on the bag-of-words.

This chapter is a follow-up of the previous two chapters, where previously feature-directions are identified in a variety of vector-spaces, and their potential applications are discussed, this Chapter focuses on improving the quality of these feature-directions to achieve better results. This chapter continues with explaining the method to fine-tune a vector space and its associated feature directions using a bag-of-words in detail. Afterwards, we show quantitative results to

see how the fine-tuning affects simple interpretable classifiers (as in chapter 4). Finally, we end with a conclusion for potential future work.

6.2 Fine-Tuning Vector Spaces and their Associated Feature Directions

To improve the directions and address these problems, we propose a method for fine-tuning the semantic space representations and corresponding feature directions. First, it is explained how to obtain target rankings from PPMI scores. Then, the neural network that uses these target rankings to improve the vector space and its associated feature-directions is described. The main idea is to use the BoW representations of the objects as a kind of weak supervision signal: if an object should be ranked highly for a given feature, we would expect the words describing that feature to appear frequently in its description. To obtain the target rankings, for each feature f we determine a total ordering $\preccurlyeq_f$ such that $o \preccurlyeq_f o'$ iff the feature f is more prominent in the BoW represention of object o' than in the BoW representation of o. We will refer to $\preccurlyeq_f$ as the *target ranking* for feature f. If the feature directions are in perfect agreement with this target ranking, it would be the case that $o \preccurlyeq o'$ iff $v_C \cdot o \leq v_C \cdot o'$. Since this will typically not be the case, we subsequently determine *target values* for the dot products $v_C \cdot o$. These target values represent the minimal way in which the dot products need to be changed to ensure that they respect the target ranking. Once these rankings have been obtained, we use a simple feedforward neural network to adapt the semantic space representations o and feature directions v_C to make the dot products $v_C \cdot o$ as close as possible to these target values.

6.2.1 Generating Target Rankings

Let $C_1, ..., C_K$ be the clusters that were found using the method from Section 4.3.1. Each cluster C_i typically corresponds to a set of semantically related words $\{w_1, ..., w_n\}$, which describe some salient feature from the considered domain. From the BoW representations of the objects, we can now define a ranking that reflects how strongly each object is related to the words from this cluster. To this end, we represent each object as a bag of clusters (BoC) and then compute PPMI scores over this representation. In particular, for a cluster $C = \{w_1, ..., w_m\}$, we define

$n(C, o) = \sum_{i=1}^{m} n(w_i, o)$. In other words, $n(C, o)$ is the total number of occurrences of words from cluster C in BoW representation of o. We then write $\mathit{ppmi}(C, o)$ for the PPMI score corresponding to this BoC representation, which is evaluated in the same way as $\mathit{ppmi}(C, o)$, but using the counts $n(C, o)$ rather than $n(w, o)$. The target ranking for cluster C_i is then such that o_1 is ranked higher than o_2 iff $\mathit{ppmi}(C_i, o_1) > \mathit{ppmi}(C_i, o_2)$. By computing PPMI scores w.r.t. clusters of words, we alleviate problems with sparsity and synonymy, which in turn allows us to better estimate the intensity with which a given feature applies to the object. For instance, an object describing a violent movie might not actually mention the word 'violent', but would likely mention at least some of the words from the same cluster (e.g. 'bloody' 'brutal' 'violence' 'gory'). Similarly, this approach allows us to avoid problems with ambiguous word usage; e.g. if a movie is said to contain 'violent language', it will not be identified as violent if other words related to this feature are rarely mentioned.

6.2.2 Generating Target Feature Values

Finding directions in a vector space that induce a set of given target rankings is computationally hard[1]. Therefore, rather than directly using the target rankings from Section 6.2.1 to fine-tune the semantic space, we will generate target values for the dot products $v_{C_j} \cdot o_i$ from these target rankings. One straightforward approach would be to use the PPMI scores $\mathit{ppmi}(C_j, o_i)$. However these target values would be very different from the initial dot products, which among others means that too much of the similarity structure from the initial vector space would be lost. Instead, we will use isotonic regression to find target values $\tau(C_j, o_i)$ for the dot product $v_{C_j} \cdot o_i$, which respect the ranking induced by the PPMI scores, but otherwise remain as close as possible to the initial dot products.

Let us consider a cluster C_j for which we want to determine the target feature values. Let $o_{\sigma_1}, ..., o_{\sigma_n}$ be an enumeration of the objects such that $\mathit{ppmi}(C_j, o_{\sigma_i}) \leq \mathit{ppmi}(C_j, o_{\sigma_{i+1}})$ for $i \in \{1, ..., n-1\}$. The corresponding target values $\tau(C_j, o_i)$ are then obtained by solving the

[1] It is complete for the complexity class $\exists\mathbb{R}$, which sits between NP and PSPACE [63].

following optimization problem:

$$\textbf{Minimize:} \quad \sum_i (\tau(C_j, o_i) - v_{C_j} \cdot o_i)^2$$

$$\textbf{Subject to:} \tau(C_j, o_{\sigma_1}) \leq \tau(C_j, o_{\sigma_2}) \leq ... \leq \tau(C_j, o_{\sigma_n})$$

6.2.3 Fine-Tuning

We now use the target values $\tau(C_j, o_i)$ to fine-tune the initial representations. To this end, we use a simple neural network architecture with one hidden layer. As inputs to the network, we use the initial vectors $o_1, ..., o_n \in \mathbb{R}^k$. These are fed into a layer of dimension l:

$$h_i = f(Wo_i + b)$$

where W is an $l \times k$ matrix, $b \in \mathbb{R}^l$ is a bias term, and f is an activation function. After training the network, the vector h_i will correspond to the new representation of the ith object. The vectors h_i are finally fed into an output layer containing one neuron for each cluster:

$$g_i = Dh_i$$

where D is a $K \times l$ matrix. Note that by using a linear activation in the output layer, we can interpret the rows of the matrix D as the K feature directions, with the components of the vector $g_i = (g_i^1, ..., g_i^K)$ being the corresponding dot products. As the loss function for training the network, we use the squared error between the outputs g_i^j and the corresponding target values $\tau(C_j, o_i)$, i.e.:

$$\mathcal{L} = \sum_i \sum_j (g_i^j - \tau(C_j, o_i))^2$$

The effect of this fine-tuning step is illustrated in the right-most column of Table 6.1, where we can see that in each case the top ranked objects are now more closely related to the feature, despite being less common, and outliers such as ‘fence’ no longer appear.

6.3 Quantitative Evaluation

To evaluate our method, as in Chapter 4 we consider the problem of learning interpretable classifiers. In particular, we learn decision trees which are limited to depth 1 and 3, which use

20 Newsgroups	F1 D1	F1 D3	F1 DN
FT MDS	**0.50**	**0.47**	**0.44**
MDS	0.44	0.42	0.43
FT PCA	0.40	0.36	0.34
PCA	0.25	0.27	0.36
FT Doc2Vec	0.44	0.42	0.41
Doc2Vec	0.29	0.34	0.44
FT AWV	0.47	0.45	0.40
AWV	0.41	0.38	0.43
FT AWVw	0.41	0.41	0.43
AWVw	0.38	0.40	0.43
LDA	0.40	0.37	0.35

Table 6.2: Results for 20 Newsgroups.

the rankings induced by the feature directions as input. This allows us to simultaneously assess to what extent the method can identify the right features and whether these features are modelled well using the learned directions. Note that depth 1 trees are only a single direction and a cut-off, so to perform well, the method needs to identify a highly relevant feature to the considered category. We can understand that the most demonstrable improvements for this method over the original directions will be in Depth 1 trees, as if the rankings for the important feature-directions are improved then these will be also. Depth 3 decision trees are able to model categories that can be characterized using at most three feature directions.

Methodology

All tasks are evaluated as binary classification tasks. We randomly split the datasets into 2/3 for training and 1/3 for testing. For the place-types, we use 5-fold cross validation.

We used the logistic regression implementation from scikit-learn to find the directions.

In Chapter 4 the hyper-parameters were chosen in stages. First, parameters for the best word-directions were found. Then, these best word-directions were taken and the best cluster parameters were found for these best word-directions. However, for these experimental results, we optimize the hyper-parameters together for word-directions, clustering and fine-tuning, where the

Movie Reviews

Genres	D1	D3	DN	**Keywords**	D1	D3	DN	**Ratings**	D1	D3	DN
FT MDS	**0.57**	**0.56**	0.51	FT MDS	**0.33**	**0.33**	0.24	FT MDS	**0.49**	**0.51**	**0.46**
MDS	0.40	0.49	**0.52**	MDS	0.31	0.32	**0.25**	MDS	0.46	0.49	**0.46**
FT AWV	0.42	0.42	0.39	FT AWV	0.25	0.25	0.15	FT AWV	0.47	0.44	0.39
AWV	0.35	0.44	0.43	AWV	0.26	0.21	0.19	AWV	0.44	0.48	0.41
LDA	0.52	0.51	0.45	LDA	0.22	0.19	0.18	LDA	0.48	0.48	0.41

Place-types

Geonames	D1	D3	DN	**Foursquare**	D1	D3	DN	**OpenCYC**	D1	D3	DN
FT MDS	0.32	0.31	0.24	FT MDS	0.41	0.44	0.41	FT MDS	0.35	0.36	0.30
MDS	0.32	0.31	0.21	MDS	0.38	0.42	0.42	MDS	0.35	0.36	0.29
FT AWV	0.31	0.29	0.23	FT AWV	0.39	0.42	0.41	FT AWV	0.37	**0.37**	0.28
AWV	0.28	0.28	0.22	AWV	0.32	0.37	0.31	AWV	0.33	0.35	0.26
LDA	**0.34**	**0.32**	**0.27**	LDA	**0.55**	**0.48**	**0.47**	LDA	**0.40**	0.36	**0.31**

Table 6.3: The results for Movie Reviews and Place-Types on depth-1, depth-3 and unbounded trees.

IMDB Sentiment	D1	D3	DN
FT PCA	0.78	0.80	0.79
PCA	0.76	**0.82**	**0.80**
FT AWV	0.72	0.76	0.71
AWV	0.74	0.76	0.71
LDA	**0.79**	0.80	0.79

Table 6.4: Results for IMDB Sentiment.

best-parameters for each of these stages are those that ultimately produce the best-performing rankings for the fine-tuning on a decision tree. This is because fine-tuning is sensitive to which clusters and directions are included, as optimizing the ranking for one feature-direction may disrupt the ranking for another. This can be illustrated by the idea of optimizing a ranking for a direction on a noisy term like 'berardin', which refers to some metadata from the review text was

optimized, then it's unlikely that this would benefit the other directions. However, if multiple directions that correspond to different genres were optimized like 'Horror' and 'Funny', then it's likely that they would all benefit from a better representation. Cluster-directions are used because if all hyper-parameters are trained together, we can expect to find a set of directions that work with each other more easily than by limiting frequency for word-directions.

We evaluate for all domains described in Chapter 3 excluding reuters. When learning word directions, only sufficiently frequent words are considered. In Chapter 4 this was chosen as a hyper-parameter, but as all parameters for each stage are tuned together it would take far too much time to optimize in this way, so it is chosen beforehand. It is chosen by pre-determining thresholds loosely based on the size of the vocabulary for the domain. We chose 100 for the movies dataset, 50 for the place-types, 30 for the 20 newsgroups datasedt, and 50 for the IMDB sentiment dataset.

For hyperparameter tuning, we take 20% of the data from the training split as development data. We choose the hyperparameter values that maximize the F1 score on the development data for a Decision Tree on the improved feature-rankings that the fine-tuning network produces. As candidate values for the number of dimensions of the vector spaces we used $\{50, 100, 200\}$. The number of directions to be used as input to the clustering algorithm was chosen from $\{500, 1000, 2000\}$. The number of clusters was chosen from $\{k, 2k\}$, with k the chosen number of dimensions. For the hidden layer of the neural network, we fixed the number of dimensions as equal to the number of clusters. As the scoring metric for the dimensions, we considered accuracy, Kappa and NDCG. In all experiments, we used 300 epochs, a minibatch size of 200, and the tanh activation function for the hidden layer of the neural network. After some preliminary tests we found that in most cases the parameters for the network could be kept the same. In all experiments: 300 epochs, batch size 200 and tanh activation for the hidden layer. The hidden layer was kept the same size as the input space V_n. We train the network using AdaGrad [20], with default values, and the model was implemented in the Keras library.

For the cluster size, we follow work by Steven Schockaert[63] and use twice the amount of clusters as there are dimensions in the space.

To learn the decision trees, we use the scikit-learn implementation of CART, which allows us to limit the depth of the trees. setting the maximum depth to one, three, or not at all. We used

information gain as the attribute selection criterion. To mitigate the effects of class imbalance, the less frequent class was given a higher weight during training.

6.3.1 Results

Table 6.2 shows the results for the 20 newsgroups dataset, where we use FT to indicate the results with fine-tuning[2]. We can see that the fine-tuning method consistently improves the performance of the depth-1 and depth-3 trees, often in a very substantial way. After fine-tuning, the results are also consistently better than those of LDA. For the unbounded trees (DN), the differences are small and fine-tuning sometimes even makes the results worse. This can be explained by the fact that the fine-tuning method specializes the space towards the selected features, which means that some of the structure of the initial space will be distorted. Unbounded decision trees are far less sensitive to the quality of the directions, and can even perform reasonably on random directions. Interestingly, depth-1 trees achieved the best overall performance, with depth-3 trees and especially unbounded trees overfitting. Since MDS and AWV perform best, we have only considered these two representations (along with LDA) for the remaining datasets, except for the IMDB Sentiment dataset, which is too large for using MDS.

The results for the movies and place-types datasets are shown in Table 6.3. For the MDS representations, the fine-tuning method again consistently improved the results for D1 and D3 trees. For the AWV representations, the fine-tuning method was also effective in most cases, although there are a few exceptions. What is noticeable is that for movie genres, the improvement is substantial, which reflects the fact that genres are a salient property of movies. For example, the decision tree for the genre 'Horror' could use the feature direction for $\{gore, gory, horror, gruesome\}$. Some of the other datasets refer to more specialized properties, and the performance of our method then depends on whether it has identified features that relate to these properties. It can be expected that a supervised variant of this method would perform consistently better in such cases. After fine-tuning, the MDS based representation outperforms LDA on the movies dataset, but not for the place-types. This is a consequence of the fact that some of the place-type categories refer to very particular properties, such as geological phenomena, which may not

[2]Since the main purpose of this first experiment was to see whether fine-tuning improved consistently across a broad set of representations, here we considered a slightly reduced pool of parameter values for hyperparameter tuning.

be particularly dominant among the Flickr tags that were used to generate the spaces. In such cases, using a BoW based representation may be more suitable.

The results for IMDB Sentiment are shown in Table 6.4. In this case, the fine-tuning method fails to make meaningful improvements, and in some cases actually leads to worse results. This can be explained from the fact that the feature directions which were found for this space are themes and properties, rather than aspects of binary sentiment evaluation. The fine-tuning method aims to improve the representation of these properties, possibly at the expense of other aspects.

6.4 Conclusions

We have introduced a method to identify and model the salient features from a given domain as directions in a semantic space. Our method is based on the observation that there is a trade-off between accurately modelling similarity in a vector space, and faithfully modelling features as directions. In particular, we introduced a post-processing step, modifying the initial semantic space, which allows us to find higher-quality directions. We provided qualitative examples that illustrate the effect of this fine-tuning step, and quantitatively evaluated its performance in a number of different domains, and for different types of semantic space representations. We found that after fine-tuning, the feature directions model the objects in a more meaningful way. This was shown in terms of an improved performance of low-depth decision trees in natural categorization tasks. However, we also found that when the considered categories are too specialized, the fine-tuning method was less effective, and in some cases even led to a slight deterioration of the results. We speculate that performance could be improved for such categories by integrating domain knowledge into the fine-tuning method.

Chapter 7

Conclusion

7.1 Introduction

This Chapter provides conclusions on how the work in this thesis contribute to the hypothesis and research questions.

7.2 Thesis Summary and Contributions

In Chapter 1 the Hypothesis was introduced. It was as follows:

> Vector space representations of documents can be re-organized such that their dimensions correspond to clear semantic features. Furthermore, these disentangled representations can be used to provide a useful inductive bias to classifiers, and as the features are disentangled, they can be used to qualitatively investigate the hidden layers of neural networks to gain valuable insights into how they represent documents.

In Chapter 4, disentangled feature representations obtained using the methods in this work performed better than the original unsupervised representations when used as input to low-depth decision trees. These results showed their potential for providing an inductive bias, as the classifiers in some cases performed better than linear SVM's trained on all features of the original unsupervised embedding. This shows that the feature representation obtained by the method resulted in disentangled and semantic features. The qualitative investigation found that the features are clear and meaningful, and these features were used to determine the difference

between document embeddings and parameters of the method. The NDCG scoring metric introduced in this work was found to generally be a good choice, in particular because it had the most consistent performance when features obtained by scoring directions with it were used as input to low-depth decision trees. Using features obtained from cluster-directions that k-means produced also resulted in better performance than the Derrac variation on low-depth decision trees.

In Chapter 5 feed-forward and auto-encoder neural networks were qualitatively investigated. Characteristics of feed-forward networks that used BOW as input were identified and compared with unsupervised representations, in particular these networks were found to have better representations for features that were relevant to the task, and had noisy terms scored less. It was found that neural networks that used the original unsupervised embeddings as input retained the noise that was present in that representation in addition to retaining features that were relevant to the task, however they had few new features. Auto encoders with increasingly low-dimensional embeddings were qualitatively investigated, and seemed to contain increasingly abstract properties. Finally, a rule-based classifier that connected the features from layers together was investigated in application to producing a domain theory. Although the results were preliminary for the auto-encoder, they lay a methodological foundation for connecting together the layers of neural networks.

In Chapter 6 a methodology was introduced to fine-tune the original document embedding that the feature representation was obtained from. This was done in-order to improve the directions, and in-turn improve the associated rankings. It was found to improve the scores of the feature representations when used as input to low-depth decision trees. Additionally, a qualitative investigation showed that the entities were more intuitive and specific than before, and that noise had been eliminated.

7.3 Research Questions

In the introduction, three research questions were asked. This section repeats those research questions and goes-over how they were answered in the thesis.

Question 1: Can directions be identified that correspond to clear semantic features in an un-

supervised way, and how can they be used to achieve good disentangled representations and across a wide range of different types of document embeddings, of documents across a range of domains and from a variety of vector space embeddings?

In Chapter 4 it was found that feature representations derived from unsupervised vector spaces contained clear semantic features, resulting in good disentangled representations. Chapter 5 also provided some quantitative analysis of disentangled representations obtained from the hidden layers of neural networks, finding that their performance when used as input to low-depth decision treees was close to, matched or exceeded the performance of the original network on the task. In Chapter 6 the associated rankings of these features was made more clear and their performance on low-depth decision trees was improved, resulting in an overall better disentangled representation using an entirely unsupervised method.

Question 2: To what extent can these directions and associated disentangled representations be used to gain qualitative insights into the characteristics of different neural networks?

Although the investigation of neural network hidden layer representations in Chapter 5 was qualitative, it was clear that the properties derived from these representations was related to how well they classified the task. Some characteristics of feed-forward networks were identified, in particular that they are representing terms related to the task better than the unsupervised representations, and that they reduced noise. Additionally, increasingly low-dimensional auto-encoders were found to contain increasingly abstract properties.

Question 3: Is it possible to obtain higher-quality disentangled representations by fine-tuning the initial vector space, while remaining in an unsupervised setting?

In Chapter 6 an unsupervised methodology was introduced that fine-tuned the initial vector space, resulting in higher performance of low-depth decision trees when the improved disentangled representations were used as input. Additionally, qualitative results showed that the entities had improved.

7.4 Future Work

This thesis has investigated the disentanglement of the representations, but it has not experimentally validated how interpretable the labels of the features are. To do so, the intruder task

from topic models could be used. Each group of words in a label (either the cluster words or the single term word and the words associated with its most similar directions) is given an intruder word from another cluster. The feature representation performs well if users on e.g. Amazon Turk are able to identify the intruder word. Another interesting test would be to use a domain, e.g. text in medicine, where some text classification task is required and verify that the decision trees make sense to expert users like doctors.

Although decision trees are interpretable, they may not make sense to end-users. This is especially pertinent when dealing with complicated concepts like ranking entities on a feature. One avenue for future work that will benefit many other applications of this method is converting low-depth decision trees that use disentangled features into human-readable explanations that can be understood by users in the domain. Although this does seem simply on the surface (e.g. The explanation could just be of the form IF Movie has "Horror", and it is "Romantic", then it is a Romantic Horror movie) it can be a bit more complicated when using features that do not have such a natural intuition or classifying things that cannot be put into such a simple format (e.g. Compare "IF Blood AND Gun THEN Mobsters" to IF There is "Blood" in the movie, and there are many "Gun" scenes, then the movie is likely about "Mobsters"). These kind of natural language explanations could result in better adoption of the method in real-world domains.

This work also has potential applications in the future to eXplainable AI (XAI), methods that explain existing learned neural network models. Essentially, the approach outlined in 5 would be used to create a "proxy model" that approximates the same behaviour that the neural network does in a simple way. Then, the quality of this proxy model is determined by how well it matches the predictions that the neural network made. Following the previous idea about transforming decision trees into natural language explanations, the method could be applied in a variety of domains to explain existing models to end-users.

Models learned from disentangled feature representations derived from neural network hidden layers could also act as interpretable alternatives to them. This brings up a primary question that follow-up work could address, can neural networks that achieve state-of-the-art e.g. BERT [35]. BERT is a neural network that achieves strong results on a variety of tasks and has entered mainstream application usage, but can it be disentangled into interpretable features, and can an interpretable classifier be learned that matches the neural network performance? This is but one of a variety of potential state-of-the-art models that could be disentangled to produce an

alternative interpretable model.

In Chapter 4 the linear SVM's that obtained hyper-planes classified words on a binary occurrence task, i.e. if the word occurred or if it did not occur in the document. Following this, the orthogonal direction was taken and entities were ranked on it using the dot product. It would be interesting to instead try an SVM like RankSVM [44] to directly learn rankings of documents on words, where the "true ranking" is determined by the Positive Pointwise Mutual Information (PPMI) score. This would potentially result in a more accurate ranking derived from the spatial representation for the word.

In Chapter 4 a potential problem with clustering algorithms adding words to clusters that disrupted their meaning, rather than keeping them as an important feature in the domain was identified. It would be interesting for a qualitative investigation into this phenomenon, seeing if manual removal of words from clusters would indeed benefit their performance in low-depth decision trees. If this is the case, there is potential for a new clustering method that performs this process automatically. Theoretically this could be done by taking a bag of PPMI scores as in Chapter

The fine-tuning process outlined in Chapter 6 originally followed the results demonstrating direction scores went down in when obtained from neural network hidden layers as they became more stacked. Given this, the idea was formed that it would be possible to fine-tune these spaces such that they can retain the semantic features in the first layer while also adjusting the representation according to the objective. However, in preliminary tests this did not yield such a result, instead by fine-tuning the representation it halted learning according to the objective. This preliminary investigation was done by taking each resulting representation, fine-tuning it, and then using it as input to the next auto-encoder. There is potential future work in pursuing this idea in application to explainability and interpretability. In particular, work in investigating if neural networks can be forced to retain disentanglement during learning. If this is the case, it may be possible to make neural networks interpretable.

7.5 Summary

In summary, this thesis demonstrated that disentangled feature representations can be obtained from a variety of document embeddings, and a qualitative analysis was conducted on these

representations. It was found that the features were clear and meaningful. Additionally, a method was introduced that fine-tunes vector spaces to improve these disentangled features. The methods seem promising in application to interpretability and explainability and offer much potential for future work.

Bibliography

[1] Maryam Mohammed Aldarwish and Hafiz Farooq Ahmad. Predicting Depression Levels Using Social Media Posts. *Proceedings - 2017 IEEE 13th International Symposium on Autonomous Decentralized Systems, ISADS 2017*, pages 277–280, 2017.

[2] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete Problems in AI Safety. pages 1–29, 2016.

[3] Robert Andrews, Joachim Diederich, and Alan B. Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 8(6):373–389, 1995.

[4] Martín Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[5] David Arthur and Sergei Vassilvitskii. K-means++: the advantages of careful seeding. In *In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007.

[6] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[7] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2012.

[8] Steven Bird, Ewan Klein, Edward Loper, Steven Bird, Ewan Klein, Edward Loper, Copyright Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*.

[9] David M. Blei and John D. Lafferty. Correlated Topic Models. *Advances in Neural Information Processing Systems 18*, pages 147–154, 2006.

[10] David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003, 2003.

[11] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. 2015.

[12] John A. Bullinaria and Joseph P. Levy. Extracting semantic representations from word co-occurrence statistics: Stop-lists, stemming, and SVD. *Behavior Research Methods*, 44(3):890–907, 2012.

[13] Grégoire Burel and Harith Alani. Crisis Event Extraction Service (CREES) â Automatic Detection and Classification of Crisis-related Content on Social Media. (May), 2018.

[14] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 1:1870–1879, 2017.

[15] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. 2016.

[16] William Cohen. Fast effective rule induction. *Twelfth International Conference on Machine Learning: 1995*, 1995.

[17] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian LDA for topic models with word embeddings. In *Proc. ACL*, pages 795–804, 2015.

[18] J. Derrac and S. Schockaert. Inducing semantic relations from conceptual spaces: a data-driven approach to plausible reasoning. *Artificial Intelligence*, pages 74–105, 2015.

[19] Joaqu??n Derrac and Steven Schockaert. Inducing semantic relations from conceptual spaces: A data-driven approach to plausible reasoning. *Artificial Intelligence*, 228:66–94, 2015.

[20] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[21] Manaal Faruqui and Chris Dyer. Non-distributional Word Vector Representations. *Acl-2015*, pages 464–469, 2015.

[22] Peter Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. MIT press, 2004.

[23] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. *Proceedings - 2018 IEEE 5th International Conference on Data Science and Advanced Analytics, DSAA 2018*, pages 80–89, 2019.

[24] Yanina Gimenez and Guido Giussani. Searching for the core variables in principal components analysis. *Brazilian Journal of Probability and Statistics*, 32(4):730–754, 2018.

[25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. *Advances in Neural Information Processing Systems 27*, pages 2672–2680, 2014.

[26] Bryce Goodman and Seth Flaxman. EU regulations on algorithmic decision-making and a "right to explanation". *2016 ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)*, (Whi):26–30, 2016.

[27] Ulrike Hahn and Nick Chater. Similarity and rules: distinct? exhaustive? empirically distinguishable? *Cognition*, 65:197 – 230, 1998.

[28] Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1367–1377, 2016.

[29] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing Deep Neural Networks with Logic Rules. *arXiv preprint*, pages 1–18, 2016.

[30] H.˜Zou, T.˜Hastie, R.˜Tibshirani, Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.

[31] Sarthak Jain, Edward Banner, Jan-Willem van de Meent, Iain J. Marshall, and Byron 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*, pages 4683–4693, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[32] Shoaib Jameel, Zied Bouraoui, and Steven Schockaert. MEmbER : Max-Margin Based Embeddings for Entity Retrieval.

[33] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[34] Vineet John and Olga Vechtomova. Disentangled Representation Learning for Non-Parallel Text Style Transfer. 2019.

[35] Ming-wei Chang Kenton, Lee Kristina, and Jacob Devlin. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (Mlm), 1953.

[36] D Kim and J Lee. Handling continuous-valued attributes in decision tree with neural network modeling. *Machine Learning: Ecml 2000*, 1810:211–219, 2000.

[37] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon, and Lee Jiwon. Learning to Discover Cross-Domain Relations with Generative Adversarial Networks.

[38] Aykut Koc, Lutfi Kerem Senel, Ihsan Utlu, and Haldun M Ozaktas. Imparting Interpretability to Word Embeddings while Preserving Semantic Structure. pages 1–11.

[39] Adriana Kovashka, Devi Parikh, and Kristen Grauman. WhittleSearch : Image Search with Relative Attribute Feedback.

[40] Suyash Lakhotia and Xavier Bresson. An Experimental Comparison of Text Classification Techniques. *2018 International Conference on Cyberworlds (CW)*, pages 58–65, 2018.

[41] Maria Larsson, Amanda Nilsson, and Mikael Kågebäck. Disentangled Representations for Manipulation of Sentiment in Text. (2015), 2017.

[42] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. *31st International Conference on Machine Learning, ICML 2014*, 4:2931–2939, 2014.

[43] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning*, pages 1188–1196, 2014.

[44] Ching Pei Lee and Chih Jen Lin. Large-scale linear rankSVM. *Neural Computation*, 26(4):781–817, 2014.

[45] David D Lewis. Natural language processing for information retrieval 1 Abstract 2 Introduction 3 Document retrieval. pages 1–22, 1993.

[46] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Topical word embeddings. In *Proc. AAAI*, pages 2418–2424, 2015.

[47] Hongyin Luo, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. Online Learning of Interpretable Word Embeddings. (September):1687–1692, 2015.

[48] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. pages 142–150, 2011.

[49] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L. Yuille. Explain Images with Multimodal Recurrent Neural Networks. *arXiv:1410.1090 [cs]*, pages 1–9, 2014.

[50] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A Survey on Bias and Fairness in Machine Learning. 2019.

[51] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? *35th International Conference on Machine Learning, ICML 2018*, 8:5589–5626, 2018.

[52] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. *Nips*, pages 1–9, 2013.

[53] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. Large-scale multi-label text classification - Revisiting neural networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8725 LNAI(PART 2):437–452, 2014.

[54] Brooks Paige, Dana H Brooks, and Jennifer Dy. Structured Disentangled Representations. 2016.

[55] Hamid Palangi, Paul Smolensky, Xiaodong He, and Li Deng. Question-answering with grammatically-interpretable representations. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 5350–5357, 2018.

[56] Sungjoon Park, Jin Yeong Bak, and Alice Oh. Rotated word vector representations and their interpretability. *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 401–411, 2017.

[57] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[58] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-Agnostic Interpretability of Machine Learning. (Whi), 2016.

[59] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. âwhy should i trust you?â: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD â16, page 1135â1144, New York, NY, USA, 2016. Association for Computing Machinery.

[60] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, and Phil Blunsom. Reasoning about Entailment with Neural Attention. (2015):1–9, 2015.

[61] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, UAI '04, pages 487–494, Arlington, Virginia, United States, 2004. AUAI Press.

[62] Emad W. Saad and Donald C. Wunsch. Neural network explanation using inversion. *Neural Networks*, 20(1):78–93, 2007.

[63] Steven Schockaert and Jae Hee Lee. Qualitative reasoning about directions in semantic spaces. In Qiang Yang and Michael Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 3207–3213. AAAI Press, 2015.

[64] Rudy Setiono, Bart Baesens, and Christophe Mues. Recursive neural network rule extraction for data with mixed attributes. *IEEE Transactions on Neural Networks*, 19(2):299–307, 2008.

[65] Rudy Setiono and Huan Liu. Neurolinear: From neural networks to oblique decision rules. *Neurocomputing*, 17(1):1–24, 1997.

[66] Vin De Silva and Joshua B Tenenbaum. Sparse multidimensional scaling using landmark points. pages 1–41, 2004.

[67] Anant Subramanian, Danish Pruthi, Harsh Jhamtani, Taylor Berg-kirkpatrick, and Eduard Hovy. SPINE : SParse Interpretable Neural Embeddings. pages 4921–4928.

[68] Matt Taddy. Document Classification by Inversion of Distributed Language Representations. *Proceedings of the 53rd meeting of the Association for Computational Linquistics (ACL'15)*, pages 45–49, 2015.

[69] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Sharing clusters among related groups: Hierarchical dirichlet processes. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, pages 1385–1392, Cambridge, MA, USA, 2004. MIT Press.

[70] Valentin Trifonov, Octavian-eugen Ganea, Z Eth, Anna Potapenko, Thomas Hofmann, and Z Eth. Learning and Evaluating Sparse Interpretable Sentence Embeddings. pages 200–210, 2018.

[71] Valentin Trifonov, Octavian-eugen Ganea, Anna Potapenko, Thomas Hofmann, and Z Eth. Learning and Evaluating Sparse Interpretable Sentence Embeddings.

[72] Amos Tversky. Features of similarity. *Psychological review*, 84:327–352, 1977.

[73] Berk Ustun and Cynthia Rudin. Methods and Models for Interpretable Linear Classification. *arXiv*, pages 1–57, 2014.

[74] Paolo Viappiani. Preference-based Search using Example-Critiquing with Suggestions. 27:465–503, 2006.

[75] Jesse Vig, Shilad Sen, and John Riedl. The tag genome: Encoding community knowledge to support novel interaction. *ACM Transactions on Interactive Intelligent Systems*, 2(3), 2012.

[76] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[77] Xuerui Wang and Andrew McCallum. Topics over time: a non-markov continuous-time model of topical trends. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433. ACM, 2006.

[78] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google ' s Neural Machine Translation System : Bridging the Gap between Human and Machine Translation. pages 1–23.

[79] Youwei Zhang and Laurent E. Ghaoui. Large-scale sparse principal component analysis with application to text data. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 532–539. Curran Associates, Inc., 2011.

[80] Jan Ruben Zilke, Eneldo Loza Mencía, and Frederik Janssen. DeepRED â Rule extraction from deep neural networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9956 LNAI:457–473, 2016.