

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

July 2021

**Cardiff University
School of Computer Science & Informatics**

**To my parents.
for their invaluable and unwavering support.**

Abstract

In contrast to traditional document representations such as bags-of-words, the kind of 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. However, this also comes at an important cost, namely that the features of the representation are entangled, i.e. each feature is not individually meaningful. The main aim of this thesis is to address this problem by disentangling vector spaces into representations that are composed of meaningful features that are closely aligned with natural categories from the given domain. For instance, in the domain of IMDB movie reviews, where each document is a review, a 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 and Schockaert [21], which derives features from low-dimensional vector spaces. The method begins by using a linear classifier to find a hyper-plane that separates documents that have a term from those that do not have a term. Then, from each hyperplane, the direction of the orthogonal vector is taken to induce a ranking from documents that are least related to the word (those furthest from the hyper-plane on the negative side) to documents that are most related to it (those furthest from the hyper-plane on the positive side). To identify which of these words describe semantically important features, they are scored by how well the linear classifier performs on a standard classification metric, which approximates how linearly separable the documents are that contain the term in the vector space. The assumption is that the more separable a term is, the better modelled it is in the space. The highest scoring terms are selected to be used as features, and documents are ranked by calculating the dot product between the orthogonal vector to the hyper-plane and each document vector. This results in a ranking of documents on how strongly expressed each

feature is, e.g. movies could be ranked on how "Scary" they are. Only the direction of this orthogonal vector is considered in this work, as our concern is to obtain document rankings. The work by Derrac and Schockaert [21] obtained semantic features from Multi-Dimensional Scaling (MDS) document embeddings and validated their work by classifying documents 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."

This work by Derrac and Schockaert [21] was focused on showing the feasibility of learning disentangled representations, but it did not make clear which components of their method were essential. The first main contribution of this thesis therefore consists in a thorough investigation of variants of their method, where a quantitative analysis is conducted of different document representations (as opposed to only MDS), different term scoring functions (as opposed to only the Kappa score) and the proposed clustering method is revisited. This extensive evaluation is across a variety of new domains, and compares the method to stronger baselines. To quantitatively analyse the impact of these design choices, the use of low-depth decision trees that classify natural categories in the domain is proposed. A qualitative analysis of the discovered features is also presented.

Neural network architectures have advanced to state-of-the-art in many tasks. The second main contribution of the thesis follows the idea that the hidden layers of a neural network can be viewed as vector space embeddings. Specifically, in our setting, meaningful feature to describe documents can be derived from the hidden layers of neural networks. In particular, to test the potential of using neural networks to discover features that cannot be discovered using standard document embedding methods, feed-forward neural networks and stacked auto-encoders are quantitatively and qualitatively investigated. Auto-encoders are stacked by using their hidden layer as the input to another auto-encoder. We find that meaningful features can indeed be derived from the hidden layers of the considered neural network architectures. We quantitatively assess how predictive these features are, compared to those of the input embeddings. Qualitatively, we find that feedforward networks tend to select and refine features that were already modelled in the input embedding. In contrast, stacked autoencoders tend to model increasingly more abstract features as additional hidden layers are added. For example, in the initial auto-encoder layers, features like "Horror" and "Comedy" can be separated well by the linear classifier. Meanwhile, features like "Society" and "Relationships" are more separable in

later layers. 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 can clarify the nature of the transformations that are learned by the neural network, for example:

IF Emotions AND Journey THEN Adventure (1)

The third contribution of this thesis is the introduction of an additional post-processing step for improving disentangled feature representations. This is done by fine-tuning the original embedding such that rankings of documents induced by our disentangled features are in agreement with rankings induced by Pointwise Mutual Information scores. The motivation for this contribution stems from the fact that 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. This method is also unsupervised, requiring only a bag-of-words representation of the documents as input. In particular, clusters of terms are identified that refer to semantically meaningful and important features of the considered domain, and a simple neural network model is used to learn a new representation in which each of these features is more faithfully modelled as a direction. It is found that in most cases this method improves the ranking of documents, and results in increased performance when disentangled feature representations are used as input to classifiers.

Overall, this thesis quantitatively and qualitatively confirms 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.

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.

Finally, my deepest thanks go to my parents and guides who were able to support me through the final years of my PhD.

Contents

Abstract	iii
Acknowledgements	vi
Contents	vii
List of Publications	xii
List of Figures	xiii
List of Tables	xv
1 Introduction	1
1.1 Context	1
1.2 Hypothesis	5
1.3 Research Questions	6
1.4 Contributions	6
1.5 Thesis Structure	8
1.6 Summary	9

2	Background	10
2.1	Introduction	10
2.2	Text Data	11
2.2.1	Text Domains	11
2.2.2	Preprocessing Text Data	12
2.3	Representations	13
2.3.1	Bag-of-Words	14
2.4	Text Document Classification	17
2.4.1	Overfitting	17
2.4.2	Evaluation Metrics	18
2.4.3	Types of Classification Problems	19
2.5	Classifiers	19
2.5.1	Decision Trees	19
2.5.2	Linear Support Vector Machines	20
2.5.3	Neural Networks	20
2.6	Low-dimensional Vector Spaces	21
2.6.1	Principal Component Analysis	22
2.6.2	Multi-Dimensional Scaling	23
2.6.3	Word Embeddings	23
2.6.4	Doc2Vec	24
2.7	Interpretable Representations	24
2.7.1	Conceptual Spaces	24
2.7.2	Topic Models	25
2.7.3	Generative Adversarial Network and Variational Autoencoders	26

2.7.4	Sparse Representations	26
2.8	Conclusions	27
3	Datasets and Semantic Spaces	29
3.1	Introduction	29
3.2	Datasets	29
3.3	Technical Details	32
3.4	Representations	33
4	Disentangling Unsupervised Document Embeddings	36
4.1	Introduction	36
4.2	Method	40
4.2.1	Obtaining Directions and Rankings From Words	40
4.2.2	Filtering Word Directions	42
4.2.3	Clustering Features	45
4.3	Qualitative Results	48
4.3.1	The Highest Scoring Directions for each Domain	48
4.3.2	Comparing Document Embedding Models	49
4.3.3	Comparing Scoring Metrics	51
4.4	Quantitative Results	55
4.4.1	Evaluation Method	56
4.4.2	Hyper-Parameters	56
4.4.3	Summary of all Results	60
4.4.4	Initial Document Embedding Vectors	61
4.4.5	Single-Term Features	63

4.4.6	Clustered Directions	66
4.5	Conclusion	71
5	Disentangling Neural Network Layers	73
5.1	Introduction	73
5.2	Background	75
5.3	Method	76
5.3.1	Stacked Denoising Auto-encoders	76
5.3.2	Feed-forward networks	77
5.4	Investigating Feed-forward Neural Networks	79
5.4.1	Parameters	79
5.4.2	Quantitative Evaluation Of Feed-forward Networks	80
5.4.3	Qualitative Investigation of Feed-forward Networks	83
5.4.4	Summary of Results	96
5.5	Evaluation of Auto-encoders	99
5.5.1	Software, Architecture and Settings	99
5.5.2	Quantitative Evaluation of Cluster Representation	100
5.5.3	Qualitative Evaluation of Cluster Representations	101
5.5.4	Quantitative Frequency Evaluation	101
5.6	Inducing Rules from Auto-Encoder Entity Embeddings	103
5.6.1	Qualitative Evaluation of Induced Symbolic Rules	104
5.7	Conclusion	107

6	Modelling Semantic Features in Fine-Tuned Vector Spaces	108
6.1	Introduction	108
6.2	Fine-Tuning Vector Spaces and their Associated Feature Directions	111
6.2.1	Generating Target Rankings	112
6.2.2	Generating Target Feature Values	112
6.2.3	Fine-Tuning	113
6.3	Quantitative Evaluation	114
6.3.1	Results	117
6.4	Conclusions	118
7	Conclusion	119
7.1	Introduction	119
7.2	Thesis Summary and Contributions	119
7.3	Research Questions	120
7.4	Future Work	121
7.5	Summary	123
	Bibliography	125

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

- 1.1 An example of a hyperplane in a toy domain of shapes. The hyperplane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples. The arrow along the bottom of the figure indicates the corresponding direction. Those shapes far from the hyperplane on the right (positively classified) side of the direction are more square than those in the opposite direction 3
- 1.2 An example of a Decision Tree classifying if a movie is in the "Sports" genre. Here, decision tree nodes use features that capture properties of the domain. These properties are represented by keywords or clusters of keywords. "Coach Sports Team Sport Football" is a cluster of keywords, that refers to the property in the domain of sports and sport, described using those words and related words like coaches and teams. This property is extremely useful for classifying if the movie is in the "Sport" genre, and as such is the first node of the tree. Other nodes, like "Virus" apply to only a very small number of samples 4
- 1.3 Toy example showing the effect of outliers in a two-dimensional embedding of geometric shapes 8
- 4.1 An example of a hyperplane in a toy domain of shapes. The hyperplane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples. Those closest to the hyperplane are less square than those further away 37

4.2	Another example of a hyperplane in a toy domain of shapes. Here we show multiple directions, one for light and one for square. The hyperplane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples for the word square	41
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)	57
6.1	Toy example showing the effect of outliers in a two-dimensional embedding of geometric shapes	109

List of Tables

3.1	Text examples from three domains. For the Movies and Place-Type domains, the original text was not available	30
4.1	Example features from three different domains, where each cluster of words corresponds to a cluster direction	39
4.2	The top-scoring words for each domain	50
4.3	Unique terms between document embedding models	52
4.4	Terms unique to different scoring metrics in the Movies domain	53
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	55
4.6	The F1-scores of trees limited to depth-one, two and three, for the best hyperparameter variation of each representation	62
4.7	Best results for the Newsgroups for a variety of classifiers when using the document embeddings as input.	64
4.8	Best results for all domains for a variety of classifiers when using the document embeddings as input.	65
4.9	The best results when the single-term disentangled feature representation is used as input to low-depth decision trees	67
4.10	All clustering size results for the Newsgroups	69

4.11	The best results when using the documents ranked on the directions output by the clustering method for the optimal number of clusters for each domain and task	70
5.1	F1-scores for each embedding type and domain	81
5.2	Top-scoring directions measured using NDCG score	85
5.3	Terms from three different document embeddings, the initial embedding, the neural network that used a bag-of-words as input and the neural network that used the unsupervised vector space as input. Arranged by NDCG, from highest to lowest	87
5.4	The largest differences in term scores between embeddings for the Movies domain. 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 . .	88
5.5	The top node of the Decision Tree for the Newsgroups task	90
5.6	The top node of the Decision Tree for Movies	92
5.7	The Place-Types clusters used at the top of the decision tree and the associated top-ranked entities	95
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 initial embedding	97
5.9	The results of depth-3 trees for a variety of tasks in the movie domain where cluster representations derived from auto-encoder layers are used as input . . .	100
5.10	A comparison between the first layers and the fourth layers of two different kinds of auto-encoders	102
5.11	The average frequency of cluster words in the corpus. Here, the frequencies of the first words in the cluster are shown	103
6.1	Comparing the highest ranking Place-Type objects in the original and fine-tuned space	110

6.2	Results for 20 Newsgroups.	114
6.3	The results for Movie Reviews and Place-Types on depth-1, depth-3 and un- bounded trees	115
6.4	Results for IMDB Sentiment.	115

Introduction

1.1 Context

In recent years, machine learning models have achieved state-of-the-art results on most natural language processing tasks¹ such as machine translation, [93] question answering [13], and document classification [94]. Although these methods perform well, they are typically not interpretable. As machine learning has extended into real-world domains like medicine and policing, concerns of safety [1] and fairness [58] have brought the legality and risk of implementing systems that are not interpretable to the attention of lawmakers. The European Union introduced a legal "Right to explanation" in 2018 [31], requiring that machine learning models must be able to explain why they have made a decision about a person.

Simple interpretable models are one solution to this problem. A prototypical example of a simple interpretable model is a low-depth decision tree. This low-depth decision tree can be used as a classifier model, for example in the Natural Language Processing (NLP) task of document classification, by assigning text documents to distinct classes. In order to use text documents as input to a low-depth decision tree, they must be first processed into a suitable representation. One standard way to obtain a suitable representation of text documents is by using the bag-of-words model. The bag-of-words model takes the form of a matrix where each row corresponds to a text document and each column corresponds to a word in the corpus. The value of a word for a document would be the frequency of the word in that document, as a standard example.

One domain investigated in this thesis is the 20 Newsgroups domain, where forum topics are

¹<https://github.com/sebastianruder/NLP-progress>

text documents. These forum topics are composed of user text posts, and each topic is submitted to separate categories. In this domain, one document classification task is to assign documents (e.g. forum topics) to the categories they originated from (e.g. automobiles, computer-science). In order for a low-depth decision tree to perform well on this task, the features must capture properties in the domain related to these categories. As an extreme example, in a decision tree limited to a depth of one, a single feature is used. For the decision tree classifier to perform well, e.g. by classifying if a topic is from the "automobile" category, the feature used must capture a property that is highly related to "automobiles". This feature would have a high value if the text document (i.e. topic) is highly related to automobiles, and the feature would have low value if the document is not related to automobiles.

In this task, the bag-of-words representation can represent a variety of words that are relevant only to automobiles, e.g. "car", "diesel", "volkswagen", or even "automobile". However, the bag-of-words is necessarily sparse, and requires a deep tree that can include many relevant features for good classification performance. As an example, in a one-depth decision tree, making a classification decision based on the frequency of the word "automobile" will likely identify topics that are related to automobiles, but will also miss many posts that are related to automobiles but do not contain the word directly.

A bag-of-words can be used as input to a dimensionality reduction method to be obtain a vector space. After this process, each document is represented as a vector, where the number of dimensions is far lower than the number of features in the bag-of-words representation. In this space, similar documents are represented using similar vectors. As an example, in the domain of newsgroup postings, topics that are similar, for instance in content or writing style, would be represented using similar vectors. However, the features (i.e. dimensions) of a vector space are not typically suitable for a simple interpretable classifier, as they do not usually correspond directly to properties of the domain. Previous work by Derrac and Schockaert [21] introduce a variety of ways to obtain properties from these low-dimensional vector space document embeddings, and derive semantic features from these properties. These semantic features capture key domain properties, like "computer-science" in the 20 Newsgroups domain.

The process to obtain these semantic features from a low-dimensional vector space is as follows. First, words are identified that could be useful as semantic features. This is done by learning a linear classifier, e.g. a linear Support Vector Machine (SVM) for each word, using the vector

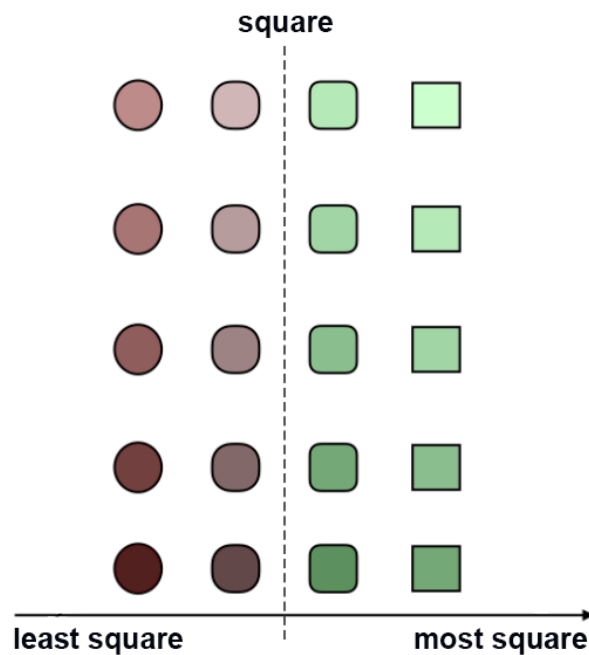


Figure 1.1: An example of a hyperplane in a toy domain of shapes. The hyperplane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples. The arrow along the bottom of the figure indicates the corresponding direction. Those shapes far from the hyperplane on the right (positively classified) side of the direction are more square than those in the opposite direction.

representations of documents as input. These Linear SVMs each obtain a hyperplane that separates documents where the word occurs from documents where the word did not occur. These words are then scored on a metric, e.g. how accurate the classifier is. The top-scoring words on that metric are assumed to be more likely to correspond to a property of the domain, as they are more separable in the space.

The direction of the normal of this hyperplane is then used to order documents, with those that are distant from the hyperplane on the negative side (those that are least representative of the word) as low rank, and those that are distant from the hyperplane on the positive side (those that are most representative of the word) as high rank. This ranking is obtained using the dot-product between the document vectors and the normal vector of the hyperplane, and this ranking of documents can be used as a feature. The orthogonal vector to the hyperplane is referred to as a direction in this work as our concern is to obtain this ranking of documents. An accompanying

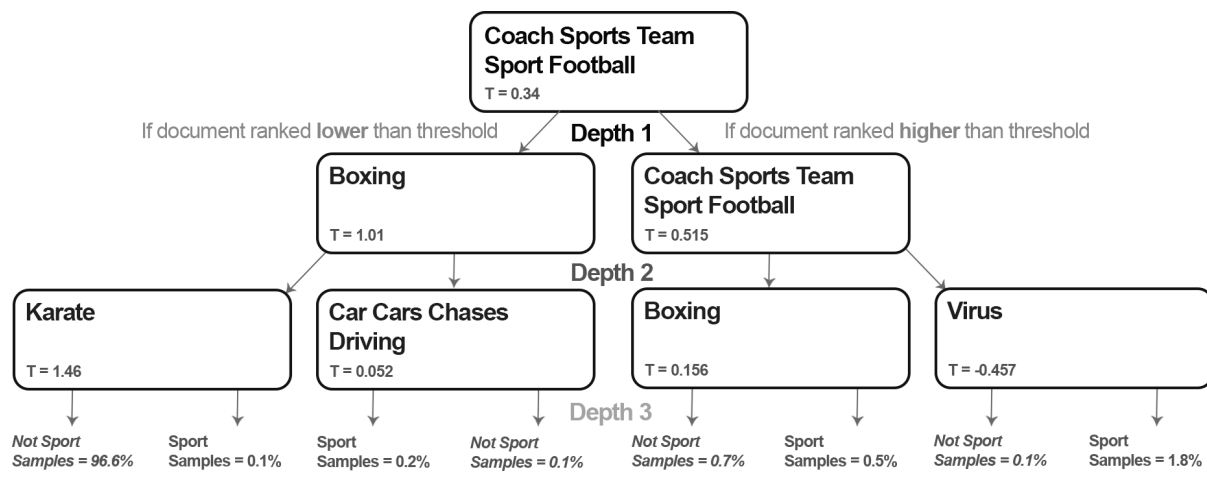


Figure 1.2: An example of a Decision Tree classifying if a movie is in the "Sports" genre. Here, decision tree nodes use features that capture properties of the domain. These properties are represented by keywords or clusters of keywords. "Coach Sports Team Sport Football" is a cluster of keywords, that refers to the property in the domain of sports and sport, described using those words and related words like coaches and teams. This property is extremely useful for classifying if the movie is in the "Sport" genre, and as such is the first node of the tree. Other nodes, like "Virus" apply to only a very small number of samples.

visual in a two-dimensional toy domain of shapes is shown in Figure 1.1. This example includes a direction of how "square" the toy shapes are.

These kind of directions in document embedding models, that go from documents that least represent a semantically meaningful word to those that most represent it, can be useful in a wide variety of applications. The most immediate example is being used as features in a simple interpretable classifier. In Figure 1.2, an example of a decision tree that uses the semantic features obtained in this thesis limited to a depth of three is shown. This example is from a domain where text documents are movies, represented by a concatenation of movie reviews. The task is to classify the genre of the movie, and this tree classifies if a movie is a sports movie using semantic features derived from a vector space embedding.

Another example is 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 [88]. For instance, Vig et al [89] 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 captured as a semantic feature, such critiques can be addressed in a straightforward way. Similarly, in [45] 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" [37]. While features such as popularity are typically not encoded in traditional knowledge bases, they can often be represented as document embedding model directions.

Although the semantic features derived from vector space embeddings do have labels, the intent of this thesis is not to optimize these labels for interpretability. Rather, the intention of this thesis is to develop methods that can transform given vector space representation, such that components of the resulting vectors correspond to semantically meaningful (i.e. predictive) properties from the considered domain. Such a representation is called "disentangled" - a representation composed of these properties is a disentangled representation.

This chapter continues as follows: In Section 1.2 the research hypothesis for the thesis is posed. This is followed by an introduction of three research questions in Section 1.3, each corresponding to a chapter in the thesis. The contributions of each chapter are then described in Section 1.4, and the structure of the thesis is laid out in Section 1.5.

1.2 Hypothesis

Semantically meaningful features can be obtained from vector space representations of documents. These features are sufficiently predictive to be useful for simple interpretable classifiers, of which low-depth decision trees are a prototypical example, allowing for a performance that is close to the performance of an unbounded decision tree using the same features as input. Further, these feature can be obtained from neural network hidden layers, and can provide valuable insights into aspects of the domain that the neural network has learned.

1.3 Research Questions

Question 1: Can meaningful semantic features be characterised as directions across a wide range of vector space encodings and domains, and do these semantic features allow us to learn effective low-depth decision trees?

Question 2: Can semantic features be obtained from the hidden layers of neural networks, and to what extent can these features be used to investigate the characteristics of different neural networks?

Question one and two focus on obtaining semantic features. However, this is not the intention of the final research question. Instead, this question looks at how to improve existing semantic features. In particular, by using a neural network to fine-tune the initial vector space embedding so that the directions contained within it are better suited for use as semantic features.

Question 3: Is it possible to obtain higher-quality semantic features in an unsupervised way by fine-tuning the initial vector space?

1.4 Contributions

In the work by Derrac and Schockaert [21], three domains were investigated with all results based on the same document embedding model. Then, these methods were evaluated with fixed hyper-parameters and a rule-based classifier. In Chapter 4 we propose a methodology for quantitatively validating the method and variants of the method by Derrac and Schockaert [21], by using low-depth Decision Trees, limited to a depth of one, two or three. An extensive quantitative and qualitative evaluation is performed with the method and variants of the method, using data from five different domains, and vector space embeddings derived from a variety of document embedding methods. This is important to confirm the generality of the results from [21].

In Chapter 5, the method from Chapter 4 is used to obtain semantic representations from the hidden layers of neural networks, in particular feed-forward networks and auto-encoders. Specifically, 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. Further, characteristics of feed-forward networks are identified, in particular that they represent new semantic features that are more relevant to the class. However, the same experiment repeated for the stacked, meaning multi-layered, auto-encoder yielded a negative result. Furthermore, the properties in a stacked denoising auto-encoder are quantitatively and qualitatively investigated, and a method is introduced to induce rules that describe relationships between semantic features from each layer, and the application of this method to better explain how the neural network represents information 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 centered objective used to build 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 in between the PPMI scores and the rankings of the documents. 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 documents to the target ranking. The intention is not to achieve 100% accuracy, but instead to rearrange the rankings so that similarity based information is de-prioritized if it allows us to learn more meaningful directions. Using these new properties as input to low-depth decision trees results in a performance increase for a variety of configurations and tasks, with some exceptions. A qualitative investigation shows that the document rankings become more specific and meaningful for the features. This work was published in The SIGNLL Conference on Computational Natural Language Learning (CoNLL) 2018.

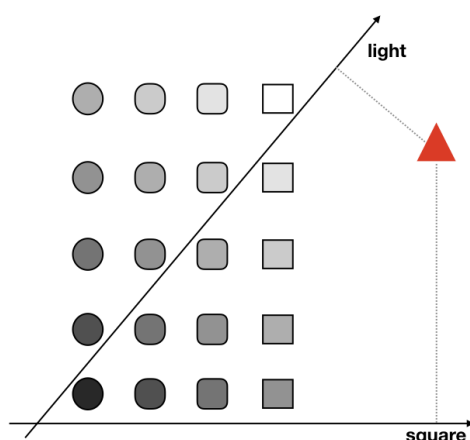


Figure 1.3: Toy example showing the effect of outliers in a two-dimensional embedding of geometric shapes.

1.5 Thesis Structure

- **Chapter 2** gives an overview of methods for processing unstructured text data, standard machine-learning classifiers, and it provides a background on other methods to discover semantic features and on related work in interpretability.
- **Chapter 3** introduces and explains the datasets used in this thesis, as well as explaining the hyper-parameters used for the machine-learning models in this thesis.
- **Chapter 4** provides a comprehensive quantitative and qualitative analysis of the method introduced by Derrac and Schockaert [21], across a much broader group of domains, tasks, and vector space embedding methods. This section introduces the use of low-depth decision trees as a mechanism for evaluating the quality of the learned semantic features. Additionally, for the sake of completeness, variations of the method are introduced, with scoring functions that were previously not used, such as NDCG, F1-score and accuracy. Standard K-means is used as an additional clustering algorithm that provides a reference point to the variation of K-means used in the original work. All variations, parameters, document embedding methods and domains are comprehensively analysed and compared.
- **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. A rule-based classifier is used to connect properties between layers of an auto-encoder.

- **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.

1.6 Summary

Vector space representations of text documents are common in natural language processing (NLP). However, their features are typically not interpretable. There is an increasing need for suitable solutions to interpretability in NLP and machine-learning. A prototypical example of a simple interpretable classifier is a low-depth decision tree. A low-depth decision tree is effective when semantic features are used as input. These features correspond to properties of the domain that are suitable for the class, e.g. A property that represents the property of a movie being "Bloody" when classifying if a movie is a "Horror". This thesis studies methods for inducing such semantic features from a variety of vector space encodings of documents, including neural networks.

Background

2.1 Introduction

In this chapter a general background for learning and evaluating low-dimensional text document embeddings is provided. For a more in-depth introduction to machine learning, see the book by Gareth James et al [38] or Bishop [7], and for a more in-depth introduction to natural language processing, see the books by Manning [56] and Jurafsky [43]. This introduction provides a general overview of how low-dimensional vector space embeddings of text documents can be obtained.

This thesis is concerned with obtaining a semantic representation of text data, that is aimed at modelling the main properties from a given domain. To evaluate these representations, we will rely on a number of key domain-specific classification tasks. These tasks rely on a classification model, which takes some vector input x and maps each such input to one of K discrete classes C_k where $k = (1, 2, \dots, K)$ [7]. Such tasks are known as document classification tasks. One example of a document classification task is e.g. classifying a movie review document x as either "negative" or "positive". A machine-learning model cannot process a document of raw text directly, hence the input x is typically a preprocessed representation of a document.

One popular and simple way to represent text documents is using the bag-of-words model [73]. Given a set of documents $D = (d_1, d_2, \dots, d_n)$ where n is the number of documents, and words $W = (w_1, w_2, \dots, w_m)$ in these documents D , where m is the number of words in all documents D , a bag-of-words representation uses a matrix M in which each document corresponds to a row vector and each word corresponds to a column vector. In other words, each document d_i is represented by the vector $d_i = (x_{i1}, \dots, x_{im})$, where the value x_{ij} reflects how much word w_j is

related to document d_i . For instance, a straightforward choice is to choose x_{ij} as the number of times $f(d_i, w_j)$ that word w_j occurs in document d_i .

Document embeddings are (typically) low-dimensional vector representations of documents. Such representations are often learned by taking the bag-of-words representation as input and applying a dimensionality reduction technique. This results in a representation where the spatial relationships between documents reflect their semantic relationships.

When documents describe particular entities of interest (e.g. movies), their embeddings thus capture semantic properties of these entities. The methods in this thesis aim to make these semantic properties explicit. In this chapter the methods to obtain vector space embeddings of text documents are explained. To begin, how the initial text documents are preprocessed is covered in Section 2.2. Then, how basic bag-of-words representations are obtained is explained in Section 2.3, followed by how vector space embeddings are obtained in Section 2.6. As machine-learning methods are used to evaluate these representations, Section 2.4 covers different machine-learning methods. Finally, a background on interpretable representations and classifiers is covered in Section 2.7.

2.2 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 20 Newsgroups domain (See Section 3.2), which is composed of online news discussion groups. In this thesis, we use a variety of datasets, each from different domains. Below an example post is provided from the Newsgroups domain that contains 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 I 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

We distinguish in this thesis between general text data across multiple domains, and domain-specific text data that has jargon and structure particular to the domain. In this thesis, representations are obtained from domain-specific text, and domain-specific tasks are used to test these representations. Machine-learning models to solve these tasks are trained on domain-specific data.

The datasets used in this thesis are composed of text documents. Some of these datasets of text documents used in this thesis have documents that correspond to entities, e.g. movies. The datasets are further explained in Section 3.2.

2.2.2 Preprocessing Text Data

One issue when using only domain-specific text data is that the model may place excess importance on structure or quirks of the domain rather than semantics, 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 not being generalizable. To address problems that can come with using domain data as-is, text data is preprocessed using rules from both the domain and in general. As an example of a general rule, so-called stop words are often removed. These are words that are highly frequent which typically convey very little or no semantic content (e.g.

the, he, it). As an example of a domain rule, in the example in Chapter 2.2.1 footer text would be removed. This pre-processing must strike a balance between removing information that is not relevant and keeping information that is relevant.

The first step to preprocessing data in order to build a representation like a bag-of-words is building a vocabulary composed of words from the corpus. In this vocabulary, it is important that words are identified regardless of grammatical differences e.g. identifying that "Dog" and "dog." both refer to the same word. The following is the methodology used in this thesis to obtain a vocabulary from text documents:

- 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" would have the infrequent word "dgo" removed.
- Domain-specific preprocessing 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 in at least two documents are removed. In the next section, we cover some methods for text representation and explain their basic advantages.

2.3 Representations

In this section, background for natural language processing techniques is introduced as necessary to produce representations of text documents. More details can be found in the book by

Jurafsky and Martin [43]. 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 $X = (x_1, x_2, \dots, x_n)$. Here, the components of this vector are called 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

Bag-of-words is a simple representation of text data that can scale to an extreme amount of data, but does not retain the order of words. The most standard Bag-of-words model (BOW) uses the number of occurrences in the document as the value for each word in the matrix. 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)".

Bag-of-words representations are encoded using a matrix, where rows correspond to documents and columns correspond to unique words in the corpus. This set of unique words across all documents in a domain is referred to as the vocabulary. A bag-of-words representation of a document is a vector $d_i = (x_{i1}, \dots, x_{im})$ where m is the number of words in the vocabulary. A common choice is to choose x_{ij} as the number of occurrences of w_j in document d_i . We will denote this number of occurrences as $f(d_i, w_j)$ throughout this thesis.

Term Frequency Inverse Document Frequency (TF-IDF)

Using the raw frequency count of a word is a basic example of how to use bag-of-words to represent documents, however longer documents have overall higher values than shorter ones. Furthermore, term frequency does not discriminate between terms that are frequent in most documents (and thus unlikely to be informative) and terms that are frequent in a few documents only. 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, introduced as "Term Specificity" by Jones [78]. 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.

$$TF(d, w) = \frac{f(d, w)}{\sum_i f(d, w_i)} \quad (2.1)$$

Where $f(d, w)$ is the number of occurrences of word w in document d , as before. The next part of TF-IDF is Inverse Document Frequency, which is a measure that rewards terms that have a low Document Frequency.

$$IDF(w) = \log\left(\frac{n}{df(w) + 1}\right) \quad (2.2)$$

Where $df(w)$ is the number of documents the word w has occurred in and n is the number of documents in the corpus, as before. Note that while Term Frequency measures the frequency of a term in a document relative to that document's length, Document Frequency measures the overall occurrences of the term across all documents. Essentially, it measures how rare it is for a term to appear in a document. Here, this value is multiplied by a logarithm to make the number of documents in the corpus less relevant (i.e. the value will not have a large difference between a corpus with 1,000,000 documents and a corpus with 1000 documents). Finally, the TF-IDF is the Term Frequency multiplied by the Inverse Document Frequency.

$$TF-IDF = TF \times IDF \quad (2.3)$$

This final multiplied TF-IDF value will balance the frequencies of the words such that terms that occur in many documents have a lower TF-IDF, and terms that occur in fewer documents will have a higher TF-IDF.

Positive Pointwise Mutual Information (PPMI)

Pointwise Mutual Information (PMI), originating from information theory [24], 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 is used as an alternative to TF-IDF that scores how dependent a word is on a document. In practice the frequency of the word is used to derive an approximation of the chance it will occur [15]. In application to e.g. a domain of movie reviews, we can understand that a word like "good" occurs frequently in many documents, while a word like "horror" occurs frequently only in some documents. In this case, "good" is not informative, as it is as likely to occur in one document as it is in any other, i.e. it is independent from the document, while "horror" is informative when it occurs in a document. For a word w in a document d , the $pmi(d, w)$ value is given by:

$$pmi(d, w) = \log \left(\frac{P(w, d)}{P(w) \cdot P(d)} \right) \quad (2.4)$$

Where $P(w, d)$ is equal to the chance of the word occurring in the document:

$$P(w, d) = \frac{f(d, w)}{\sum_i \sum_n f(d_i, w_n)} \quad (2.5)$$

Here, $f(d, w)$ is the frequency of a word for a document, i is the number of documents, and n is the number of words in the vocabulary.

To calculate the chance that a word will occur, $P(w)$, we simply take the chance the word will occur in any document (estimated by its summed frequency) over all frequencies, and to calculate the chance that the document will occur, $P(d)$, (represented by the sum of the frequencies of all words that occur in it) over all frequencies:

$$P(w) = \frac{\sum_i f(d_i, w)}{\sum_i \sum_j f(d_i, w_j)} \quad P(d) = \frac{\sum_j f(d, w_j)}{\sum_i \sum_j f(d_i, w_j)} \quad (2.6)$$

As this value can sometimes be negative when words are less correlated than expected, we use Positive Pointwise Mutual Information (PPMI) [43], as we are only interested in words which are positively correlated.

$$ppmi(d, w) = \max(0, pmi(d, w)) \quad (2.7)$$

In this thesis, a PPMI BOW is the representation used for bag-of-words, as it was shown to achieve better results than TF-IDF in previous work by Derrac and Schockaert [21]. It forms the basis of more complex representations and is also sufficient as a simple interpretable representation.

2.4 Text Document Classification

A classification problem has labels (or "classes") where each example either has a label or does not have one. 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 have a truth value $(d, c) = 0, 1$. These are the known examples, and from these values, a classifier finds a function that assigns unlabelled documents $d \in D$ to predicted labels (d, 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 Overfitting

If a machine-learning model is given training data, and tested on that training data, then the model could learn a function that is only useful for that data and still performs well, for example by memorizing the training examples. 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 [28]. 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.

Machine learning models have hyperparameters that determine how they function when training, and these can be adjusted to obtain results that do not overfit to the data. In order to determine the best hyperparameters to solve the problem, rather than choosing hyperparameters that solve the problem only on a particular set of data, 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 Validation data is used for parameter tuning. The separation of validation data from test data is to ensure that the parameters are not overfit on the test data.

The intention of this is to *validate* the models hyper-parameters using validation data, and *test* how well the model performs using test data. By keeping these examples separate, it is easier to see if the model is overfitting or if it has found generalizable patterns.

2.4.2 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.

$$a = \frac{C_n}{P_n} \quad (2.8)$$

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 .

There are a variety of other informative measures that can be used [52]. One such measure, recall (rec) is the proportion of true positives tp identified correctly.

$$rec = \frac{tp}{tp + fn} \quad (2.9)$$

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*).

$$pre = \frac{tp}{tp + fp} \quad (2.10)$$

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.

$$F1 = 2 \cdot \frac{pre \cdot rec}{pre + rec} \quad (2.11)$$

2.4.3 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, where a movie is either suitable for ages 18+, 15+, 12+ or suitable for everyone. The final kind of classification problem is multi-label [77]. 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".

2.5 Classifiers

2.5.1 Decision Trees

Classification decision trees [11] are classification models that are structured as trees, where 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 the class to which an example is assigned. Due to the way in which decision trees are learned, we can often think of the structure of the tree as identifying the importance of the different features, 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 [87].

2.5.2 Linear Support Vector Machines

In the case of a linear support vector machine (SVM), the documents are viewed as points in a vector space and the dimensions of that space correspond to the features. The SVM then finds a linear support vector machine [91] finds a hyperplane that maximizes the margin between documents belonging to different classes. To classify new documents, 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.

2.5.3 Neural Networks

Neural networks [32] 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 transforms the input. Some example activation functions are *tanh*, *sigmoid*, and *relu*.

$$\tanh(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)} \quad \text{sigmoid}(x) = \frac{1}{1 + \exp(-x)} \quad \text{relu}(x) = \max(0, x) \quad (2.12)$$

Generally, a neural network is composed of an input layer, which encodes 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 documents in that hidden layer are better spatially organized for solving the given task. For example, documents 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.

2.6 Low-dimensional 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 have not been seen during training. Moreover, the importance of words with few occurrences cannot be reliably determined. To address these issues, it has been proposed to represent the information in a bag-of-words in a lower number of dimensions while preserving the most relevant information as much as possible [20].

Low-dimensional vector spaces [35] are generally learned by taking semantic information in a sparse 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 low-dimensional vector space representation: the features are no longer meaningful, i.e. they are "entangled".

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 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.

2.6.1 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 dimensionality n . Essentially, PCA works by linearly combining features in order to create new features that can differentiate documents well and are uncorrelated with previous features. This results in a new low-dimensional representation that retains information and has features ordered by the amount of variance they capture. As they are a linear combination of all input variables [97, 29], they tend to be difficult to interpret, especially for high-dimensional input spaces. However, in some domains they have proven useful for exploratory analysis. For example, in spatial data investigating historical trends [79], the first six principal components were interpreted as known historical events, e.g. a spread of farming in the Middle East, or the retreat of the Basque language. In facial recognition, principal components have been found to correspond to semantic, albeit difficult to interpret features called 'eigenfaces' [85].

Latent Semantic Analysis (LSA) [27] is the same approach as PCA, obtaining principal com-

ponents, but with a bag-of-words as input [76]. The individual dimensions have been used to e.g. do exploratory analysis on scientific papers to identify communities of research [47]. Given that it relies on PCA, features are not always interpretable, and typically post-processing e.g. in the form of rotation is applied [40] to interpret components. To address the problem of interpretability in PCA, Sparse Principal Component Analysis (SPCA) [97] was introduced. SPCA instead obtains principal components that are composed of only some of the original variables [98]. A more computationally efficient method has also been applied to text data [95], showing some promise of interpretable features. Non-negative matrix factorization [51] is an approach similar to PCA but with non-negativity constraints, resulting in more interpretable semantic features [65].

2.6.2 Multi-Dimensional Scaling

Multi-Dimensional Scaling (MDS) [79] is a dimensionality reduction algorithm. In the same way as PCA, the dimensionality of the output space is specified. As input, MDS takes a dissimilarity matrix of documents, where both rows and columns are documents and the values are the dissimilarity between those documents. In this work, following the approach by Derrac and Schockaert [21], to calculate the dissimilarity between two documents d_i and d_j the normalized angular difference is used between their vectors v_{d_i} and v_{d_j} .

$$\text{ang}(d_i, d_j) = \frac{2}{\pi} \cdot \arccos\left(\frac{v_{d_i} \cdot v_{d_j}}{\|v_{d_i}\| \cdot \|v_{d_j}\|}\right) \quad (2.13)$$

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

2.6.3 Word Embeddings

Word embeddings are a vector space representation for words. They are typically learned using a large corpus of text, e.g. Wikipedia. There are many ways to obtain word-vectors, one

traditional approach is matrix factorization [12]. Recently modern methods like GloVe [66] and Word2Vec [60] 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 $\text{vec}(\text{word})$ is the word vector for a word, $\text{vec}(\text{King}) - \text{vec}(\text{Man}) \approx \text{vec}(\text{Queen})$.

2.6.4 Doc2Vec

Doc2Vec [48] extends the neural network method of learning word vectors introduced by Word2Vec [60] 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.

2.7 Interpretable Representations

2.7.1 Conceptual Spaces

The inspiration of the work by Derrac and Schockaert [21] 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 [86]. Conceptual spaces [26] 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 [33], 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 of entities in the domain. The representation is considered disentangled if the features obtained are interpretable and predictive when used in key domain tasks. In this thesis, the term "property" is sometimes used in place of the term "feature", to make the distinction between the "properties", which are directions in the document embedding, and "features" that are the resulting rankings of documents on those features.

2.7.2 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 [9]. Topics tend to correspond to salient features, and are typically labelled with the most probable words according to the corresponding distribution.

Vector space models 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 [18]. Topic models have also been used to improve word embedding models, by learning a different vector for each topic-word combination [53]. 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 [82], modelling correlations between topics [8], or by incorporating meta-data such as authors or time stamps [71, 92]. Such techniques for extending LDA offer less flexibility than neural network models, e.g. for exploiting numerical attributes

or visual features.

2.7.3 Generative Adversarial Network and Variational Autoencoders

Generative Adversarial Network (GAN) [30] 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, GANs 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 [14]. GANs have also been applied in text [10, 42] with some success, despite being noted as ‘particularly difficult to train’ in the text domain [3] even with advancements in this direction [59]. The work in this thesis differs from the disentangled representations found in GANs 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.

2.7.4 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 [61] or [54] adapting neural networks. Alternatively, some works follow a similar line to ours in that they post-process existing dense embeddings [80, 64, 25]. In the former category, this approach has also been extended to sentences [83], and follows the idea that PCA and other dense representations are effective at compressing information into a small number of dimensions, although this usually 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 features. 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 [84].

There are also document representations that use sparsity constraints to obtain interpretable sparse representations such as sparse PCA learned using the l_1 -norm, [97, 95] or Sparse MDS [19]. 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 features of the domain then we should only need a small number of features to perform well at key domain tasks e.g. text classification.

One method that does not produce a sparse representation but still learns an interpretable representation is [44], 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.

2.8 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 dimensionality 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.

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 characteristic of our representation.

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 quantitatively and qualitatively validated across a variety of domains and vector space embedding methods. 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.

Datasets and Semantic Spaces

3.1 Introduction

For the experiments in this thesis, we will use five different domains, 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. This is a binary classification task.

20 Newsgroups¹ 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

¹<http://qwone.com/~jason/20Newsgroups/>

Domain	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.

concatenated 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. This is a multi-class classification task.

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 Reuters², as a result of which they often 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. This task is multi-label.

Place-Types Taken from work by Derrac and Schockaert [21]. Originating from the photo-

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

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 and Schockaert [21]. 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. All of these tasks are multi-label. Although it may be expected that the age-ratings task is multi-class, it is actually multi-label, as a movie can be both UK-12-12A and USA-PG-PG13.

³<https://snap.stanford.edu/data/web-Amazon.html>

3.3 Technical Details

In this section, we describe the vocabulary and document characteristics of each domain. Each domain is preprocessed by converting all words to lower-case, non-alphanumeric characters are removed and whitespace is stripped such that words are separated by a single space. Words were removed from a standard list of English stop words from the NLTK library [6] and we filter out terms that do not occur in at least two documents. Additionally, any words that are not in the top 100,000 most frequent are removed, this is a standard preprocessing method to remove terms that are not informative.

Each task in the domain has a different number of labelled documents. For each task in each domain, only the labelled data are used and that data is split into 2/3 training data, 1/3 test data. Additionally, 20% of the training data is removed and used as development data for the hyper-parameters.

IMDB Sentiment⁴ 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 Newsgroups⁵ 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 size of the original vocabulary is 141,321 and after filtering out terms that did not occur in at least two documents, the end result is a vocabulary of size 51,064. This is a larger change after filtering the vocabulary than for the Sentiment dataset, despite beginning with a larger vocabulary, this shows that many words in this dataset are only appear in one Newsgroups document. The data is not shuffled. The number of positive instances averaged across all classes is 942, exactly 5%.

Reuters-21578, Distribution 1.0 Obtained from NLTK⁷ originally containing 10788 docu-

⁴Obtained by: <https://keras.io/datasets/>, Originally from <https://ai.stanford.edu/~amaas/data/sentiment/> [55]

⁵<http://qwone.com/~jason/20Newsgroups/>

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

⁷<https://www.nltk.org/book/ch02.html>

ments. 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 size is 51,001 and as in each other domain 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%.

Place-Types 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 words that are not in at least 6 documents. As most classes in this domain are extremely sparse (less than 100 positive instances). 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, we ended up with 13978 documents. In the same way as the Place-Types, the vocabulary hit the hard limit of size 100,000.

3.4 Representations

We use the bag-of-words representation of the documents as a baseline. In this case, terms are additionally filtered out that do not occur in at least 0.001% of documents, as to scale with the number 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, frequency bag-of-words are not able to distinguish between terms that are frequent for a document and terms that are frequent overall. The former are less meaningful for distinguishing the difference between documents than the latter. To accommodate this, words are weighed such that words which occur frequently in a small number of documents are given a higher value than those that occur frequently overall.

Specifically, Positive Pointwise Mutual Information (PPMI) scores are used, following success in similar work by Derrac and Schockaert [21]. See Section 2.3.1 for more detail.

For the work in the following chapters, we wanted a variety of different vector space models to test the generality of the proposed methods. Below the choices for the vector space models that are formally described in Section 2.6 are explained:

Multi-Dimensional Scaling (MDS) (See Section 2.6.2): Multi-Dimensional Scaling (MDS) is used for comparison, as it was the only space used in the work by Derrac and Schockaert that introduced this method [21]. In this case, the input is a matrix of dissimilarity values between the PPMI vectors of documents of size $n \times n$, where n is the number of documents.

Principal Component Analysis (PCA) (See Section 2.6.1): 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 2.6.4): Doc2Vec is inspired by the Skipgram model [49]. It is distributional in the sense that the context of words and documents is used during its learning process. It is used here as 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 *window size*(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 2.6.3) that was trained on the Wikipedia 2014 + Gigaword 5 corpus⁸. While simply averaging word vectors may seem naive, this was found to be a competitive approach for unsupervised representations in several applications [34]. We simply average the vector representations of the words that appear at least

⁸<https://nlp.stanford.edu/projects/glove/>

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 $C(1.0, 0.01, 0.001, 0.000)$, 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 4.4.2).

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.

Disentangling Unsupervised Document Embeddings

4.1 Introduction

Vector space models encode meaning spatially, but their features are typically 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 [66], learning visual data alongside text data to explain the content of images [57], and enforcing grammatical structure to perform better at question answering tasks [63].

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 and Schockaert [21], 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 2.5.2), where the semantic importance of words in the domain can be evaluated using standard model evaluation metrics like F1-score (see Section 2.4.2), as this measures how linearly separable documents are for that term.

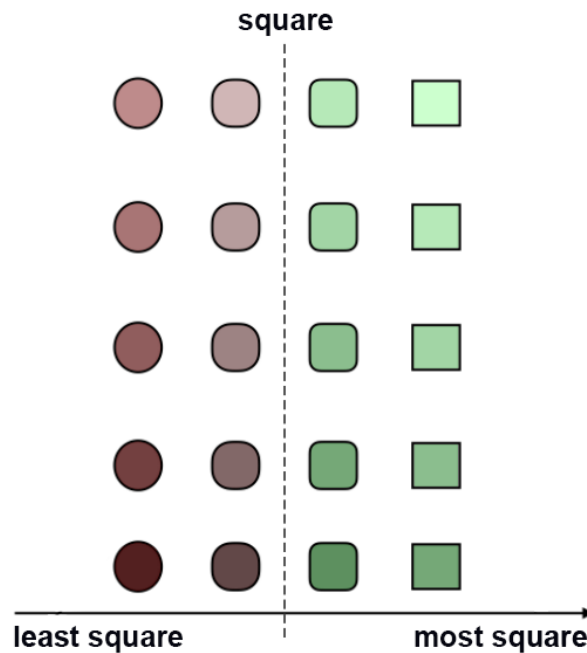


Figure 4.1: An example of a hyperplane in a toy domain of shapes. The hyperplane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples. Those closest to the hyperplane are less square than those further away.

The Linear SVMs obtain a hyperplane 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 hyperplane. 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 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 Figure 4.1).

In a text domain like movie reviews, a hyperplane can be obtained that separates documents which contain a word from those that do not contain that word. The dot products between the

orthogonal direction vector and the document vectors can then be obtained, essentially measuring how strongly related the document is to that word. These dot products can be taken as a ranking of documents according to how related they are to that word, e.g. how ‘Funny’ each movie is relative to the other movies. This process can be done for all words in the vocabulary. where one way to measure how useful they are as features is to evaluate how well the documents can be separated by the SVM. We can expect that if the documents are well separated, the considered word should correspond to a good feature.

The first step of the approach to obtain semantic features from a Multi-Dimensional Scaling space as described by Derrac and Schockaert is to obtain hyperplanes for all words based on binary word occurrence. Then, words that are semantically important are scored (e.g. by the accuracy of the SVM classifier). Finally, orthogonal directions for the hyperplanes of the highest scoring SVMs are obtained and documents are ranked on how far up they are along these highest-scoring directions. These rankings are then used as the new disentangled representation.

However, directions labelled with single words 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" could be clustered as "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 4.2.3) that uses the word direction vectors as input. Then, a new direction can be obtained that more accurately models the feature described by e.g. taking the cluster-center of the K-means algorithm as direction vectors for the cluster. 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 and build on the method introduced by Derrac and Schockaert [21], in-order to disentangle a document embedding into semantic features. Derrac and Schockaert’s quantitative investigation was of disentangled cluster feature representations derived from a Multi-Dimensional Scaling document embedding model (See Section 2.6.2), used as input to rule-based classifiers. The domains investigated by Derrac and Schockaert were the Place-Types and Movies domains, as well as a domain of text describing wines (not included in this work as it is very noisy and contains too few documents). The key contribution of this chapter is an extensive quantitative examination in Section 4.4 and qualitative investigation in Section 4.3 across five domains (as described in Chapter 3), where

IMDB Movie Reviews	Flickr-Place-Types	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.

disentangled feature representations are derived using four document embedding models. In particular, this chapter evaluates how effective the method by Derrac and Schockaert [21] is at producing semantic features when moving beyond the particular settings that were considered in their work. We also aim to evaluate the quality of these features in a more direct way. To this end, we propose the use of low-depth decision trees that are trained on natural categorisation problems for the considered domains. Using these disentangled feature representations as input, a low-depth decision tree uses a limited number of features to classify. The extreme case, which is of particular interest for our purpose, is a decision tree of depth one (i.e. a root node only), which can only use a single feature. If a low-depth decision tree can perform well on these natural categorisation problems when using the disentangled feature representation as input, then it must contain semantic features that capture the salient properties from the domain. To add to the depth of the analysis, variants of the method by Derrac and Schockaert [21] are introduced, specified in Section 4.2. First, the scoring metric Normalized Discounted Cumulative Gain (NDCG) is used, following the insight that measuring the quality of the ranking rather than the performance of the linear classifier may better measure which features are most salient. This metric performs well in a majority of cases. Additionally, the standard K-means clustering algorithm is used for comparison to the K-means variation used in Derrac and Schockaert. Finally, we pay particular attention to the choice of the document embedding model. In particular, an important question which we aim to answer is whether the method from Derrac and Schockaert [21] is limited to spaces learned using Multi-Dimensional Scaling (MDS), or whether similar, or even better, results are possible with other embedding strategies.

This is important because of the limited scalability of Multi-Dimensional Scaling.

4.2 Method

This section details the methodology to disentangle a document embedding model starting with the document vectors and their associated bag-of-words representations.

4.2.1 Obtaining Directions and Rankings From Words

The method starts with a given document embedding induced from a set of text documents D and their associated bag-of-words $d_i = (x_{i1}, \dots, x_{im})$ where m is the number of words in the vocabulary. In this section, x_{ij} is the number of times $f(d_i, w_j)$ that word w_j occurs in document d_i .

Obtaining directions for each word

Let us write $df(w)$ for the number of documents a word has occurred in. Directions are only obtained for words higher than a threshold $df(w) > T$. Each document is represented by a vector v in the document embedding. For each word w , a linear classifier is trained to separate documents v where the word w occurs at least once, from documents where the word does not occur. Both a Logistic Regression classifier and a Linear SVM were tested, and both achieved similar results. 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.¹ This is to ensure that the classifier does not overfit on negative samples. For more information, see Section 4.4.2.

The distance between each document vector v_d and the hyperplane h_w is meaningful, as it can be expected that the documents which are most strongly related to the word are further away from the hyperplane 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

¹Using scikit-learn, class_weight: 'balanced'

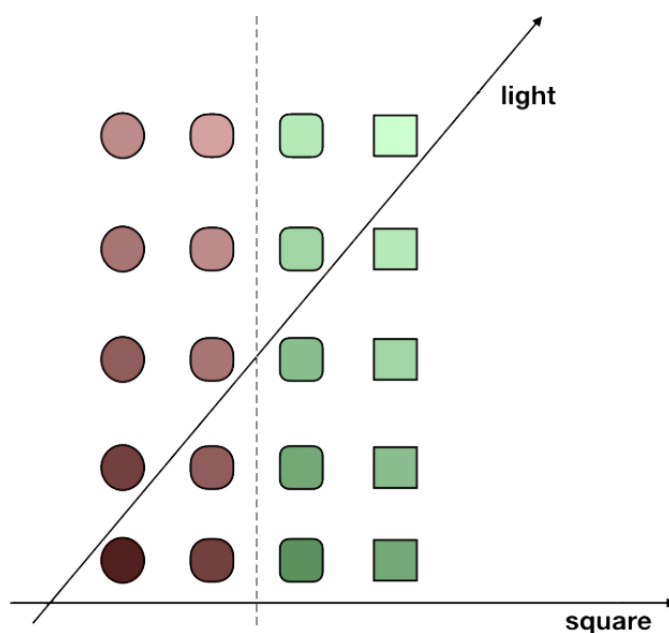


Figure 4.2: Another example of a hyperplane in a toy domain of shapes. Here we show multiple directions, one for light and one for square. The hyperplane for the word square is the dotted line. Green shapes are positive examples and red shapes are negative examples for the word square.

an example of two directions in Figure 4.2. Here, shapes that are "more square" are far away from the hyperplane on the right side, and shapes that are more circular are further away from the hyperplane on the left side. To give an example in a real domain, a direction for the word "Scary" would point from the least scary movies to the most scary movies in the document embedding space.

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 $r(w, d)$ for the dot product $\mathcal{D}_w \cdot v_d$. Then, d_1 is ranked higher than d_2 for word w if $r(w, d_1) > r(w, d_2)$. These rankings measure how relevant the document is in the spatial representation for the word, rather than just based on frequency. For instance, a document that contains the word "scary" but is not 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 is assumed to be duller than lower 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 select the most important features, given the hypothesis found empirically to be true in Derrac and Schockaert [21] that if a word is not well represented in the space it is not semantically important. This hypothesis is based on the intuition that important features should have a larger impact on similarity scores than less important features, and since Multi-Dimensional Scaling representations are obtained from similarity scores, it can thus be expected that important features should also be reflected more clearly in the structure of the vector space. Other document embedding methods are not directly learned from similarity scores, but they are typically also designed to capture similarity, hence we can expect to see a similar effect in other types of embedding spaces.

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" killer². To solve this problem, Section 4.2.3 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.

4.2.2 Filtering Word Directions

Although we are able to obtain word directions for every word, not every word describes a salient property. This section describes how to filter out word directions that are not semantic.

²The real cluster of words that this example comes from is "gore gory bloody blood gruesome serial investigate deaths"

The assumption made by Derrac and Schockaert [21] 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 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 discuss four potential scoring metrics, three of which have not previously been considered for this problem.

Cohen’s Kappa. This is the only metric used in the work by Derrac and Schockaert [21]. This metric evaluates the performance of the classifier, and also deals with the problem that these words typically only appear in few documents, making the classification problem 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. 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 semantic feature for the given domain. However, despite balancing the weights of the original SVM used to obtain the hyperplane, 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 2.4.2, 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 features 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 (PPMI, see Section 2.3.1) scores for the word in each document. Other metrics were briefly tested, such as Spearman’s Rho [17], but these metrics are not useful for this purpose as these ranking-based scoring functions give equal weight to the lowest and highest ranked documents, and PPMI ranks many lower ranked documents the same. 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 [39], and prioritizes higher rankings.

First, the ranking of documents induced by a word w , obtained using the dot product with a direction $\mathcal{D}_w \cdot v_d$, we define a mapping from documents to integers, where each document is mapped to its position in the ranking. The relevance scores are determined by the Pointwise Positive Mutual Information (PPMI) score $ppmi(d, w)$, of the word w in the BoW representation of document d (See Section 2.3.1).

$$\begin{aligned} \text{DCG}_R^w &= \sum_{i=1}^{pr_d} \frac{ppmi_i^w}{\log_2(i+1)} \\ \text{IDCG}_R^w &= \sum_{i=1}^{|documents|} \frac{2^{ppmi_i^w} - 1}{\log_2(i+1)} \\ \text{nDCG}_R^w &= \frac{\text{DCG}_R^w}{\text{IDCG}_R^w} \end{aligned} \tag{4.1}$$

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 documents ordered according to the relevance scores for the term up to position pr_d . NDCG is then simply the DCG normalized by the IDCG.

4.2.3 Clustering Features

It can sometimes be ambiguous what feature is captured by the direction induced by a given 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 naive 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 $c = (w_1, w_2, \dots, w_n)$, 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 4.2.1 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 "Gory" are clustered together, then the corresponding cluster direction will model both of these terms. A cluster direction may be desirable if both "Bloody" and "Gory" 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 "Gory". 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 features 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, we will analyse the performance of the standard K-means algorithm alongside the clustering algorithm introduced by Derrac and Schockaert [21]. For both clustering methods, the only hyper-parameter tuned is the number of clusters:

Derrac and Schockaert's K-Means Variation

This is the clustering method used in Derrac and Schockaert [21]. As input to the clustering algorithm, it considers the n best-scoring candidate words $B = (w_1, \dots, w_n)$, and the number of clusters j . The main idea underlying the 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 clusters $C = (c_1, \dots, c_j)$ and their associated directions $CD = (\mathcal{D}_{c_1}, \dots, \mathcal{D}_{c_j})$, where j is the number of clusters and \mathcal{D}_c is a direction that represents all words in a cluster. Each cluster is composed of words $c = (w_1, w_2, \dots, w_n)$.

As before, we will write \mathcal{D}_w for a direction corresponding to a word w . First, a single word cluster center is chosen for each cluster. The first cluster center is chosen by taking the top-

scoring word for a chosen scoring metric; see Section 4.2.2 for possible metrics. In the work by Derrac and Schockaert [21], only Kappa score was used. Then, in each iteration, a new cluster c is chosen by selecting a word direction v , among the directions of the words in B , that is least similar to all other chosen clusters:

$$v = \operatorname{argmin}_{w \in B_n} (\max_{u \in CD} (\cos(\mathcal{D}_w, \mathcal{D}_u))) \quad (4.2)$$

Where B_n are the candidate word directions. In principle, only candidate word directions that haven't already been chosen as cluster centers should be considered. In practice, as chosen cluster centers get the highest score regardless, they would never be picked by the argmin function. Once j cluster centers have been selected, each candidate direction \mathcal{D}_w is added to the most similar cluster center, i.e. the cluster represented by the chosen center word $j \in c$ for which $\cos(\mathcal{D}_w, \mathcal{D}_j)$ is maximal. Finally, each cluster's direction is taken as the average of all the word directions in the cluster.

K-Means

K-means is a standard baseline clustering algorithm. In the experimental results, it was found that Derrac and Schockaert's variation relies too much on the word initially chosen to be each cluster center, as 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. The K-means algorithm is implemented in scikit-learn with default parameters ³.

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 [4], 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

³<https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

already been chosen:

$$\frac{D(x)^2}{\sum_{x \in X} D(x)^2} \quad (4.3)$$

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.

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).

4.3 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 an intuition of what semantic features can be discovered in each domain, the highest scoring terms are shown from a variety of domains in Section 4.3.1. Then, the differences between different document embedding models in the domain of Movies is investigated in Section 4.3.2. Next, in Section 4.3.3 the differences between scoring metrics are investigated by looking at the top scoring terms for each one. Finally, in Section 4.3.3 the Doc2Vec representation is investigated in the Newsgroups domain, as it could not be investigated in the Movies domain.

4.3.1 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 4.2.3.

There is an interesting difference between the Sentiment directions and the Movies directions in Table 4.2. In the Movies domain, features of movies e.g. genre related directions like "Horror" or "Romantic" are the highest scoring. However, in the Sentiment domain the features 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 features about movies. Similarly for the Newsgroups domain, directions that seem relevant for 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", "tranquil"), or awards that are given to sub-categories of photos (like "mygearandme, a group competition where users submit photographs). These salient directions probably score well on tasks like the Foursquare task, as the classes for that task 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)").

4.3.2 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 with 200 dimensions 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). Theoretic-

Movies (50 MDS NDCG)	Sentiment (100 D2V NDCG)	News (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, fuici)	wk (5173552178, 180844mbmclmsuedu)	arquitectura (edificio, arquitetura)	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 (comedy, 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 (ohwm, cipertext)	stones (mygearandme, flickrsbest)	1st (qtr, pennsylvania)
western (outlaw, unforgiven)	rouke (mickey, walken)	gd3004 (35894, intergraph)	wilderness (glacier, peak)	26 (approve, inadequate)
romantic (romance, chemistry)	ustinov (warden, cassavetes)	rtfmittedu (newsanswers, iee)	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)	poroshe (nanao, mercedes)	geology (mineral, formations)	june (july, venice)
noir (noirs, bogart)	homer (pacino, macy)	gebcdredslpittedu (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)	boil (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 (streeep, 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 (bobbeiceicote.com, beauchaine)	fauna (mammal, insect)	13 (dlrsbbl, groups)

Table 4.2: The top-scoring words for each domain

ally, 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 features of the domain being represented in all of the different document embedding models.

When examining the table of results, the common terms seem mostly semantic features relevant to the domain, e.g. "noir", "comedies", "western", "documentary". The space learned using MDS captures the most unique features (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)", "comics (comedian, comedians)", "kidnapping (kidnapped, torture)", "gambling (vegas, las)", despite also capturing some unique noise "berardinelli (employers, distributor)", "crawford (joan, davis)". The AWW space capture some features that are interesting (train, slaves), but there are not many of them. Meanwhile PCA seems to capture many unique features 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*)".

4.3.3 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 features 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 clas-

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 (marial, 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

NDCG	F1	Accuracy	Kappa	Common
gay (homosexuality, sexuality)	company (sell, pay)	kennedy (republic, elected)	definitely (alot, awesome)	horror (scares, scares)
arts (hong, chan)	street (city, york)	bags (listened, salvation)	guns (gun, shoot)	laughs (funnier, funnier)
spots (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)	email (web, web)
evidence (investigation, accused)	values (exception, wise)	confined (cooler, downward)	thrilling (thrill, exciting)	web (email, email)
adorable (cute, sweet)	unusual (odd, seemingly)	inhabit (quentin, drawback)	horror (scares, scares)	laughs (funnier, funnier)
episodes (episode, television)	lovers (lover, lovely)	latin (communities, mount)	suspense (suspenseful, suspenseful)	
teenager (teen, teenage)	frame (image, effect)	reception (como, finishes)		
magical (fantasy, lovely)	mans (ultimate, sees)	uptight (suspensful, stalked)		
health (medical, suffering)	efforts (generally, nonetheless)	brink (inexplicable, freddy)		

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

sifier on the genres task. Following this, it can be seen that a lot of features 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, 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 some meaningful terms e.g. "guns (*gun, shoot*)", "flying (*air, force*)" we do not find unique meaningful and general features the way we did for NDCG. These qualitative examples contribute to the idea that NDCG, by evaluating the ranking rather than the separability, ends up discovering unique features that are salient for natural categorisation 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 [81]. 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 features that are less meaningful compared to Doc2Vec, specifically related to insignificant conversational words, e.g. "hi (*folks, everyone*)", "sorry (*guess, hear*)", "say (*nothing, anything*)", that do not seem related to 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 features, e.g. "cryptology (*attendees, bait*)", which is very relevant to the newsgroup

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, noing)	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.

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.

4.4 Quantitative Results

This section evaluates the semantic features using low-depth decision trees. To begin, the evaluation method is explained in Section 4.4.1. This is followed by an overview of the hyperparameters that are used for the classifiers in Section 4.4.2. Following this, the experimental

results are obtained. First, a summary of all the results is shown in Section 4.4.3. 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 input to a variety of classifiers in Section 4.4.4. In Section 4.4.5 the results of a disentangled feature representation composed of only single-term features are obtained and compared to the baselines. Finally, in Section 4.4.6 a disentangled feature representation composed of cluster features is obtained and compared.

4.4.1 Evaluation Method

Low-depth CART Decision Trees are used (See Section 2.5.1) limited to a depth of one, two or three. The idea behind this is that if the features are semantically meaningful, then the decision trees with a limited depth should be able to perform well in natural categorization tasks.

Essentially, if depth-one decision trees can model a class well using the disentangled representation as input, this means that a single feature in the disentangled representation essentially captures this class. 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. 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 features 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.

4.4.2 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 implementation from the gensim library[67]⁴ is used.

⁴Doc2Vec implemented in gensim.

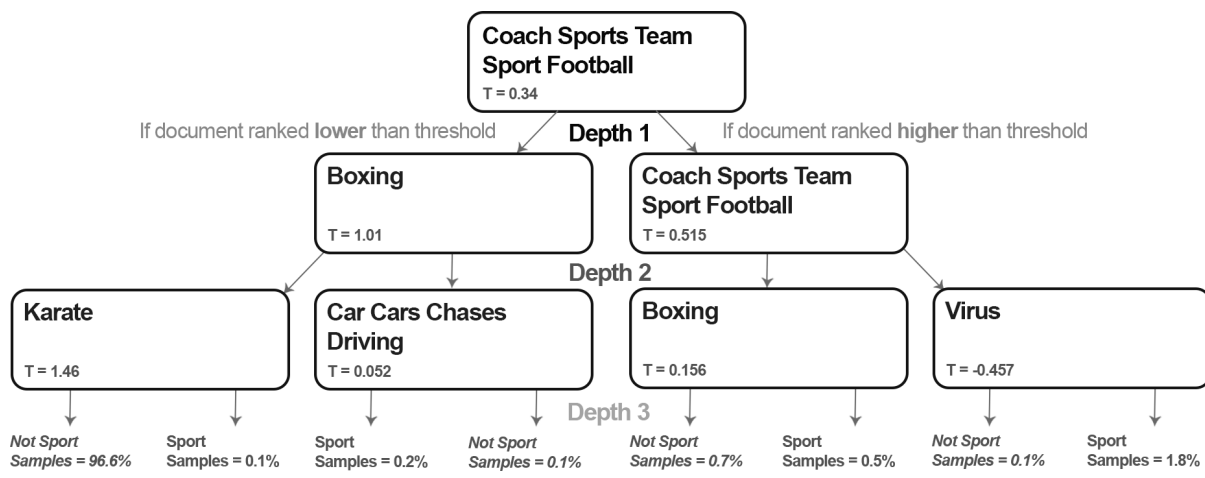


Figure 4.3: An example of a decision tree classifying if a movie is in the "Sports" genre. Each Decision Tree Node corresponds to a feature, and the threshold T is the required value of the feature for a document to traverse right down the tree instead of left. One interesting point to note is that the 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) .

An exhaustive search of all combinations of hyper-parameters are tested using grid search. In all hyper-parameter tuning, the model is tested on the validation data to find the hyper-parameters that result in the best performance.

Classifier Parameters

There are two classifiers that are used across all experiments. CART Decision Trees⁵ and linear SVMs⁶.

The hyper-parameters of each SVM are also tuned to find the best C values (1.0, 0.01, 0.001, 0.000) (See Section 2.5.2 for more detail) and to decide whether the weights should be balanced such that positive instances are weighted in proportion to how rare they are⁷. In particular if weight balancing is used, where n is the number of documents, and y is the number of positive examples

⁵Decision Tree implemented in scikit-learn

⁶Implemented in scikit-learn

⁷Weights implemented in scikit-learn

in the training data, the training sample is weighted equal to:

$$\frac{n}{2 * y} \quad (4.4)$$

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

- The number of features⁸ 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* means that the number of features considered is $\log_2(n)$
- 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.⁹
- If the weights should be balanced.

Experiment One: Obtaining baselines

This first experiment is used to determine the best-performing document embedding models 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 Dirichlet Allocation (LDA) topic model. The BOW-PPMI is used as a reference for a standard 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, as explained in Section 3.3, 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:

⁸"max_features" in scikit-learn

⁹"criterion" in scikit-learn

- The *window_size*(5, 10, 15) referring to the context window of the words that used during training.
- 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 Dirichlet Allocation topic models¹⁰, 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 features

This experiment shows 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 single-term 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 tuned 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 documents on the top *s* directions.
- The score-type *t* (*F1, Kappa, Accuracy, NDCG*) which is the method used to score the directions.

¹⁰Topic models implemented in scikit-learn

The reason these hyper-parameters are tuned is 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-means¹¹ and Derrac's K-means variation (as described in Section 4.2.3), are used. The best single-term feature 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 number of clusters.

4.4.3 Summary of all Results

Table 4.6 is a summary of the three experiments just described, listing results for the different representations on trees limited to depth-one, two and three. The primary takeaway is that the initial document embedding vectors did not perform as well on the task as single directions and cluster directions. Further, these disentangled feature representations outperform our baseline PPMI bag-of-words in most cases with two exceptions. The first exception is the tree of depth two for the Place-Types domain, and the second is the depth-one tree on the Movies keywords task. The general takeaway is that the feature representations obtained using the method described in this chapter outperform our baselines in almost all cases.

To explain these exceptions, we can look at the keywords task. In a tree of depth one, 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 trees

¹¹K-means implemented in scikit-learn

limited to a depth of two and one, the score is no longer as good for PPMI, which is likely due to overfitting (See Section 2.4.1 for more on overfitting in general).

With a more focused view on variance in the decision trees, sometimes decision trees of depth-two outperform those of depth-one, but generally depth-three trees perform best. This is understandable, as there are many semantic features that can be used to refine the Decision Trees. With view to the case of the Place-Types, we find that the PPMI representation outperforms the feature representation, however it is not by a wide margin. Meanwhile the single directions perform better in all other tasks and tree types, with a wider margin.

To summarize, the overall takeaway from this section is that with some small exceptions, the feature representations obtained using the methods detailed in this chapter are superior to our baselines of Latent Dirchlet Allocation topic models and PPMI bag-of-words representations.

4.4.4 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.

	Genres			Keywords			Ratings		
	D1	D2	D3	D1	D2	D3	D1	D2	D3
Movies	0.301	0.358	0.354	0.185	0.198	0.201	0.463	0.475	0.486
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		
Place-Types	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

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 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 disentangled 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.

4.4.5 Single-Term Features

In Table 4.9 the results for the feature representation composed of rankings of documents 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 dimensionality 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 document is religious like the word direction for "celestial". However, variants of the space dimensionality 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 Reu-

Newsgrroups	D1			D2			D3			DN			SVM							
	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 Newsgrroups 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	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					
Place-Types	D1			D2			D3			DN			SVM		
OpenCYC	ACC	F1	ACC	F1	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					
Place-Types	D1			D2			D3			DN			SVM		
Foursquare	ACC	F1	ACC	F1	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					
Place-Types	D1			D2			D3			DN			SVM		
Geonames	ACC	F1	ACC	F1	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					
Sentiment	D1			D2			D3			DN			SVM		
PCA	ACC	F1	ACC	F1	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	0.811	0.811	0.815	0.822					
Movies	D1			D2			D3			DN			SVM		
Genres	ACC	F1	ACC	F1	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					
Movies	D1			D2			D3			DN			SVM		
Keywords	ACC	F1	ACC	F1	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					
Movies	D1			D2			D3			DN			SVM		
Ratings	ACC	F1	ACC	F1	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.

ters 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 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 features 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 dimensionality 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.

4.4.6 Clustered Directions

Clustering has two main goals: the first is that features are improved by e.g. obtaining a new feature that is more semantic in the domain, 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.

Surprisingly, the default K-means outperforms the Derrac and Schockaert variant in many cases. This shows that although the Derrac and Schockaert variant is useful in many cases, it is also

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
AWV	0.975	0.598	0.979	0.656	0.98	0.66	AWV	0.7	0.699	0.711	0.736	0.723
MDS	0.975	0.678	0.98	0.706	0.982	0.72	D2V	0.776	0.784	0.782	0.801	0.822
D2V	0.977	0.583	0.979	0.664	0.98	0.632						
Place-Types							Movies	D1	D2	D3		
	D1	D2	D3	Genres			D1	D2	D3			
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
OpenCYC	0.632	0.371	0.704	0.381	0.735	0.365	PCA	0.824	0.412	0.82	0.441	0.913
PCA	0.66	0.404	0.734	0.428	0.755	0.39	AWV	0.81	0.421	0.837	0.436	0.912
AWV	0.658	0.374	0.711	0.385	0.746	0.35	MDS	0.849	0.446	0.839	0.463	0.918
MDS							Keywords	ACC	F1	ACC	F1	ACC
Foursquare	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
AWV	0.918	0.541	0.881	0.498	0.889	0.466	AWV	0.656	0.201	0.672	0.203	0.652
MDS	0.82	0.416	0.879	0.482	0.897	0.485	MDS	0.745	0.23	0.74	0.233	0.708
Geonames							Ratings	ACC	F1	ACC	F1	ACC
	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
AWV	0.711	0.444	0.795	0.533	0.802	0.473	AWV	0.646	0.463	0.692	0.474	0.677
MDS	0.591	0.289	0.772	0.333	0.764	0.352	MDS	0.62	0.463	0.692	0.489	0.686

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

useful to vary the clustering method. 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 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 and Schockaert variant, the corresponding cluster was ("franchise sequels sequel instalments entry returns"). This cluster was also chosen for the Derrac and Schockaert 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 and Schockaert clustering

Newsgrroups	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 Newsgrroups

Reuters	D1		D2		D3		Sentiment		D1		D2		D3	
	ACC	F1	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	
Place-Types	D1	D2	D3	Movies	D1	D2	D3							
OpenCYC	ACC	F1	ACC	F1	ACC	F1	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 documents ranked on the directions output by the clustering method for the optimal number of clusters for each domain and task.

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 soldier army sergeant sgt platoon infantry"). In conclusion, Derrac for the Keywords task captured certain features 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 Conclusion

In conclusion, this chapter described an extensive analysis of methods to go from an initial document embedding and an associated bag-of-words to a disentangled feature representation. Low-depth decision trees that classify on key domain tasks in a variety of domains perform well when using these representations as input, showing that these representations are disentangled. 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 [21], 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 [21]. 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.

Disentangling Neural Network Layers

5.1 Introduction

The previous chapter showed how in a vector space of documents (e.g. different movies) semantic features can be identified with directions (e.g. more violent than) and used as features in a document classification task.

This chapter addresses two research questions. The first is whether meaningful features can be obtained from neural networks. This question is addressed using a quantitative investigation of a disentangled representation obtained from each layer of the network, which is evaluated in the same way as the previous chapter. The second question is whether neural networks can be better understood through the lens of semantic features. This question is about the insight into how these networks achieve their results, and is investigated by examining these networks quantitatively and qualitatively.

Two kinds of neural networks are used, feed-forward networks and auto-encoders. The hidden layers of these networks are viewed as vector spaces, and disentangled feature representations can be obtained from them using the method from Chapter 4. These disentangled feature representations are quantitatively evaluated, and the labelled outputs are qualitatively investigated. In the case of the feed-forward networks, it is found that predictive features can be derived from their hidden layers. This is validated using low-depth decision trees, and qualitative examples further validate this claim. In particular, it is found that a low-depth decision tree that uses a disentangled feature representation obtained from a feed-forward neural network hidden layer can outperform the original neural network, in this case, in the 20 Newsgroups domain. It is also found that feed-forward networks do not necessarily introduce new properties in order to perform well, but rather select and refine relevant properties to the classification task.

Auto-encoders are also investigated by obtaining semantic features from their layers. First, a negative result is found in a quantitative experiment using low-depth decision trees, showing that there is a significant drop in accuracy and F1-score between properties from the original representation and those obtained from the hidden layers of a stacked denoising auto-encoder. However, it is observed that the properties obtained in the representation derived from the stacked denoising auto-encoder layers tend to be meaningful. In particular, it is found that earlier in the stacked denoising auto-encoder less frequent (i.e. specific to fewer documents) features are obtained, and in later layers more frequent (i.e. more general to more documents) features are obtained. Following this insight, symbolic rules are induced that relate less frequent properties to more frequent 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.

$$\text{IF Emotions AND Journey THEN Adventure} \quad (5.1)$$

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 5.7 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 [2]. 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 [41] 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 [75] 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 [72] is a pedagogical approach that calculates changes to the input of the network to find hyperplane rules that explain how the network functions.

In Section 5.6 a disentangled representation is derived from the hidden layers of a neural network, and rules are learned from that disentangled representation, rather than learning rules that describe the relationships between units in the network itself. The focus of this section 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 [69]. 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 [68]. 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 [96]. 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 representation 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 document embeddings using symbolic rules [36]. In [70] 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.

5.3 Method

5.3.1 Stacked Denoising Auto-encoders

In this section, the method to obtain a series of entity embeddings from stacked denoising auto-encoders is explained. The intention is to evaluate what is represented in the hidden layers of stacked denoising auto-encoders, and to understand the relationship between the layers of the representation.

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 embeddings that are different from the initial embedding provided by the MDS method, stacked denoising auto-encoders [90] 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 [5]. The task of recreating the input is made non-trivial by constraining the dimensionality 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 generalize specific information into abstract information, as it must "fill in the gaps". By repeatedly using the hidden layer as input to another auto-encoder, we can repeatedly transform the representation. 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.

5.3.2 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 has a number of nodes equal to the number of classes. Each network is trained on one task from each domain. Our main focus is on the qualitative analysis, because the features in these domains are easier to analyse. 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 (See Section 2.4.3), 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 [46] [62], 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.

In preliminary experiments, a network with two layers outperformed a single layer network consistently. As such, NNET-B uses a bag-of-words as input and has two hidden layers. The first of these hidden layers has 1000 nodes while the second has 100. In the case of Newsgroups, this network outperformed the top-performance linear SVM that 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 dimensionality 100 in the learned network. For the qualitative examples derived from this network, the final hidden layer of dimensionality 100 was used as well.

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 4.6. For example, in the Newsgroups domain, this is the Doc2Vec embedding with 100 dimensions. 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 number of nodes in the hidden layer is set to have the same dimensionality as the input representation. This enables straightforward comparison between the initial embedding and the embedding derived from the neural network.

5.4 Investigating Feed-forward Neural Networks

5.4.1 Parameters

Three different domains are investigated: Newsgroups, Place-Types 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 Section 4.2.1 the words used when obtaining candidate directions are limited by a frequency 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 disentanglement 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 4.2.2) 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 4.4.2.

5.4.2 Quantitative Evaluation Of Feed-forward Networks

The intention of this experiment is to validate that the disentangled feature representations derived from feed-forward networks are composed of semantic features, and this is measured as in Chapter 4 by the performance of low-depth decision trees. The second intention is to analyse 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 between the depth-3 limited decision trees that use disentangled feature representations derived from neural networks as input to those 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.

The results in Table 5.1 first show the performance in F1-score of the neural networks NNET-U and NNET-B. Alongside this, the best performing unsupervised document embedding (U-Embedding) used as input to a linear SVM is shown. These results serve as a reference point for the performance of the properties. The results for properties derived from these networks and this unsupervised document embedding are then shown when used as input to decision trees limited to a depth of three. If the decision tree results do indeed outperform or match the neural networks, then they are performing well.

To clarify the meaning of Table 5.1, the terms used are explained below:

- **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.

	Movies	Place-Types	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

- **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 good 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 gives credence to the idea that decision trees of depth three 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. with a high NDCG score) features of the disentangled representation derived from NNET-U will be predictive for 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 decision trees of depth three. 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. NNET-U performs 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. This can be understood by seeing 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 idea is expanded on 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 features 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 features 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 Feed-forward 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 initial 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 initial embedding, NNET-U and NNET-B are used in this section for each domain.

Terms are scored using NDCG 4.2.2 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 initial embedding and the NNET with the initial 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 initial 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

Movies		
Initial 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)
Place-Types		
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, canonef2470mmf28lusr)	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)	itches (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

embeddings. The intention is to see if NNET-U and NNET-B embeddings have introduced any new features 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 initial embedding, and only 19 for the neural network that used the initial embedding as input. Meanwhile, there were 770 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 initial 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 initial 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 terms are listed that have are high-scoring in the initial embedding, but low scoring in the NNET-U embedding. It is shown that terms that are not relevant to genres have a high score in the initial embedding, indicating that NNET-U decreases the importance of terms that are not relevant to the task of genres. For example, "yup", "wright", "wimpy" and "zoom", which are the top four highest score differences, do not seem related to

Movies			
Unique Initial	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 initial embedding, the neural network that used a bag-of-words as input and the neural network that used the unsupervised vector space as input. Arranged by NDCG, from highest to lowest.

genres at all. Additionally, when looking at the terms that have gained in NDCG score derived from NNET-U, they are clearly relevant to genres, e.g. "animation", "adventure", "comedies". This gives us some indication of what the neural network is doing to solve the task. The neural network seems to be decreasing separability of noisy terms and increasing separability of terms relevant to the task. This could explain the difference in score where NNET-U outperformed a linear SVM on the initial embedding. Although, it has also increased the separability of some metadata e.g. the term "listinfo", which is a problem already identified when using the initial embedding as input to the neural network.

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 initial 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)
yarns (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 domain. 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 task, 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 5.5, we examine the clustered features used in the decision trees for the Newsgroups task. 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 initial 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 features, 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, sun-view, phigs, pex, colormap, pixmap)" compared to the initial 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 5.5, 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	Initial 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)	(gemm, 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, tvrtwm, 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, ciphertxt, 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, simlless, moles, ..., corinthians, baptized)	(pastor, baptized, congregation, corinthians, repent, pagan)	(scripture)
talk.politics.guns	(federalist, unregistered, tyranny, ..., wright, shotguns)	(tpg, progun, 9mm, ..., fkba, shotguns)	(batf)
talk.politics.mideast	(arab, rebelllion, syrian, ..., zionist, gaza)	(asasadpaarf, revisionism, zionism, ..., hamas, grandparents)	(palestinian)
talk.politics.misc	(croats, redraw, shelling, ..., bosnians, yugoslavia)	(wiretapping, safeguards, visi)	(homosexual)
talk.religion.misc	(divorce, simlless, moles, ..., corinthians, baptized)	(thy, forgiveness, churchs, ..., protestant, ephesians)	(scripture)

Table 5.5: The top node of the Decision Tree for the Newsgroups task

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 features (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 initial embedding. This follows the previous observation as the "adventure" direction has a higher associated NDCG score in the NNET-U embedding versus the initial embedding, as seen in Table 5.4. 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 results in Table 5.5 for NNET-B, where instead new and meaningful cluster-features are found that are relevant to the task.

Movies	Initial 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, 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 initial 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)
ArtsAndEntertainment	NNET-B	(sugar mill, silo, corral, abandoned farm, ranch)
CollegeAndUniversity	(ruraldecay)	(commercial real estate, large construction, retirement home, ..., rental property)
Food	(investment)	(football field, training camp, strait, baseball stadium, continent)
NightlifeSpot	(dolphins)	(dive bar, beer garden, karaoke bar, hotel bar, cocktail bar)
ParksAndOutdoors	(booze, vodka, whiskey, liquor)	(piano bar, public housing, waiting room, residential street, dining room)
ProfessionalAndOtherPlaces	(condos, venice, entertainment, ..., socks, americanflag)	(sand bar, vineyard, monsoon forest, ski lodge, topless beach)
Residence	(gardening, tiki, cocktail, ..., candle, horticulture)	(battlefield, national monument, battlefield park, cliff dwelling, space shuttle)
ShopsAndService	(tribute, plains, des, acropolis, az)	(diocese, convent, tomb, college theater, study)
TravelAndTransport	(cloister)	(railroad tunnel, terminal, airport tram, railroad signal, railroad yard)
ArtsAndEntertainment	(dmctz3, panasonicdmctz3, panasonicitz3)	
CollegeAndUniversity	NNET-U	(cubicle, newsroom, workroom, bedroom closet, detached house)
Food	(messy, schoolhouse, beatles, ..., halftimbered, publictransport)	(video game store, theme park, space shuttle, launch pad, speedway)
NightlifeSpot	(snowboard, tourism, amusementpark, ..., waltdisney, disneyworld)	(molecular gastronomy restaurant, whisky bar, gourmet shop, cocktail bar, juice bar)
ParksAndOutdoors	(boutique, liquor, olive, ..., counter, dessert)	(dance studio, tattoo studio, shoulder, topless beach, strip club)
ProfessionalAndOtherPlaces	(shoulder, darkroom, topless, ..., boobs, lips)	(rookery, hunting reserve, wetland, wildlife reserve, nest)
Residence	(nesting, tits, mouth, ..., gulls, avian)	(retail outlet, electronics store, tourist information center, jewelry store, souk)
ShopsAndService	(islam, muslim, supermarket, ..., sales, marruecos)	(ventilation shaft, abandoned prison, sanatorium, abandoned complex, reformatory)
TravelAndTransport	(heath, ward, lock, ..., stained, argyll)	(video game store, jewelry store, thrift store, workroom, nail salon)
	(video, candy, clothing, ..., skin, blonde)	(crop farm, flowerbed, tongue, nest, crop circle)
	(survey, mammal, antenna, ..., paws, fur)	

Table 5.7: The Place-Types 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 initial 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 initial 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 does not change the representation enough to make these directions less separable, despite improving other directions so that the embedding better suits 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. This is expected, as NNET-B is not starting from a vector space, however, despite starting from a bag-of-words the directions that were common to all embeddings seemed to be the most relevant properties to the task. This explains the reason that the disentangled feature representations

Cluster Features	Top 5 Ranked Entities
Initial 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 initial embedding.

derived from NNET-U, NNET-B and the initial embedding all performed similarly: they all contained the directions that are most relevant to the task.

In Section 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 better represented terms that were not relevant to the task, while NNET-U focused on improving existing relevant directions. In Section 5.4.3 it was hypothesized that the reason that the disentangled feature representation derived from NNET-B did not result in overall higher performance in a low-depth decision tree is because although it had good performance on the "Adventure" genre, it did not capture any properties to classify the "Animation" genre. In summary, the feed-forward neural network that performed well refined directions relevant to the task in order to achieve a good classification accuracy, rather than finding new properties.

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 initial embedding. This explains the increase in performance for the cluster-features used in the decision tree, NNET-U achieves strong performance by focusing on improving the feed-forward neural network features, but NNET-B achieves strong performance when using the cluster features because it is able to find unique properties that are relevant to the task.

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 Evaluation of Auto-encoders

This section is separated into four parts. First, the parameters and methodology of the semantic features to be used in the experiments is explained in Section 5.5.1. Then, the auto-encoders are quantitatively evaluated in Section 5.4.2 to determine how good the derived semantic features are in each layer. In summary, it is found that these derived semantic features do not perform well, but they seem to contain interesting properties. In particular, it seems that as more layers are added, the more general the resulting features seem to be. These features are further investigated qualitatively in Section 5.5.3 and validated quantitatively in Section 5.5.4, where it is found that the deeper the layers of the auto-encoder are the more frequent highly separable properties become.

5.5.1 Software, Architecture and Settings

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.

To implement the denoising auto-encoders, the Keras¹ library is used. As in Chapter 4, scikit-learn is used for the SVM implementation. 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

¹Keras

Clusters	Genres		Keywords		Ratings	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
L0	0.91	0.43	0.71	0.22	0.65	0.49
L1	0.64	0.27	0.63	0.17	0.64	0.41
L2	0.71	0.32	0.65	0.19	0.64	0.44
L3	0.66	0.27	0.62	0.17	0.61	0.44
L4	0.66	0.26	0.54	0.15	0.55	0.40

Table 5.9: The results of depth-3 trees for a variety of tasks in the movie domain where cluster representations derived from auto-encoder layers are used as input.

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 centers in every space.

5.5.2 Quantitative Evaluation of Cluster Representation

This section evaluates if it is possible to obtain semantically meaningful properties from the hidden layers of stacked denoising auto-encoders following the same procedure as in Section 5.4.2. A representation composed of cluster properties is obtained from the hidden layers of the stacked denoising auto-encoder described in Section 5.5.1. Then these representations are used as input to low-depth decision trees to classify the genres, keywords and ratings tasks for the Movies domain. The results are shown in Table 5.9, where it can be seen that the accuracy and F1-scores of the properties obtained from the hidden layers of stacked auto-encoders are significantly lower than those properties obtained from the original representation. This indicates that the hidden layers of these auto-encoders do not represent key domain properties. However, when observing examples these properties do seem to be meaningful. The nature of these properties is further analysed in Section 5.5.3.

5.5.3 Qualitative Evaluation of Cluster Representations

In this section, the properties induced from the auto-encoder spaces described in Section 5.5.2 are qualitatively investigated. In Table 5.10, 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 different properties across their layers, but the properties obtained when using a denoising auto-encoder seem to be significantly different. In particular, they seem to be increasingly general, i.e. used in multiple documents. For example, the normal auto-encoder captures properties like "Horror" and "Thriller", but does not capture properties 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 idea of budget "Horror" movie, which is significantly different to a "Zombie" movie. This change, from "Zombie" to "Effects" seems to be a difference between a low-frequency word that is specific to only some movies "Zombie", and a high-frequency word that is meaningful for many movies. In Section 5.5.4 the relationship between properties in the earlier layers and those in the later ones is more precisely quantified as a relationship of frequency, i.e. relevancy to many documents rather than few.

5.5.4 Quantitative Frequency Evaluation

Although there is an indication that the properties in the later layers of the stacked denoising auto-encoder change in a particular way, it is still unclear in what way exactly they change. Following the qualitative analysis in Section 5.5.3 the results seem to indicate later properties are more frequent across the corpus, while the earlier properties are less frequent. In other words, the earlier representations model properties that are more specific, in the sense that they are relevant to only some documents, and the later representations model properties that are more general, in the sense that they are relevant to more documents.

Standard Auto-encoder		Denoisng 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 5.10: A comparison between the first layers and the fourth layers of two different kinds of auto-encoders

# Of Cluster Words	L0 Freq	L1 Freq	L2 Freq	L3 Freq	L4 Freq
First 1	23298	29984	37277	41952	36489
First 2	20399	25356	31184	37764	35785
First 3	17065	22595	30465	36935	34893
First 4	14986	20555	28780	36798	35107
First 5	13352	18982	27724	35446	34705

Table 5.11: The average frequency of cluster words in the corpus. Here, the frequencies of the first words in the cluster are shown.

To evaluate the frequency of the clusters, the average frequency of each term in the cluster is used. However, the first terms of a cluster are generally more separable in the space, as the highest score terms are added to the cluster first. Given this, the first N terms of each cluster for a layer of the auto-encoder are averaged rather than all terms. There is a maximum cluster size, and all terms must be added, so taking the top N terms also removes the problem of irrelevant and less separable terms being included.

To determine the frequency of an auto-encoder layer, for each cluster the first N words are taken and their frequencies are averaged. This is the cluster frequency. Then the overall frequency of the auto-encoder layer is obtained by taking the average of all clusters. This overall frequency for each layer is shown in Table 5.11 for values of N from 1 to 5. Additionally, results for the first N words at different thresholds are shown. In accordance with the observations in 5.5.3, we can see that the best-modelled terms in the later auto-encoder layers tend to have a higher frequency than the first layer. We can also see that there is a drop in average frequency from layer 4 to layer 3, which can be explained from the fact that the representations in higher layers also become noisier, which leads to more randomness in the terms that are selected.

5.6 Inducing Rules from Auto-Encoder Entity Embeddings

As described in the previous section, stacked denoising auto-encoders model increasingly frequent properties. Based on this observation, in this section a method is introduced 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. In this setting, attributes are discretized as understanding what is meant by the rules is the main focus. These results are not intended for making predictions, but only for getting insight into data and potentially generating explanations of domain knowledge.

5.6.1 Qualitative Evaluation of Induced Symbolic Rules

In this section, the method used to obtain rules that describe the semantic relationships between properties derived from the stacked denoising auto-encoder is qualitatively investigated. 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.

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 "Scares₂". Similarly, a condition such as `funniest-L2 >= 22`, which indicates that the property is not in the top 22%, will be written as "NOT Funniest₂". In this simpler notation the above rule will look as follows:

```
IF Scares2 AND Blood2 AND NOT Funniest2 THEN Gore3
```

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 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 Flick2 AND Sexual2 AND Cheesy2 AND NOT Amazon2 THEN Nudity3
```

```
Flick2: {Flicks, Chick, Hot}
```

```
Amazon2: {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 Emotions2 AND Actions2 THEN Emotions3
```

```
IF Emotions2 AND Emotion2 AND Impact2 THEN Journey3
```

```
IF Emotions3 AND Journey3 THEN Adventure4
```

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 Romance1 AND Poignant1 AND NOT English1 AND NOT French1
AND NOT Gags1 AND NOT Disc1 THEN Relationships2
```

```
IF Relationships2 AND Emotions2 AND Chemistry2 THEN Romantic3
```

```
IF Emotions2 AND Compelling2 THEN Beautifully3
```

```
IF Warm2 AND Emotions2 THEN Charming3
```

```
IF Emotions2 AND Compelling2 THEN Emotional3
```

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

```
IF Touching3 AND Emotions3 AND NOT Unfunny3 THEN Relationship4
IF Laughs3 AND Laugh3 AND NOT Compelling3 THEN Stupid4
IF Touching3 AND Social3 AND NOT Slasher3 THEN Touching4
```

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 Emotions2 AND Actions2 THEN Emotions3
```

```
Emotions2: {Acted, Feelings, Mature}
```

```
Actions2: {Control, Crime, Force}
```

```
Emotions3: {Emotion, Issue, Choices}
```

```
IF Horror2 AND Creepy2 AND Scares2 THEN Horror3
```

```
Horror2: {Terror, Horrific, Exploitation}
```

```
Creepy2: {Mysterious, Twisted, Psycho}
```

```
Scares2: {Slasher, Supernatural, Halloween}
```

```
Horror3: {Creepy, Dark, Chilling}
```

```
IF Touching2 AND Chemistry2 THEN Touching3
```

```
IF Touching2 AND Emotions2 THEN Touching3
```

```
IF Compelling2 AND Emotional2 AND Suspense2 THEN Compelling3
```

```
IF Romance2 AND Touching2 AND Chemistry2 THEN Romance3
```

```
IF Emotionally2 AND Emotions2 AND Compelling2 THEN Emotionally3
```

5.7 Conclusion

Feed-forward 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.

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 document 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 an 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-centered objective, e.g. Multi-Dimensional Scaling (see Section 2.6.2), 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 based on 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".

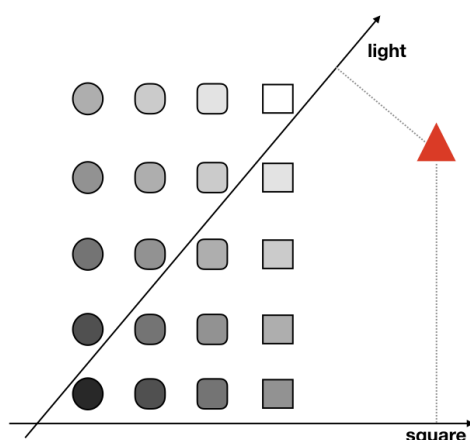


Figure 6.1: Toy example showing the effect of outliers in a two-dimensional embedding of geometric shapes.

Ideally, representations would be learned with knowledge of feature-directions. For example, the method to learn a representation of the toy domain would take into account 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. In other words, 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 Movies domain 3.2 Multi-Dimensional Scaling (MDS) space 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. This chapter introduces a method that can solve this problem by attempting to match its ranking in the vector space that is very high in the initial space, to its bag-of-words value, which is zero. This results in this outlier movie being moved

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.

down drastically in the rankings.

To give some additional examples, in Table 6.1, names of Place-Types (that correspond to text documents in the Place-Types dataset) are shown ranked on feature directions from their domain (See Section 3.2). In these examples, for the cluster-feature $\{steep, climb, slope\}$, the top ranked Place-Type *mountain* is clearly relevant. However, the next two Place-Types — *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 the problems described in the previous section, 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 documents as a kind of weak supervision signal: if a document 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 x we determine a total ordering \preceq_x such that $d \preceq_x d'$ iff the feature x is more prominent in the BoW representation of document d' than in the BoW representation of d . We will refer to \preceq_x 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 $d \preceq d'$ iff $\mathcal{D}_c \cdot v_d \leq \mathcal{D}_c \cdot v_{d'}$. Since this will typically not be the case, we subsequently determine *target values* for the dot products $\mathcal{D}_c \cdot v_d$. 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 feed-forward neural network to adapt the semantic space representations v_d and feature directions \mathcal{D}_c to make the dot products $\mathcal{D}_c \cdot v_d$ as close as possible to these target values.

6.2.1 Generating Target Rankings

Let $CD = (\mathcal{D}_{c_1}, \dots, \mathcal{D}_{c_j})$ be the cluster directions that were found using the method from Section 4.2.1. Each cluster c_i typically corresponds to a set of semantically related words $c = (w_1, \dots, w_n)$, which describe some salient feature from the considered domain. From the BoW representations of the documents, we can now define a ranking that reflects how strongly each document is related to the words from this cluster. To this end, we represent each document as a bag-of-clusters (BoC) and then compute PPMI scores over this representation. In particular, for a cluster $c = (w_1, w_2, \dots, w_n)$, we define $fc(c, d) = \sum_{i=1}^n f(w_i, d)$. In other words, $fc(c, d)$ is the total number of occurrences of words from cluster c in the BoW representation of d . We then write $ppmi(c, d)$ for the PPMI score corresponding to this BoC representation, which is evaluated in the same way as $ppmi(w, d)$, but using the counts $fc(c, d)$ rather than $f(w, d)$. The target ranking for cluster c_i is then such that d_1 is ranked higher than d_2 iff $ppmi(c_i, d_1) > ppmi(c_i, d_2)$. 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 document. For instance, a document describing a violent movie might not actually mention the word "Violent", but would likely mention 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 hard¹. 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 $\mathcal{D}_{c_j} \cdot v_{d_i}$ from these target rankings. One straightforward approach would be to use the PPMI scores $ppmi(c_j, d_i)$. 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 $\tau(c_j, d_i)$ for the dot product $\mathcal{D}_{c_j} \cdot v_{d_i}$, which respect the ranking induced by the PPMI scores, but otherwise remain as close

¹It is complete for the complexity class $\exists\mathbb{R}$, which sits between NP and PSPACE [74].

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 $d_{\sigma_1}, \dots, d_{\sigma_n}$ be an enumeration of the documents such that $ppmi(c_j, d_{\sigma_i}) \leq ppmi(c_j, d_{\sigma_{i+1}})$ for $i \in \{1, \dots, m-1\}$. The corresponding target values $\tau(c_j, d_i)$ are then obtained by solving the following optimization problem:

$$\begin{aligned} \text{Minimize: } & \sum_i (\tau(c_j, d_i) - \mathcal{D}_{c_j} \cdot v_{d_i})^2 \\ \text{Subject to: } & \tau(c_j, d_{\sigma_1}) \leq \tau(c_j, d_{\sigma_2}) \leq \dots \leq \tau(c_j, d_{\sigma_m}) \end{aligned} \quad (6.1)$$

6.2.3 Fine-Tuning

We now use the target values $\tau(c_j, d_i)$ 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 $v_{d_1}, \dots, v_{d_m} \in \mathbb{R}^k$. These are fed into a layer of size l :

$$h_i = f(Wv_{d_i} + b)$$

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 i^{th} document. 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 $g_i = (g_i^1, \dots, g_i^K)$ 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 $\tau(c_j, d_i)$, i.e.:

$$\mathcal{J} = \sum_i \sum_j (g_i^j - \tau(c_j, d_i))^2$$

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 documents are now more closely related to the feature, despite being less common, and outliers such as "fence" no longer appear.

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 AWV _w	0.41	0.41	0.43
AWV _w	0.38	0.40	0.43
LDA	0.40	0.37	0.35

Table 6.2: Results for 20 Newsgroups.

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 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. Recall that depth 1 trees only use 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. As in all other experiments, 20% of the training data is taken for hyper-parameter tuning. As we are not tuning as many parameters and we only have a small number of documents for the Place-type domain, we use 5-fold cross validation in this domain

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.

for these experiments.

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 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, then it is 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 is 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 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 here we will fix this value in advance. 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 dataset, 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 [23], with default values, and the model was implemented in the Keras library.

To learn the decision trees, we use the scikit-learn implementation of CART, which allows us to limit the depth of the trees. We will consider experiments with depth-1 trees, depth-3 trees, and with unbounded trees. 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-tuning². 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 semantically meaningful feature 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,

²Since 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.

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 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 features, possibly at the expense of other aspects.

6.4 Conclusions

We have introduced a method to fine-tune an existing document embedding in order to obtain more semantically coherent features. 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 documents 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.

Conclusion

7.1 Introduction

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

7.2 Thesis Summary and Contributions

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

Semantically meaningful features can be obtained from vector space representations of documents. These features are sufficiently predictive to be useful for simple interpretable classifiers, of which low-depth decision trees are a prototypical example, allowing for a performance that is close to the performance of an unbounded decision tree using the same features as input. Further, these feature can be obtained from neural network hidden layers, and can provide valuable insights into aspects of the domain that the neural network has learned.

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 as semantic features for interpretable classifiers, especially as the classifiers in some cases performed better than linear SVMs trained on all features of the original unsupervised embedding. 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 for use as a scoring metric for directions in this work was found to generally be a good choice, in particular because it had the most consistent performance when the resulting features were used as input to low-depth decision trees. Varying the clustering method was found to be successful in some cases.

In Chapter 5 feed-forward and auto-encoder neural networks were quantitatively and qualitatively investigated. Characteristics of feed-forward networks that used BOW as input were identified and compared with unsupervised representations. For feed-forward networks, we found that higher-quality feature directions could be found, for those features that were relevant to the classification task on which the network was trained. Furthermore, we noticed that terms that were less relevant to the classification task scored less. Auto-encoders with increasingly low-dimensional embeddings were qualitatively and quantitatively investigated, and despite not being effective at learning natural categories, they seemed to capture increasingly abstract features. This was validated quantitatively. Finally, a rule-based classifier that connected the features from layers together was used to further investigate what the neural network was doing.

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 discusses how they were answered in the thesis.

Question 1: Can meaningful semantic features be characterised as directions across a wide range of vector space encodings and domains, and do these semantic features allow us to learn effective low-depth decision trees?

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 trees 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:** Can semantic features be obtained from the hidden layers of neural networks, and to what extent can these features be used to investigate the characteristics of different neural networks?*

From the qualitative analysis of the hidden layer representations in Chapter 5, it was clear that the properties derived from these representations was related to how well they classified the task. Quantitatively, properties derived from feed-forward networks when used as input to low-depth decision trees allowed them to achieve good performance classifying key domain tasks. 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 that they reduced how well terms unrelated to the task are captured in the space. Additionally, increasingly low-dimensional auto-encoders were found to capture increasingly abstract properties.

***Question 3:** Is it possible to obtain higher-quality semantic features in an unsupervised way by fine-tuning the initial vector space?*

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 vector spaces, 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 words and the words associated with its most similar directions) is given an intruder word from another cluster. The feature representation performs well if users 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. 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 might seem straightforward (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 to eXplainable AI (XAI), methods that explain existing learned neural network models. Essentially, the approach outlined in Section 5 would be used to create a "proxy model" that approximates the 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 disentangled feature representations be derived from neural networks that achieve state-of-the-art e.g. BERT [22]? BERT is a neural network that achieves strong results on a variety of tasks and has entered mainstream application usage, but can the hidden layers of BERT be disentangled into interpretable features, and can an interpretable classifier be learned? 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 SVMs were used to obtain hyperplanes that classify 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 documents were ranked on it using the dot product. It would be interesting to instead try RankSVM [50] 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 cluster features was identified, in particular that their meaning may be disrupted due to terms unrelated to the task being clustered with them. 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 are 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 of fine-tuning the representation it stopped learning information related to the task. 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] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete Problems in AI Safety. pages 1–29, 2016.
- [2] 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.
- [3] 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.
- [4] 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.
- [5] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.
- [6] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.
- [7] Christopher M CM Christopher M. Bishop. *Pattern Recognition and Machine Learning*, volume 4 of *Information science and statistics*. Springer, 2006.
- [8] David M. Blei and John D. Lafferty. Correlated Topic Models. *Advances in Neural Information Processing Systems 18*, pages 147–154, 2006.
- [9] David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003, 2003.

-
- [10] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. 2015.
- [11] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.
- [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] 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.
- [14] Xi Chen, Yan Duan, Rein Houthoofd, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. 2016.
- [15] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. pages 76–83, 1989.
- [16] William Cohen. Fast effective rule induction. *Twelfth International Conference on Machine Learning: 1995*, 1995.
- [17] Douglas E Critchlow. *Metric methods for analyzing partially ranked data*, volume 34. Springer Science & Business Media, 2012.
- [18] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian LDA for topic models with word embeddings. In *Proc. ACL*, pages 795–804, 2015.
- [19] Vin De Silva and Joshua B Tenenbaum. Sparse multidimensional scaling using landmark points. 2004.
- [20] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

- [21] Joaquin Derrac and Steven Schockaert. Inducing semantic relations from conceptual spaces: A data-driven approach to plausible reasoning. *Artificial Intelligence*, 228:66–94, 2015.
- [22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186, June 2019.
- [23] 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.
- [24] Robert M Fano. Transmission of information: A statistical theory of communications. *American Journal of Physics*, 29:793–794, 1961.
- [25] Manaal Faruqui and Chris Dyer. Non-distributional Word Vector Representations. *Acl-2015*, pages 464–469, 2015.
- [26] Peter Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. MIT press, 2004.
- [27] David Gefen, James E Endicott, Jorge E Fresneda, Jacob Miller, and Kai R Larsen. A guide to text analysis with latent semantic analysis in r with annotated code: Studying online reviews and the stack exchange community. *Communications of the Association for Information Systems*, 41(1):21, 2017.
- [28] Stuart Geman, Elie Bienenstock, and René Doursat. Neural Networks and the Bias/Variance Dilemma, 1992.
- [29] 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.
- [30] 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.
- [31] 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.

- [32] Kevin Gurney. *An introduction to neural networks*. CRC press, 1997.
- [33] Ulrike Hahn and Nick Chater. Similarity and rules: distinct? exhaustive? empirically distinguishable? *Cognition*, 65:197 – 230, 1998.
- [34] 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.
- [35] Thomas Hofmann. Probabilistic latent semantic indexing. *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1999*, pages 50–57, 1999.
- [36] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing Deep Neural Networks with Logic Rules. *arXiv preprint*, pages 1–18, 2016.
- [37] Shoaib Jameel, Zied Bouraoui, and Steven Schockaert. Member: Max-margin based embeddings for entity retrieval. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 783–792, 2017.
- [38] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013.
- [39] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.
- [40] Ian T Jolliffe. Rotation of principal components: choice of normalization constraints. *Journal of Applied Statistics*, 22(1):29–35, 1995.
- [41] D Kim and J Lee. Handling continuous-valued attributes in decision tree with neural network modeling. *Machine Learning: Ecml 2000*, 1810:211–219, 2000.
- [42] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *International Conference on Machine Learning*, pages 1857–1865. PMLR, 2017.

- [43] Ralf Klabunde. Daniel Jurafsky/James H. Martin: Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. *Zeitschrift fur Sprachwissenschaft*, 21(1):134–135, 2002.
- [44] Aykut Koc, Lutfi Kerem Senel, Ihsan Utlu, and Haldun M Ozaktas. Imparting Interpretability to Word Embeddings while Preserving Semantic Structure. pages 1–11.
- [45] Adriana Kovashka, Devi Parikh, and Kristen Grauman. WhittleSearch: Interactive Image Search with Relative Attribute Feedback. *International Journal of Computer Vision*, 115(2):185–210, 2015.
- [46] Suyash Lakhota and Xavier Bresson. An Experimental Comparison of Text Classification Techniques. *2018 International Conference on Cyberworlds (CW)*, pages 58–65, 2018.
- [47] Kai R Larsen, David E Monarchi, Dirk S Hovorka, and Christopher N Bailey. Analyzing unstructured text data: Using latent categorization to identify intellectual communities in information systems. *Decision Support Systems*, 45(4):884–896, 2008.
- [48] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. *31st International Conference on Machine Learning, ICML 2014*, 4:2931–2939, 2014.
- [49] 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.
- [50] Ching Pei Lee and Chih Jen Lin. Large-scale linear rankSVM. *Neural Computation*, 26(4):781–817, 2014.
- [51] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.
- [52] David D. Lewis. Evaluating and optimizing autonomous text classification systems. *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, pages 246–254, 1995.
- [53] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Topical word embeddings. In *Proc. AAAI*, pages 2418–2424, 2015.

- [54] Hongyin Luo, Zhiyuan Luan, Liuv Huanbo, and Maosong Sun. Online learning of interpretable word embeddings. *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, (September):1687–1692, 2015.
- [55] Andrew L Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, 2011.
- [56] Christopher D. Manning, Hinrich Schütze, and Gerhard Weikurn. Foundations of Statistical Natural Language Processing. *SIGMOD Record*, 31(3):37–38, 2002.
- [57] 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.
- [58] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A Survey on Bias and Fairness in Machine Learning. 2019.
- [59] 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.
- [60] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. *Nips*, pages 1–9, 2013.
- [61] Brian Murphy, Partha Pratim Talukdar, and Tom Mitchell. Learning effective and interpretable semantic models using non-negative sparse embedding. *24th International Conference on Computational Linguistics - Proceedings of COLING 2012: Technical Papers*, (December 2012):1933–1950, 2012.
- [62] 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.

- [63] 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.
- [64] 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.
- [65] V Paul Pauca, Fariyal Shahnaz, Michael W Berry, and Robert J Plemmons. Text mining using non-negative matrix factorizations. In *Proceedings of the 2004 SIAM International Conference on Data Mining*, pages 452–456. SIAM, 2004.
- [66] 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.
- [67] Radim Řehřek, Petr Sojka, et al. Gensim-statistical semantics in python. *Retrieved from genism.org*, 2011.
- [68] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.
- [69] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16*, pages 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery.
- [70] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, and Phil Blunsom. Reasoning about Entailment with Neural Attention. (2015):1–9, 2015.
- [71] 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.
- [72] Emad W. Saad and Donald C. Wunsch. Neural network explanation using inversion. *Neural Networks*, 20(1):78–93, 2007.

- [73] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613â620, November 1975.
- [74] 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.
- [75] Rudy Setiono and Huan Liu. Neurolinear: From neural networks to oblique decision rules. *Neurocomputing*, 17(1):1–24, 1997.
- [76] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.
- [77] Ms Sorower. A literature survey on algorithms for multi-label learning. *Oregon State University, Corvallis*, (March):1–25, 2010.
- [78] Karen Sparck Jones. *A Statistical Interpretation of Term Specificity and Its Application in Retrieval*, page 132â142. Taylor Graham Publishing, GBR, 1988.
- [79] Mark Steyvers. Multidimensional scaling. *Encyclopedia of cognitive science*, 2006.
- [80] Anant Subramanian, Danish Pruthi, Harsh Jhamtani, Taylor Berg-kirkpatrick, and Eduard Hovy. SPINE: SParse Interpretable Neural Embeddings. 32(1):4921–4928, 2018.
- [81] Matt Taddy. Document Classification by Inversion of Distributed Language Representations. *Proceedings of the 53rd meeting of the Association for Computational Linguistics (ACL'15)*, pages 45–49, 2015.
- [82] 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.
- [83] Valentin Trifonov, Octavian-Eugen Ganea, Anna Potapenko, and Thomas Hofmann. Learning and evaluating sparse interpretable sentence embeddings. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for*

- NLP*, pages 200–210, Brussels, Belgium, November 2018. Association for Computational Linguistics.
- [84] Valentin Trifonov, Octavian-Eugen Ganea, Anna Potapenko, and Thomas Hofmann. Learning and evaluating sparse interpretable sentence embeddings. 2018.
- [85] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.
- [86] Amos Tversky. Features of similarity. *Psychological review*, 84:327–352, 1977.
- [87] Berk Ustun and Cynthia Rudin. Methods and Models for Interpretable Linear Classification. *arXiv*, pages 1–57, 2014.
- [88] P. Viappiani, B. Faltings, and P. Pu. Preference-based Search using Example-Critiquing with Suggestions. *Journal of Artificial Intelligence Research*, 27:465–503, 2006.
- [89] 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.
- [90] 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.
- [91] Lipo Wang. *Support vector machines: theory and applications*, volume 177. Springer Science & Business Media, 2005.
- [92] 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.
- [93] 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.
- [94] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, 32(NeurIPS):1–18, 2019.
- [95] 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.
- [96] Jan Ruben Zilke, Eneldo Loza Mencía, and Frederik Janssen. DeepRED - 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.
- [97] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.
- [98] Hui Zou and Lingzhou Xue. A selective overview of sparse principal component analysis. *Proceedings of the IEEE*, 106(8):1311–1320, 2018.