Exploiting the Properties of Word Embeddings to Improve the Representations of Word Sequences

Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Engineering

by

Narendra Babu Unnam
2019801005
narendra.babu.unnam@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
July 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " **Exploiting the Properties of Word Embeddings to Improve the Representations of Word Sequences**" by **Narendra Babu Unnam**, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Prof. P. Krishna Reddy

To
My Parents

Acknowledgments

As I submit my Ph.D. thesis, I would like to take this opportunity to acknowledge all the people who helped me in my journey at IIIT-Hyderabad.

Firstly, I would like to express my deepest gratitude to my guide, Prof. P. Krishna Reddy, whose invaluable support and guidance were instrumental in bringing this work to fruition. I cannot overstate the importance of his consistent encouragement and unwavering assistance throughout the process of formulating the problem statement and writing the thesis, and providing tuition support, RA support, and travel grants during my Ph.D. Even in moments of uncertainty, he remained patient and open-minded, always providing me with the right suggestions and insights. His remarkable passion and dedication to his work serve as a shining example and inspiration for me. Working with him has taught me the significance of addressing real-world problems, how to conduct thorough research, and effectively present ideas in the paper. Beyond all of this, I greatly admire his remarkable qualities of patience and kindness toward all those around him. I aspire to have imbibed some of these traits myself through our interactions.

I am grateful to Prof. Naresh Manwani for his pivotal assistance and input in advancing my research work. I would also like to extend my appreciation to my co-author, Amith Pandey, for his invaluable feedback and suggestions in refining our ideas and improving our work.

I would like to express my gratitude to my colleagues at DSAC, namely Kumaraswamy, Mamatha, Srinivas Reddy, Saideep, Aravinda, Haranadh, Harshita, Vipul, Bhoomendra, Abinash, and Chandrashekar, with whom I have spent numerous hours having both productive discussions and casual conversations. I have enjoyed each one of your company and I loved us as a team when executing projects, conducting conferences, or any other social event. I would always cherish the memories we shared during these times. I deeply appreciate your unwavering support, especially in trying times, and wish you the best of luck in your future endeavors and continued professional growth.

I acknowledge the financial support provided by the Crop Darpan project, which is sponsored jointly by the Department of Science and Technology, India (DST) and the Japan Science and Technology Agency (JST). Additionally, I am grateful for the financial support provided by the Indian Legal Information System project through iHub Anubhuthi during my Ph.D. tenure.

Finally, I would like to express my gratitude to my parents, Rama Devi and Jagan Mohan Rao, for their unconditional love and infinite patience in enabling me to follow my aspirations. I owe a debt of gratitude to my brother, Nava Krishna, and sister-in-law, Lavanya, for their constant support, and finally, to Akshara, for bringing everything and everyone together.

Abstract

In the current digital era, about 80% of the digital data which is being generated is unstructured and unlabeled natural language text. Research efforts are going on for improving text mining techniques to automatically organize, analyze, and extract useful information from the voluminous text data. In the development cycle of information retrieval and text mining applications, *text representation* is the most fundamental and critical step as its effectiveness directly impacts the performance of the application. Three important properties of the text representation models, which make them suitable for practical applications, are representational power, interpretability of features, and unsupervised learnability.

Words and word sequences (such as sentences and documents) are the natural units of the text which are subjected to vector representation to handle the given text data. With the advent of deep learning, a family of neural language models emerged to represent words as low dimensional, distributed, and dense vectors. These word representations are popularized as *word embeddings*. Word embeddings are learned from huge corpora, so they encode a lot of information and have high expressive power. In the case of word sequences, traditional methods provide interpretability and support unsupervised learning, but they suffer from the issues of low representational power and scalability. The deep learning based word embedding methods are successful in producing vectors with high representational power. However, most of the deep learning based methods are notorious for human uninterpretabilty and work only in supervised learning environments.

In this thesis, we propose improved word sequence representation models by exploiting the frequency distributional and spatial distributional properties of the word embeddings. Firstly, we propose an alternative word sequence representation framework in the context of longer texts such as documents. The existing vector averaging based models represent the document as a position vector in the same word embedding space. As a result, they are unable to capture the multiple aspects as well as the broad context in the document. Also, due to their low representational power, the existing approaches perform poorly at document classification. Furthermore, the document vectors obtained using such methods have uninterpretable features. In this work, we propose an improved document representation framework that captures multiple aspects of the document with interpretable features. In this framework, instead of representing a document in word embedding space, it is represented in a distinct feature space where each dimension is associated with a potential feature word that has relatively high discriminatory power. A given document is modeled as the distances between the feature words and the document. We have proposed two criteria for the selection of potential feature words and a distance function to measure the distance between the feature word and the document. Experimental results on multiple datasets

show that the proposed model consistently performs better at document classification over the baseline methods.

Secondly, we propose a weighted averaging based word sequence embedding method in the context of shorter texts such as sentences. The proposed weighting scheme captures the contextual diversity of words based on their geometry in the embedding space. It is observed that the simple unweighted vector averaging ignores the discriminative power of words and treats all the words as equal in the given sentence. Assigning the weights to words based on their discriminative power helps in building better word sequence representations. Recent literature introduced word weighting schemes based on the word frequency distribution into the simple averaging model. The frequency-based weighted averaging models augmented with the denoising steps are shown to outperform many complex deep learning models. However, these frequency-based weighting schemes derive the word weights solely based on their raw counts and ignore the diversity of contexts in which these words occur. The proposed weighting algorithm is simple, unsupervised, and non-parametric. Experimental results on semantic textual similarity tasks show that the proposed weighting method outperforms all the baseline models with significant margins and performs competitively to the current frequency-based state-of-the-art weighting approaches. Furthermore, as the frequency distribution-based approaches and the proposed word embeddings geometry-based weighting approach capture two different properties of the words, we define hybrid weighting schemes to combine both varieties. We also empirically demonstrate that the hybrid weighting methods perform consistently better than the corresponding individual weighting schemes.

Overall, we have proposed a new word sequence representation framework and weighting scheme by exploiting the geometrical properties of word embeddings. Combined with existing frequency based approaches, the proposed spatial geometry based framework exhibits a potential for better representation of word sequences which will improve the performance of text mining based applications in diverse domains.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the current information age, about 80% of the generated data is unstructured and unlabeled natural language text [31, 90]. The most common text sources in organizations and the Internet are emails, social media posts and messages, product reviews, survey responses, news portals, blogs, and crowd-sourcing platforms such as Wikipedia. Several information retrieval and text mining techniques are being investigated to automatically organize, analyze, and extract useful information from the voluminous text data. The most popular information retrieval and text mining techniques include information search, clustering, classification, summarization, topic detection, and sentiment analysis. Using these techniques, massive information systems such as search engines, social network platforms, e-commerce websites, crowd-sourcing platforms, recommendation systems, and decision support systems are being operated, which have become an integral part of our current society.

Typically, the development life cycle of information retrieval and text mining based systems involves with four-step pipeline: pre-processing of the data, *text representation*, designing an algorithm for the task at hand, and finally, the evaluation of the results. The model employed for text representation is an essential aspect as it directly affects the system's performance. Text representation is the process of generating numerical representations of raw text units such as words, sentences, and documents so that they are mathematically computable. Text representation models involve extracting the features which encode the information about the given text units. The most important properties of the text representation models which make them suitable for practical applications are representational power, interpretability of features, and unsupervised learnability.

In recent years, with the advent of neural network models, a family of neural language models [63, 70, 54] have been proposed to represent the words as position vectors or *word embeddings* in low dimensional feature spaces. The word embeddings are the result of internal weights of the neural network models, which implicitly learn about the words co-occurrence counts information [16, 88]. The intrinsic nature of these word embeddings is that the words which are contextually similar are closer in the corresponding feature space. The recent word embeddings models [63, 70, 54, 16, 88] are typically trained on huge text corpora. So the resultant vectors encode many linguistic regularities [65] along with substantial syntactic and semantic information. Due to the rich information encoded in these word

embeddings, they are used in multiple downstream tasks such as dependency parsing, named entity recognition, parts of speech tagging, text clustering, and text classification.

Now, many works are focusing on extending the successful concept of embeddings to word sequences such as sentences and documents by defining the composition functions. A composition function takes the word sequences and the corresponding embeddings of words as input and produces embeddings for word sequences. Most of the existing methods are not suitable for practical applications as they suffer from issues such as low representational power, uninterpretable features, and lack of support for unsupervised environments. In this thesis, we propose two improved methods to obtain the word sequence representations by exploiting the frequency distributional and spatial distributional properties of the word embeddings.

In the rest of this section, firstly, we will discuss the background of text representation. Secondly, we discuss the observations and research opportunities, and finally, we present the overview of the proposed models.

1.1 Background

1.1.1 Text representation

In the current digital era, the amount of unstructured and unlabeled textual data is growing at a rapid speed. Many information systems are being built either to manage or to extract knowledge from the textual data. A conceptual framework of a text information system [82] is shown in Fig. 1.1. As shown in the figure, first, the raw text data is refined with pre-processing steps. Next, the refined text is passed to representation models to convert the text to a numeric form like a vector of numbers. Finally, the text representations are passed as input to machine learning algorithms designed to organize the text data in various tasks such as classification, clustering, summarization, topic analysis, etc. The performance of the system over these tasks not only depend upon the algorithms used but also on the way the text is represented. Words and word sequences such as sentences and documents are basic and popular units to conduct text representation. The fundamental aim of text representation is to represent each unit as a point in a vector space, where the distance between each pair of units in the space is well-defined, and the distance between any two text units reflects the semantic relationship between them. Apart from modeling the semantic relationships, important properties of the text representation model are high representational power, unsupervised learnability, and interpretability of features which are explained in detail below.

- **Representational power:** The quality of the text representations is determined by their representational power. Good representations should be expressive [13], meaning that they should capture many underlying characteristics of any given text unit. Significant representational power can be achieved by choosing the features which can effectively describe and distinguish the given text

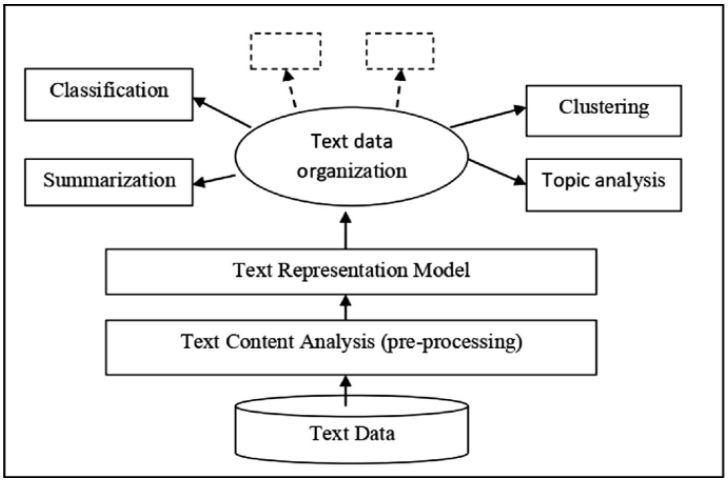

Figure 1.1: A conceptual framework of text information system

unit from another. Overall, the performance of any text mining application largely depends on its text representational power.

- **Unsupervised learnability:** Large percentage of the textual data that is generated is unlabeled, i.e., the class labels for most of the generated data are not available. Manual labeling of the text data is very labor intense and costly. In this scenario, the supervised learning based models which require class labels have limited scope. So, the development of unsupervised learning models is vital for practical text mining applications.

- **Interpretability:** In any practical application scenario, a typical user will always prefer solutions that are interpretable and understandable. Users often mistrust the black-box models as they don't allow the users to interpret the process of decision making. Moreover, if a model is not human interpretable, it is hard to find whether the model is biased and exhibits inequality. It is observed that several cases employing black box models have resulted in bias. For example, it has been reported that Amazon's AI-based recruiting tool showed bias against women [1]. Furthermore, in the USA, it was reported that predictions by a widely used criminal risk assessment tool are unreliable and racially biased [41]. To prevent this kind of incidents from happening, European Parliament adopted a regulation named "right to explanation" in April 2016 to enable a user to ask for an explanation of an algorithmic decision that was made about them [33]. In summary, learning interpretable data representations is becoming ever more important as applications of AI are being employed in critical domains like social, economic, and healthcare.

1.1.2 Word representation models

Words are the smallest meaningful units of natural language [57]. Larger text units of language such as phrases, sentences, and documents are further composed of words. To understand a language,

knowing the meanings of the words has a key role. So, to make computers automatically do the text mining related tasks, it is very important to accurately represent the words.

Traditionally to represent the words one-hot encoding is used. In this representation system, each word is assigned to a dimension in finite vector space. The length of the word vector is equal to the size of the vocabulary. Each word gets a binary vector representation where only one element which corresponds to the given word's dimension has a value 1, and the elements corresponding to the rest of the dimensions are 0. A sample one-hot encoding with a vocabulary size of 5 is shown in Table 1.1. One-hot encoding is an unsupervised and human-interpretable method. However, to represent a word, essentially only one dimension (feature) is being used, and also, in this representation system, the distance between any two resultant word vector representations is the same. Hence, it considers words as atomic units and doesn't capture any semantic relatedness among the words. Also, it produces sparse vector representations for words and consequently has very low representational power.

Co-occurrence counts vector representation approaches are introduced to solve the word atomicity issue of one-hot encoding. The co-occurrence counts vector representations are based on the distributional hypothesis [28, 35, 80] which states that the words which occur in similar contexts will have a similar meaning. Here, the context of a word is a window of words occurring along with it in the given corpus (linguistic neighbors). This hypothesis is modeled mathematically by constructing a word-context co-occurrence matrix. A word vector consists of the word's co-occurrence counts with all the words from the vocabulary as its elements. In the co-occurrence count vector representation system, words that occur in similar contexts get similar vector representations and the distance between them will be smaller compared to the other dissimilar words. In this representation system, compared to one hot vector encoding, the features are used efficiently. Nevertheless, even this system also produces large sparse vector representations, which demand high memory and computational resources to process. Also, the length of the vector representations increases as the vocabulary size increases. These drawbacks drove the researchers' attention toward low-dimensional dense vector representations.

Table 1.1: Sample one-hot encoding word representations

	airport	plane	flight	book	weather
airport	1	0	0	0	0
plane	0	1	0	0	0
flight	0	0	1	0	0
book	0	0	0	1	0
weather	0	0	0	0	1

1.1.2.1 Word embeddings

In recent years, with the introduction of the neural word embeddings concept, word representation models are undergoing a revolution. The word embedding models produce distributed, fixed-length, real-valued, low-dimensional, and dense-featured word representations learned under an unsupervised

Table 1.2: Sample word-context co-occurrence count vectors

	airport	plane	flight	book	weather
airport	0	5	4	0	3
plane	5	0	3	1	2
flight	4	3	0	0	2
book	0	1	0	0	0
weather	3	2	2	0	0

setting. There are many different architectures such as Word2vec [63, 64], Glove [70], and contextualized models [71, 25] which arrive at word embedding space in different ways; however, the underlying theories and assumptions are the same. The distributional hypothesis, similar to the co-occurrence count vector representations, is the fundamental basis of the word embedding models. So, even word embedding models also derive similarities between words from word-context co-occurrence statistics; however, the words are represented in low dimensional dense vector space. In the word embedding space, the words which are related (semantically have similar meanings) are positioned closer compared to the unrelated words. In actuality, the word embeddings are the internal weight parameters of the neural language models; consequently, the word embeddings have uninterpretable features. Being neural models, they can be trained on large amounts of text data in an online fashion. In this process, they understand the language by encoding a lot of semantic and syntactic information into the weights, which are in turn used as word representations. Due to this rich information encoded, they have high representational power.

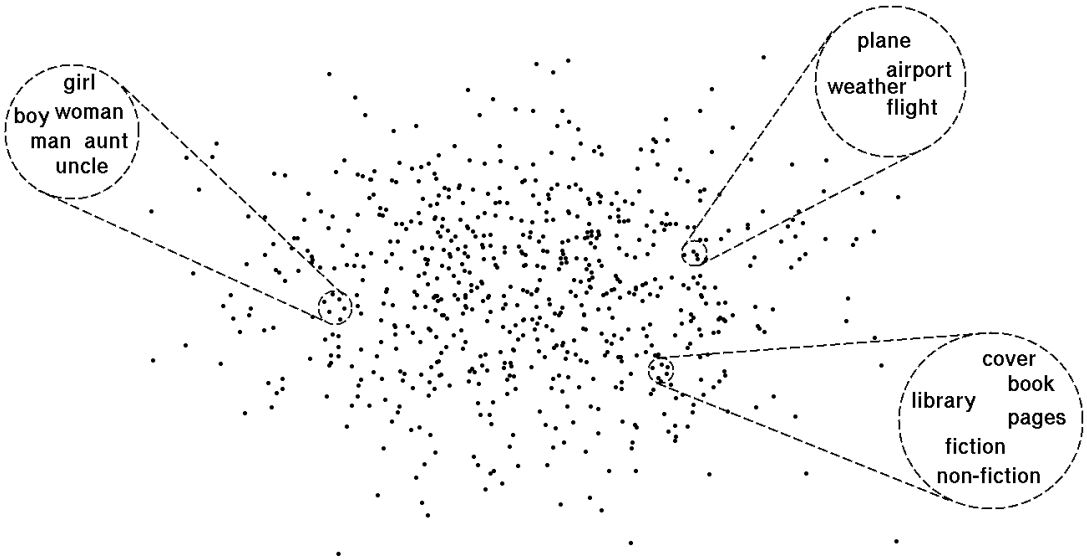

Figure 1.2: Sample word embedding space

1.1.3 Word sequence representation models

Text mining and information retrieval tasks such as information search, clustering, classification, summarization, topic detection, and sentiment analysis are performed at a larger text unit level than at the word level. So, the representation of word sequences such as sentences and documents plays an important role in many real-world applications [58]. Traditionally, word sequences are represented using Bag of Words (BoW) model. In this model, the larger text unit is modeled as a bag or a multiset of words occurring in it. This model can be understood as a simple extension of the word representation model one-hot encoding to word sequences. BoW simply adds up the one-hot vectors of words occurring in the given text. Table 1.3 shows the BoW representations for the text "the cat sat on the mat." BoW model has interpretable words as features of representation, and the associative strength between the given document/sentence and a word (feature) is defined by the word frequency. However, as an extension of one-hot encoding, BoW also suffers from word atomicity issues, sparsity in vector representations, and consequently, low representational power.

Table 1.3: Sample BoW representation

	the	cat	sat	on	mat
the	1	0	0	0	0
cat	0	1	0	0	0
sat	0	0	1	0	0
on	0	0	0	1	0
mat	0	0	0	0	1
the cat sat on the mat	2	1	1	1	1

Solving the issues of the BoW model, a generative probabilistic model named latent dirichlet allocation (LDA) is introduced [15]. The basic intuition behind LDA is that documents are a mixture of topics and topics are a mixture of words. Through a generative process, LDA defines topics as a probability distribution over words from the vocabulary. Also, models the text documents as a probability distribution over a fixed number of topics. Table 1.4 shows the sample topic modeling of documents by LDA. As BoW employs words as features, LDA employs latent topics as features, thereby representing documents as compact and dense probability vectors.

With the introduction of the successful concept of distributed representations or embeddings in the field of word representations, now many works are focusing on extending the embeddings concept to the word sequence representations. The works on word sequence embedding models can be broadly classified into two types: complex deep learning models and simple compositional models. The deep learning models employ convolutional, recursive, and recurrent neural networks like architectures to learn the word sequence embeddings in a supervised manner from the scratch [45, 32]. In contrast, the compositional models use simple vector operations like averaging on word embeddings to construct the word sequence embeddings [9, 27] as shown in Fig. 1.3. Notably, many works have shown that the simple vector averaging model, particularly in a transfer learning setting, where the word sequence

Table 1.4: Sample topic modeling of documents by LDA

	Word 1	Word 2	Word 3	Word 4	...
Topic 1	0.2	0.03	0.01	0.1	...
Topic 2	0.01	0.16	0.3	0.11	...
Topic 3	0.01	0.12	0.2	0.31	...

	Topic 1	Topic 2	Topic 3
Document 1	0.7	0.2	0.1
Document 2	0.1	0.5	0.4

embeddings are formed by composing the pre-trained word embeddings, outperforms the complex supervised deep learning models [32]. The success of this compositional model has been attributed to the geometry of word embeddings learned on large volumes of data in an unsupervised manner, as they can encode substantial syntactic and semantic information.

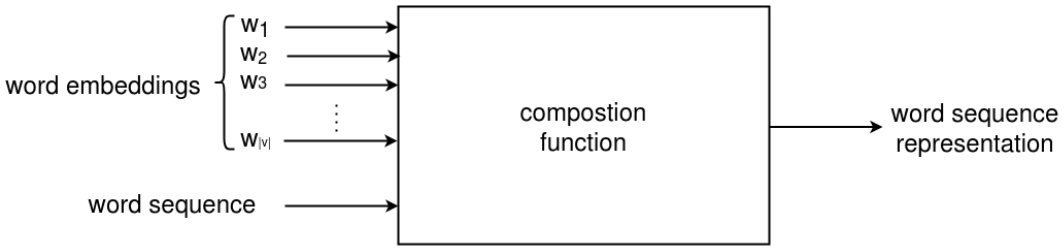

Figure 1.3: Composition function

1.2 Observations and research opportunity

From the literature survey, it is observed that the traditional models for word sequence representation provide human interpretable features such as words or topics. Also, they provide support for an unsupervised environment; however, due to their simplicity, they suffer from low representational power. Complex deep learning models are scalable to large datasets and thus have good expressive power, but they are known for uninterpretablity, and most of them are developed in a supervised setting. In a transfer learning setting, simple compositional models like vector averaging are achieving success as they are using information rich pre-trained word embeddings to construct word sequence embeddings.

To get the representation of a given text, vector averaging simply takes the component-wise mean of embeddings of words occurring in the given text resulting in a vector in the same word embeddings space. i.e, vector averaging essentially reduces the given text as one more word in the embedding space. This may be a reasonable choice for shorter texts such as phrases and sentences. However, for longer texts such as documents, this oversimplifies all topics/themes which are being discussed in it. Unlike traditional word sequence representation methods, vector averaging doesn't produce human

interpretable representations. Moreover, it gives equal importance to all the words and doesn't exploit the discriminative properties of the words.

The traditional models exploit the frequency information to build the text unit representations. In the context of word embeddings, the rich information about the meanings and semantic similarities of words are encoded in terms of distances between the words in the embedding space. There is an opportunity to utilize the semantic information encoded in the word embeddings to develop text representation models with higher representational power. In this thesis, we propose improved transfer learning based word sequence representation models over averaging in terms of one or more aspects of high representational power, interpretability, and unsupervised learnability.

1.3 Overview of the proposed models

In this thesis, we first investigate an alternative document representation framework to vector averaging with interpretable features using pre-trained word embeddings. Secondly, we investigate a word weighting scheme to improve vector averaging based on the geometry of word embeddings in the context of shorter texts such as sentences.

1.3.1 Overview of the proposed document representation framework

From the earlier sections, it is understood that the most popular composition function averaging oversimplifies all the themes/topics in the longer texts (documents) to a simple word in the word embedding space. Also, the resulting vector inherits the uninterpretable features. So, the research issue is to investigate an improved text representation framework that captures multiple aspects of the document with interpretable features under an unsupervised setting. From the traditional models, it can be observed that words/topics are employed as features, and the associative strength between the words and documents is derived from frequency statistics. In the case of word embeddings, meanings of the words (the language understanding) are encoded into the distances among them.

From the above observations, we proposed a framework where the document is represented in a vector space other than word embedding space where the dimensions/features correspond to a fixed number of potential feature words with relatively high discriminating power. In this framework, the associative strength between the document and the feature word is derived from the distance statistics from the word embedding space. As part of the proposed framework, we have proposed two criteria for the selection of discriminative feature words based on the frequency distribution and spatial distribution words. Also, we have proposed a distance function to measure the distance between the feature word and the document. Experimental results on multiple datasets show that the proposed model consistently performs better at document classification over the baseline methods. The proposed approach is simple and represents the document with interpretable word features. Overall, the proposed model provides an alternative framework to represent the larger text units with word embeddings and provides the scope to develop new approaches to improve the performance of document representation and its applications.

1.3.2 Overview of the proposed word weighting scheme

As discussed in the earlier sections, vector averaging is a simple yet effective composition function for shorter texts such as phrases and sentences. Even though the simple vector averaging model is proven to be effective, it has the drawback of attributing equal importance to all the words in the given sentence. Whereas, in reality, different words possess different degrees of information content. So, assigning weights to the words based on their discriminative power helps in building better sentence representations. Recent literature introduced word weighting schemes based on the word frequency distribution into the simple averaging model. The frequency-based weighted averaging models augmented with the denoising steps are shown to outperform many complex deep learning models. However, these frequency-based weighting schemes derive the word weights solely based on their raw counts and ignore the diversity of contexts in which these words occur.

As a part of the above-discussed framework, we proposed a feature words selection approach based on the spatial distribution (geometry) of words in the word embedding space. The approach extracts the discriminative words based on criteria of contextual diversity of words. From the insights gained from the feature selection approach, there is an opportunity to derive a weighting technique based on the spatial distribution of the words. In this work, we propose an alternative weighting scheme to frequency based weighting that captures the contextual diversity in the word embedding space. The proposed weighted average model is simple, unsupervised, and non-parametric. Experimental results on semantic textual similarity tasks show that the proposed weighting method outperforms all the baseline models with significant margins and performs competitively to the current frequency-based state-of-the-art weighting approach. Furthermore, as the frequency distribution-based approaches and the proposed word embeddings geometry-based weighting approach capture two different properties of the words, we define hybrid weighting schemes to combine both varieties. We also empirically prove that the hybrid weighting methods perform consistently better than the corresponding individual weighting schemes.

1.4 Contributions of the thesis

The key contributions of this work are

(I) **Text representation framework.** Given the dataset of documents and pre-trained word embeddings, we present a semantic distance-based document representation framework. The framework is simple and captures multiple aspects of the documents. Furthermore, the proposed framework operates in an unsupervised learning environment and its features are interpretable words.

(II) **Feature selection approaches.** We have analyzed the frequency and spatial distributional properties of the words with respect to their discriminative power. Utilizing the insights gained, we have proposed two criteria for the selection of potential feature words based on the contextual frequency and contextual diversity of words.

(III) **Word-document distance measure.** As a part of the proposed text representation framework, we have discussed multiple point-set distance measures from literature and discussed their adaptation to word embedding space. We proposed a distance measure to find the distance between a feature word and the given document.

(IV) **Word weighting schemes.** We propose a novel word weighting approach based on the geometry of word embeddings that captures the contextual diversity properties of words in spatial terms and provide a theoretical justification for it. Moreover, we have proposed hybrid weighting schemes by leveraging both the frequency distribution (contextual frequency) and spatial distribution (contextual diversity) properties of the words.

(V) **Sentence embedding method.** By utilizing the proposed weighting schemes, we have also presented a sentence embedding method based on the weighted averaging composition function. The proposed sentence embedding method also includes a post-processing denoising step to improve the final word sequence representations. The proposed sentence embedding method is also simple, completely unsupervised, and non-parametric.

(VI) **Experimental Results.** We have conducted extensive experiments on multiple datasets to demonstrate the utility of the proposed word sequence embedding models. We have compared the semantic distance based text representation framework against many baselines at the document classification task. The proposed framework consistently outperformed the traditional and deep learning based methods. We have compared the proposed word weighting based sentence embedding method against many other sentence embedding methods. Experimental results show that the proposed sentence embedding method outperforms all the baseline models with significant margins and performs competitively with the current state-of-the-art word sequence embedding models.

1.5 Organisation of the thesis

The rest of the thesis is organized as follows.

- **Chapter 2: Related Work.** In this chapter, initially, we present the related work about the word embedding models. Subsequently, we discuss the composition functions that employ algebraic operations and deep learning models, including those used in both supervised and unsupervised regimes. Lastly, we highlight the distinguishing features of the proposed models in comparison to other approaches in the literature.

- **Chapter 3: Semantic Distance Based Document Representation Framework.** This chapter introduces the proposed framework for text representation. It includes a discussion of the feature word selection methods and the word-document distance measures. Finally, the experimental results comparing the proposed framework with the baseline methods are presented.

- **Chapter 4: Word Weighting Scheme Based on the Geometry of Word Embeddings.** In this chapter, we present the proposed weighting scheme. The chapter also includes the theoretical analysis of the spatial distribution of words and hybrid weighting techniques. Additionally, we present the sentence embedding method using the proposed weighting schemes and provide experimental results to demonstrate its performance.
- **Chapter 5: Summary, Conclusions, and Future work.** In this chapter, we present the summary, conclusion, and directions for the future scope to work.

Chapter 2

Related Work

In this section, we discuss the related work regarding the word embedding models. We also discuss the related work about both supervised and unsupervised composition functions to represent the larger text units such as sentences and documents using word embeddings.

2.1 Word embeddings

Since the beginning of the representation of words as numerical vectors, the distributional hypothesis has been the fundamental principle, which states that words that occur in similar contexts have similar meanings. The early models of word representations are count-based models. The basic idea is to construct a term co-occurrence matrix and use matrix factorization to reduce the dimensionality of the matrix [79].

In 2003, Bengio et al. for the first time introduced the idea of *learning* the distributed representations for words in [13] which are now popularized as word embeddings. In this work, Bengio et al. proposed to assign a real-valued vector to each word and express the joint probability function of word sequences in terms of word vectors. The word vectors are learned along with the parameters of the probability function. A multi-layer neural network is trained to predict the next word given the previous ones. The word vectors and parameters are updated to maximize the log-likelihood of the training data. Even though the model is built to perform the language modeling task, it is considered to be *fake task*, and the actual interest is in the learned word vectors. As the model learns to perform well on language modeling tasks it encodes the relationships of the words in their vectors such that the similar words get similar feature vectors.

Later, Collobert et al. [21, 22] proposed a unified architecture for natural language processing. They proposed a general convolutional network architecture to perform multiple NLP tasks including parts of speech tagging, chunking, named entity recognition, language modeling, and semantic role labeling. All of these tasks are integrated into a single system which is trained jointly. The first layer in the architecture corresponds to word embeddings and the later layers use these word embeddings for multitask learning. Through this work, the authors established that word embeddings trained on vast amounts of text corpora, are a highly effective tool when used in downstream tasks.

2.1.1 Word2vec

As discussed above the concept of learning the word representations has a long history. However, above discussed models are computationally expensive due to the hidden layers in their architectures. It was Mikolov et al., who really brought word embeddings to the fore through the creation of a computationally efficient model named Word2Vec [63, 64] model in 2013.

There are two main variants of Word2Vec, namely Continuous Bag of Words (CBoW) and Skip-Gram. Both algorithms work by learning the relationships between words in a large text corpus. However, they differ in the way they predict the context words given a target word, and the objective function they optimize. Word2Vec is a shallow neural network and its architecture consists of an input layer, one hidden layer, and an output layer. The functionalities of these layers vary depending on the variant of Word2Vec being used (CBoW or Skip-Gram).

The main idea behind CBoW is to predict the target word given the context words. The input layer takes in the context of words where each word is represented as a one-hot vector. The hidden layer is a dense, linear, and fully-connected layer that learns the representations of the words. The output layer predicts the target word for the given context words. The softmax activation function is used in the output layer, which converts the raw scores into probabilities that sum to 1. The model's objective is to minimize the error between the predicted target word and the actual target word.

On the other hand, the Skip-Gram algorithm predicts the surrounding words (context) given a target word. The input layer takes the one hot vector of a word as input. The hidden layer, like in CBoW, is a linear layer. The output layer predicts the context words for the given input word. The objective of the model is to maximize the probability of predicting the correct context words given a target word while updating the word feature vectors in the hidden layer. The softmax activation function employed in the output layer is computationally expensive, especially when working with a large corpus of text. So, the Skip-Gram algorithm uses optimization techniques like negative sampling and hierarchical softmax to speed up the training process.

2.1.2 GloVe

GloVe (Global Vectors for Word Representation) [70] is an alternative method to Word2Vec for word embeddings proposed in 2014. Given a corpus of text, the algorithm constructs a word-context co-occurrence matrix. The co-occurrence matrix provides information about the relationship between words however, this matrix is usually sparse and high dimensional. GloVe uses a neural log-bilinear regression model to factorize this matrix to yield a lower-dimensional matrix, where each row yields an embedding for each word. The Glove architecture consists of an input layer, a hidden layer, and an output layer. The input layer takes in the word indices, and the hidden layers learn the representations of the words. The objective function of the model is to minimize the error between the predicted co-occurrence probabilities and the actual co-occurrence probabilities, which are obtained from the co-occurrence matrix.

Wor2Vec is a language modelling based technique where it learns the word embeddings while predicting the surrounding words of another word. Whereas, GloVe is a matrix factorization based model. The advantage of GloVe is that, unlike Word2vec, GloVe does not rely just on local statistics (local context information of words), but incorporates global statistics (word co-occurrence) to obtain word vectors.

2.1.3 Other advancements in word emebddings

Since the introduction of Word2Vec and GloVe, the field of NLP has seen many advancements in the area of word embeddings such as [53, 54, 16, 88, 40, 37, 68, 18].

In works [53, 54], Levy et al. theoretically analyzed the Skip-Gram training method from Word2Vec model [52] and showed that word embedding models are implicitly factorizing the word-context shifted PMI (Pointwise Mutual Information) matrix. Additionally, in [54], they demonstrated that, with proper hyper-parameter tuning, conventional distributional models can achieve comparable performance to neural embedding models. They also produced their own high-performance word embeddings by reducing the dimensionality of the positive PMI matrix using truncated SVD (Singular Value Decomposition).

Bojanowski et al. presented FastText, a word embedding model based on morphemes in [16, 40]. FastText is a Word2Vec extension that generates embeddings for each character n-gram or subword, and it represents words as the sum of these subword embeddings. This method enables FastText to capture sub-word information and handle out-of-vocabulary words more efficiently.

In their work [88], Speer et al. created word embeddings by decomposing a knowledge graph called ConceptNet 5.5. Here, ConceptNet is a semantic network that contains knowledge about the world in the form of concepts and their relationships. These relationships are represented by nodes and edges that connect them, including relationships like “is-a”, “part-of”, and “related-to”. To generate the word embeddings, the authors converted the graph into a sparse symmetric term-by-term matrix, where the context of a term is determined by the other nodes it is connected to in ConceptNet. The authors then calculated the pointwise mutual information of the matrix entries and reduced the dimensionality of the resulting matrix using SVD, following the approach suggested in [53].

Despite their success in many NLP applications, word embeddings suffer from a fundamental limitation in that they are not able to encode complex data such as hierarchical data in low dimensional linear Euclidean space. To address this limitation, a new embedding technique called Poincaré embeddings has been proposed by Nickel et al. in [68] that can capture hierarchical structures in complex data. The idea is to embed data points in a hyperbolic space rather than a Euclidean space, where data points are mapped to points in hyperbolic space, such that the distance between them reflects their hierarchical relationship.

Another important problem with the context-based word embedding techniques is disregarding the sentiment information present in the text, which results in words with opposite polarity being mapped to similar (or closer) vectors and causing sentiment analysis to fail. To address this issue there are works

[73, 99, 91] that attempt to create sentiment-specific word embeddings. The work in [73] modified the word embeddings so that they are equipped with information about the sentiment. First, they construct the vectors for each word based on its polarity, relative position in the sentence, and POS tag, which are then appended to the original Word2Vec or Glove vectors. The modified embeddings are then used for sentiment analysis tasks. On the other hand, in [99, 91], the embeddings themselves are modified by identifying sentimentally similar words using sentiment lexicons and defining a new loss function that results in the new embeddings being closer to sentimentally similar words and further away from sentimentally dissimilar words, while not moving too far away from the original vector.

The above-discussed word embedding models fail to capture the polysemous nature of words. Words can take different meanings in different contexts. Despite this, these models generate a single global vector for each word. In order to alleviate this issue in [37], SenseEmbed is proposed which generates word embeddings by considering different senses or meanings that a word can have in different contexts. SensEmbed identifies different senses of words by employing a large sense annotated inventory named BabelNet and word sense disambiguation algorithms. Once the text is disambiguated, i.e., each sense of the word is uniquely identified, the regular Word2Vec model is used to generate word embeddings for each sense.

In recent times, a new seet of deep contextualized word representation models has emerged, including CoVe, ELMo, and BERT. Unlike the earlier discussed models that generate static embeddings for words, contextualized embedding models take the entire word sequence as input and generate dynamic embeddings for words based on their contextual usage. CoVe (Contextualized Vector) was introduced in [62] and is based on a bidirectional LSTM network trained on a combination of machine translation and language modeling tasks. ELMO (Embeddings from Language Models), introduced in [71], is also based on a bidirectional LSTM network but uses a different language modeling objective. ELMO generates word embeddings by combining the hidden states of the LSTM network at multiple layers. BERT (Bidirectional Encoder Representations from Transformers), introduced in [25], is based on a transformer architecture that processes input data in parallel using self-attention mechanisms. BERT is trained on a masked language modeling task and a next-sentence prediction task. BERT generates word embeddings by considering the entire context of the sentence in both directions, making it highly effective in capturing long-range dependencies and context-dependent meaning.

To use pre-trained word embeddings, for non-contextualized models such as Word2Vec and GLove, downloading the vector packages is sufficient. However, in the case of contextualized word embeddings, the entire pre-trained model is required to generate embeddings for the given words or word sequence.

In summary, researchers have been continually working to improve the capabilities of word embeddings through ongoing research as they are now widely recognized as critical elements in a variety of applications.

2.2 Word sequence embeddings

Many important applications in NLP fields rely on understanding word sequences which are more complex language units beyond words. Here, a word sequence could be any level of semantic unit, such as a phrase, a sentence, a paragraph, or even a document.

Prior to the widespread adoption of deep learning, word sequences were often encoded as one-hot vectors or Bag of Words (BoW) [10, 59], including the weighted version known as Term frequency-Inverse document frequency (Tf-Idf) vectors [87, 74]. In these methods, a vocabulary-sized vector is used to represent the word sequence, with each element corresponding to a particular word. As a result, the dimensions of these vectors can reach thousands or even millions, leading to issues with the curse of dimensionality. Moreover, these representations are sparse, which can cause computational inefficiencies.

In recent years, as the word embeddings are proven to be information rich, there has been a surge of interest in using word embedding models to represent word sequences. The goal is to represent the word sequences, akin to word embeddings, as dense lower dimensional vectors using compositional models. Compositionality is an important aspect of natural language, allowing to construct complex semantic units from the combinations of simpler semantic elements. This principle is rooted in the idea that the semantic meaning of a whole is a function of the semantic meanings of its individual parts. The semantic meanings of complex structures rely on how their semantic elements are combined. So, composition functions are applied to vector representations of words obtained from co-occurrence statistics to produce word sequence representations.

In certain cases, a composition function, or the rules for combining word representations, can be defined in an unsupervised manner using algebraic operations such as summation. Conversely, in other cases, the composition function is learned through supervised training, often by employing deep neural networks, for a specific task.

Vector averaging [50, 29, 64, 66, 38, 32, 67, 9, 27, 83] by far is the most commonly used composition function. In this method, component-wise mean across all the word embeddings corresponding to all the words in a word sequence are calculated and the resultant vector is used as the final word sequence representation. The work in [64] represented the short phrases using unweighted averaging of Word2Vec embeddings. The work in [32] aimed at creating general purpose universal sentence embeddings. The authors, compared six compositional architectures on a wide range of domain data sets. They initialized their model with pre-trained word embeddings and then fine-tuned them via supervision from the Paraphrase pairs dataset (PPDB). The experimental results proved that the basic vector averaging based model vastly outperforms complex architectures such as Long Short-Term Memory(LSTMs). Mitchell et al. examined models of semantic composition based on additive and multiplicative vector combinations in [66]. The authors concluded that while simple vector averaging has limitations such as the inability to model word order or incorporate contextual and syntactic information, this approach serves as a fairly robust baseline.

In supervised regimes, deep neural networks such as convolutional neural networks (CNNs), and recurrent neural networks (RNNs) are employed as composition models. These models learn word embeddings as a part of their architecture or use pre-trained word embeddings. Alternatively, they may initialize with pre-trained word embeddings and fine-tune them to the task at hand such as text classification.

CNN based: In [44], Kim et al. were the first to propose the use of CNN models trained on top of pre-trained word embeddings for sentence classification. The proposed model is rather a shallow neural net with one convolutional layer (using multiple widths and filters) followed by a max pooling layer over time. The final classifier uses one fully connected layer with drop-out. The publicly available Word2Vec embeddings introduced in [63] were used to initialize the weights of the word embedding layer. They experimented with both keeping the word embeddings static and fine-tuning them, and both alternatives achieved excellent results on multiple benchmarks.

The work in [42] presents a novel CNN model that is capable of modeling sentences of different lengths and generating a fixed-size vector for each input sentence. The model is designed with a deep architecture that comprises five convolutional layers. Additionally, the authors introduce a new technique called dynamic k-max pooling, which is an extension of the max pooling operator. Unlike max pooling, dynamic k-max pooling returns the k highest values in a linear sequence of values, enabling the model to detect the k most relevant features in a sentence regardless of their position and preserving their relative order. The value of the pooling parameter k is dynamically determined based on the sentence length and the layer position in the network.

In [101], a character-level CNN for text classification. This model accepts one-hot encoded characters of a fixed size as input and consists of six convolutional layers with pooling operations and three fully connected layers. Previous research has shown that CNNs typically require large datasets to achieve good performance, so the authors created multiple large datasets to train the CNN model from scratch. Because the model operates at the character level, they claimed that its architecture is not affected by the language being used. In [23], a similar concept was investigated, where they introduced an architecture called VD-CNN (Very Deep CNN) that works directly at the character level. This study suggests that deeper models produce better results in sentence classification and are capable of capturing hierarchical information that starts from individual characters and extends up to entire sentences.

RNN based: CNNs excel at identifying local patterns that remain invariant across space. In contrast, Recurrent Neural Networks (RNNs) are designed to recognize patterns over time. RNNs treat text as a sequence of words and aim to capture word relationships and text organization. There have been works [85, 89] which use RNN and LSTM models to learn from tree structures of natural language.

In [85], Socher et al. introduced the MV-RNN model, which utilizes a recursive matrix-vector approach to capture the semantic compositionality of sentences through a tree structure. Initially, a parser is employed to generate a syntactically plausible parse tree. This tree specifies the order in which the words, which are lower-level constituents, are combined to form word sequences (phrases and sentences). MV-RNN assigns both a vector and a matrix to each word. The vector captures the meaning

of that word or constituent. The matrix captures how it modifies the meaning of the other word that it combines with. Representations for longer phrases are computed from the bottom-up by recursively combining the words according to the syntactic structure of the parse tree. At each node, the left and right contexts are combined using weights that are shared for all nodes. This results in the model learning a non-linear composition or merging function with an RNN.

In [89], Tai et al. investigated a similar research direction. They proposed a tree-structured LSTM framework to capture information from the constituent or dependency parsing tree structure. Their framework included two types of tree-structured LSTMs: the Child-Sum Tree-LSTM and the N-ary Tree-LSTM. Unlike [85], the N-ary Tree-LSTM can incorporate information from multiple child units, not just two. In contrast to basic LSTMs, which only allow for strictly sequential information propagation, the Tree-LSTM uses a principled approach to consider long-distance interactions over hierarchies and has demonstrated better performance.

Hybrid architectures based: Hybrid models combining deep learning architectures such as CNN, RNN, and LSTM have been proposed for text classification in several works [47, 92, 103]. In [47, 92], Recurrent CNN (RCNN) models are employed for text classification, which applies a recurrent structure to capture contextual information. The recurrent structure captures the long-range contextual dependencies for learning word representations and max-pooling is employed to automatically select only the salient words that are crucial to the text classification task. Another hybrid model, named C-LSTM, has been introduced by Zhou et al. This model combines CNN with LSTM for text classification. In this model, the CNN layers are employed to learn phrase-level representations, which are then fed to LSTM to learn long-term dependencies.

DAN: The deep learning models perform well because they consider word order and the semantic and syntactic structure of sentences. Despite their superior performance, constructing these complex deep learning architectures is extremely expensive as they require huge amounts of data and longer training times. In [38], Iyyer et al. introduced a supervised average based model named Deep Averaging Network (DAN) which performs competitively with more complicated neural networks on a range of sentence and document-level tasks. DAN operates by taking the average vector of pre-trained word embeddings in the input word sequence and passing it through linear layers for classification on the final output layer. The model also employs a dropout-inspired regularizer. For each input sequence, some of the words are dropped before computing the averaging, which forces the network to learn the essential words using a supervised training objective. Although the DAN model is based on basic averaging and linear layers, it achieves near-state-of-the-art accuracies with just a few minutes of training time.

HAN: As the attention mechanism gained popularity in the field of natural language processing, Yang et al. [96] introduced the Hierarchical Attention Network (HAN) as a way to utilize the attention technique. The HAN architecture is designed to capture the hierarchical structure of documents, recognizing that words combine to form sentences, and sentences combine to form documents. HAN generates a representation of a document by first creating representations of its constituent sentences from the individual words, and then combining those representations to create a final document rep-

resentation. The HAN architecture includes both sentence-level and word-level attention mechanisms within an RNN-based model.

Multi-task learning based: Typically, deep neural network based methods are trained on a single task, and sometimes the corresponding datasets may not be large enough. However, a multi-task learning framework allows the model to learn multiple related tasks simultaneously. In their research [55], the authors proposed a multitask learning framework for learning word sequence representations. The basic multi-task architecture involves shared lower layers that determine common features, followed by task-specific layers that are split across the multiple related tasks. The work [55] employed an RNN-based multi-task learning architecture to learn across multiple sentence classification tasks jointly. The lower layers learn the common embeddings for words and the later layers learn how to compose sentence representations for the specific task.

BERT based: By leveraging attention mechanisms and multi-task learning, pre-trained transformer-based language models like BERT [25] and its variants (e.g., ELECTRA [20], ALBERT [48], and RoBERTa [56]) have achieved state-of-the-art results on various natural language processing tasks, such as question answering, sentence classification, and sentence pair regression. As we discussed in Section 2.1.3, BERT generates embeddings for tokens in an input sequence using self-attention and produces an embedding for a special token [CLS], which represents the entire sequence of tokens. Typically, researchers obtain word sequence embeddings by averaging the token embeddings obtained from the output layer (referred to as BERT embeddings) or using the [CLS] token embedding. However, recent research by Reimers and Gurevych in [72] has shown that using BERT embeddings for sentence embeddings in an unsupervised regime yields inferior performance compared to averaging GloVe embeddings. BERT requires supervised fine-tuning to perform better on any specific task. The authors reasoned that this is because BERT produces sentence embeddings that are not compatible with cosine-similarity or Euclidean distance.

2.2.1 Unsupervised word sequence models

As it has become evident that vector averaging is a strong unsupervised baseline, many works proposed improved averaging based models for word sequence representations. The paper [67] introduces a method called All But the Top (ABT), which involves a pre-processing step for words before constructing word sequence embeddings by averaging the word embeddings. The authors found that Word2Vec and GloVe embeddings have a non-zero mean, meaning that they share a large common vector with a length of up to half the average of norms of word vectors. Additionally, the word embeddings are not isotropic, meaning that much of the energy of most word embeddings is contained in a low dimensional subspace. Since all words share the same common vector and have the same dominating directions, these factors strongly influence word representations. To address this, the authors propose to "purify" the embeddings by zero-centering the word embeddings and nullifying a few top dominating directions in the word embeddings. They also demonstrate that the processed word representations are more effec-

tive in capturing linguistic regularities and outperform the original embeddings on tasks such as word similarity, concept categorization, and word analogy.

The work in [9] introduced a parametric word frequency-based weighted averaging method for word sequence embeddings named Smooth Inverse Frequency (SIF). The underlying intuition of SIF is that the embeddings of too frequent words should be down-weighted when summed with those of less frequent ones. The weight of a word w is calculated as $a/(a + p(w))$, where a is the parameter and $p(w)$ is the frequency of the word w. In this method, first, weighted averages of the word embeddings are calculated and then projections of the average vectors on their first principal component are removed. In addition, the authors offer a theoretical explanation for the reweighting approach by presenting a generative model for constructing sentences, which is based on the random walk on discourses model proposed in [8], with some modifications. Furthermore, the authors provide empirical evidence that incorporating the SIF weighting method enhances performance by 10% to 30% in textual similarity tasks. They also demonstrate that this method surpasses more advanced supervised methods, such as RNNs and LSTMs.

The above-discussed SIF weighting scheme requires hyper-parameter tuning, so, to make it completely unsupervised, [27] proposed a modified frequency-based weighting scheme named unsupervised Smoothed Inverse Frequency (uSIF) in which parameter values are derived based on information from the dataset itself. The authors contend that the log-linear word production model based on random walks suggested in [8] is sensitive to word vector length. In [27], a modified random walk model was introduced that is immune to distortion by vector length, resulting in a modified word weighting scheme that requires no hyper-parameter tuning. The authors proposed removing the projections of the average vectors on their first few principal components in this approach. Finally, they demonstrated that the uSIF approach outperforms SIF by up to 44.4% on textual similarity tasks and is even competitive with state-of-the-art methods.

The works by Zhang et al. and Kim et al. [101, 43] presents bag of concepts approach which uses clustering of word embeddings for document representations. In [43] Kim et al. argue that BoW approach has the advantage of producing document vectors that are easily interpretable. However, the resulting vectors are high-dimensional and sparse, making conventional distance metrics ineffective at representing document proximity. Additionally, the BoW approach assumes that all words within a document are independent, despite the fact that synonym and hypernym word types typically convey similar information. Traditional dimensionality reduction techniques such as LSA sacrifice the interpretability of the BoW approach. To address these limitations, the authors propose reducing the dimensionality of BoW vectors using pre-trained word embeddings. In the proposed bag of concepts approach, all words in the vocabulary are clustered using k-means algorithm to identify concepts, which are then used as features for document representation. The strength of association between a document and a concept is calculated as the number of shared words. This approach outperforms BoW and retains model explainability, making it easier to understand the underlying logic of the text mining model.

For word sequence embeddings, Kiros et al proposed an unsupervised (or self-supervised) deep learning model named Skip-thought vectors in [45]. Skip-thoughts is an adaption of the skip-gram training method of the Word2Vec model for sentence embeddings. In this method, the basic units of operation are sentences rather than words. For a given sentence, the objective is to reconstruct its surrounding sentences. To achieve this objective, an encoder-decoder model is used. The encoder takes the word embeddings in a sentence as input and maps to a sentence embedding. The decoder takes the sentence embedding as the input and tries to generate the words for sentences surrounding the input sentence. The skip-thoughts model is trained on BookCorpus dataset, a large collection of novels of different genres. Once the training is complete, the skip-thoughts model is frozen, and use only the encoder is used to generate representations for any given word sequence.

In [52], for document representation, an unsupervised model named Doc2Vec is proposed. Doc2Vec is an extension of Word2Vec but extended to handle the entire document. In Doc2Vec, each document is treated as a unique context word and is assigned a unique vector representation. The algorithm trains the model to predict the words in a given context, with the documents themselves serving as the context. During training, the model learns the relationship between the words and the documents, and the vectors generated for each document and word capture the meaning and context of the words within the documents. There are two main approaches for training a Doc2Vec model: Distributed Bag of Words (DBoW) and Distributed Memory (DM). In the DBoW approach, the model only trains on the documents and uses the context of the document to predict the words that appear in the document. The main idea behind this approach is to treat the documents as bags of words, and the model only considers the presence of the words in the document, not the order in which they appear. In contrast, the DM approach trains on both the documents and the words and uses the context of the document as well as the previous words in the document to predict the next word. The main idea behind this approach is to treat the documents as sequences of words, and the model takes into account the order in which the words appear.

Without explicitly constructing vector representations, the works in [46, 102, 100] proposed similarity measures to find the distance between two word sequences. Kusner et al. [46] proposed a method called Word Mover's Distance (WMD) which employs word alignment to find the dissimilarity (distance) between two word sequences. WMD is a variant of the Earth Mover's Distance [78], and can be computed by solving the transportation problem. WMD views text documents as a weighted point cloud of embedded words, and calculates the minimum distance that the embedded words of one document must travel to reach the embedded words of another document. WMD is hyper-parameter free and provides a highly interpretable distance metric, as the distance between two documents can be explained in terms of the sparse distances between a few individual words.

Another semantic text similarity (distance) metric named DynaMax is introduced in [102]. In this paper, Zhelezniak et al. proposed fuzzy bag-of-words representation for text. Unlike classical BoW, the fuzzy BoW contains all the words in the vocabulary simultaneously but with different degrees of membership. Each word in the vocabulary is converted to a fuzzy set by calculating its similarity with

the rest of the words. Pre-trained word embeddings are used to find the similarity scores between two words. The obtained set contains all the words with different membership degrees ranging from 0 to 1 based on their similarity scores. Each sentence is converted to fuzzy BoW by using fuzzy union operation on all the fuzzy sets of the words occurring in that sentence. The membership degree of a word in the fuzzy union is determined as the maximum of membership degrees (max-pooling) of that word in all the fuzzy sets that are being unioned. The similarity between the two word sequences is calculated using the fuzzy Jaccard index. However, converting the words to fuzzy sets by considering the whole vocabulary is computationally expensive. So, DynaMax suggests using only the words occurring in the two sentences that are being compared as the vocabulary. DynaMax is completely unsupervised and non-parametric.

BERTScore [100], a new metric originally developed for evaluating natural language generation. BERTScore is used to compare the candidate sentences generated by the text generation model and the gold standard references. However, due to its task-agnostic nature, BERTScore has also been utilized as a word sequence similarity metric in [98]. The metric computes the similarity of two sentences as a sum of cosine similarities between their tokens' embeddings weighted with Inverse document frequency (Idf) weights. Initially, pair-wise cosine similarity between the token embeddings of the given two sentences is computed. Then, for each token in the reference sentence, the closest token is identified in the candidate sentence using pre-computed cosine similarity scores. Finally, the distance between the reference and candidate sentences is calculated as the sum of the similarity scores of the matched token pairs. Optionally, these similarity scores can be weighted with Idf values.

Despite their simplicity, WMD, Dynamax, and BERTScore measures have limitations in their usage. They can only be used as standalone text distance metrics, meaning that they cannot be directly used as inputs for most machine learning algorithms. This is because most machine learning algorithms require a fixed-length feature representation as input, and these similarity measures do not provide such representations. Nevertheless, these similarity measures can still be beneficial in specific circumstances. For instance, they can be integrated with KNN (k-nearest neighbors) or K-means algorithms to perform classification and clustering tasks.

2.3 Methodological differences of the proposed approaches

From the literature survey, it is evident that word embeddings are a well explored area and continuous efforts are being made to improve their quality. For word sequence representation, while most of the traditional models are word counts based, the recent methods are deep learning based. It is observed that simple composition functions in a transfer learning setting, where the pre-trained word embeddings are composed to generate the word sequence embeddings are shown to be very effective. However, in the current literature, only the word frequency properties are studied and the spatial distribution of word embeddings and their contextual diversity properties are not explored to develop the composition functions. In this work, with the motivation of utilizing the latent information stored as distances in the

word embedding space, we proposed a document representation framework. Also, by exploiting both frequency and spatial distributional (geometrical) properties of the words we proposed a word weighting scheme for sentence embeddings.

In the subsequent chapters, we present the above-mentioned semantic distance based document representation framework and the word weighting scheme based on the geometry of word embeddings.

Chapter 3

Semantic Distance Based Document Representation Framework

In this chapter, we present the proposed word sequence representation framework using word embeddings. After the introduction, we discuss the components of the proposed framework in Section 3.2. In Section 3.3, we present the performance evaluation, and finally, in Section 3.4, we present the summary of the chapter.

3.1 Introduction

In the previous chapters, we discussed the concept of word embeddings and their intrinsic properties (Refer to Chapter 1, Section 1.1.2 and Chapter 2, Section 2.1). Also, we have discussed that pre-trained word embeddings are information rich and they are useful in composing word sequence embeddings (Refer to Chapter 1, Section 1.1.3 and Chapter 2, Section 2.2).

In this chapter, we address the issue of improving document representations by exploiting word embeddings. The most commonly used composition function to represent a document is *vector averaging* of words in the given document. In [9], a weighted word averaging method is proposed, and in [52], the Doc2Vec model is proposed, by adopting Word2Vec for documents. The Doc2Vec model treats each document as another context word and co-learns the embeddings for both words and documents.

The models based on averaging [9, 52] represent the document in the same feature space as that of word embeddings i.e., essentially treating the document as another word in the word embedding space. So, they suffer from the natural drawbacks of averaging. Firstly, the existing approaches oversimplify the document representation by representing the document as a position vector in the same word embedding space. Secondly, two different distributions in a feature space can have the same average. As a result, two documents conversing about different topics can end up with fairly close vector representations. So, the existing models based on averaging are unable to capture the multiple aspects as well as the broad context of the document. Also, word embeddings are weights from hidden layers of neural language models, so the semantic space in which they are represented often has latent features/dimensions. The meaning encoded in each feature/dimension of an individual word embedding is unclear and uninterpretable. Consequently, document representations that are derived by compositing the word embeddings component-wise [9, 52] are also limited by the inherent uninterpretable features.

Instead of representing the document as another word in word embedding space, there is a scope to represent the document in terms of distances between the fixed number of potential interpretable feature words and the document using word embeddings. With this, it is possible to capture the multiple aspects of the document in a comprehensive manner. In other words, it is possible to form a composition function to represent the document by considering a different feature space instead of the word embedding space and use the words as the features of the document representation rather than with the latent features (dimensions) of the word embedding space.

Given a dataset of documents and pre-trained word embeddings, we present a novel framework to represent the documents as fixed length feature vectors. In the proposed framework, which we call as DIFW (Document representation with Interpretable Features using Word embeddings) framework, each dimension in the document feature vector corresponds to a potential feature word with relatively high discriminating power, and the given document is represented as its distances from the potential feature words.

The DIFW framework is simple and represents the document by capturing the multiple topics in the document in an effective manner. Also, it represents the document with interpretable word features. As a part of the DIFW framework, we propose two criteria for the selection of potential feature words based on the frequency distribution and spatial distribution of words. Also, we discussed multiple distance definitions from the literature and proposed a new distance measure to find the distance between the given word and document. We have conducted extensive experiments on multiple datasets to demonstrate the utility of the DIFW framework. Experimental results show that the proposed framework consistently performs better at document classification over the baseline methods.

In the remainder of this chapter, we present the proposed framework and explain the experimental results.

3.2 Proposed Framework

In this framework, initially, we will discuss how the documents are modeled, followed by feature selection approaches, and finally, the word-document distance measures.

3.2.1 Proposed model

Notably, the existing document representation approaches based on averaging essentially model the document as another word in the given word embedding space. Since word embeddings have latent features, the document representation which is the mean of word embeddings also inherits these latent features. As an illustrative example, consider Figure 3.1 which depicts the word embedding space with two dimensions: *Latent feature 1* and *Latent feature 2*. In this figure, each *dot* represents a word in the vocabulary and the *stars* represent the words in the document $D1$. The *square* represents the *mean* of the words in $D1$. Since the *mean* also lies in the same word embedding space, the document $D1$ is represented as another word in the word embedding space.

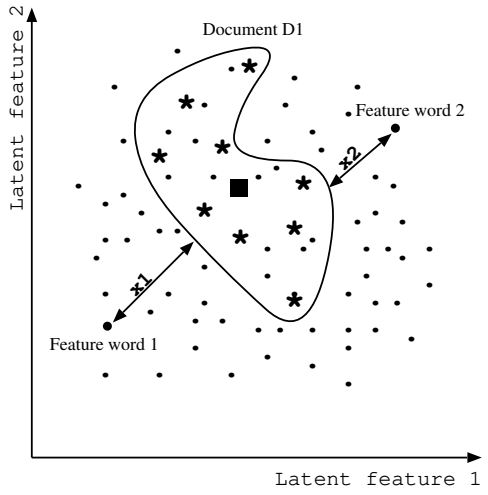

Figure 3.1: Word embedding space (WS)

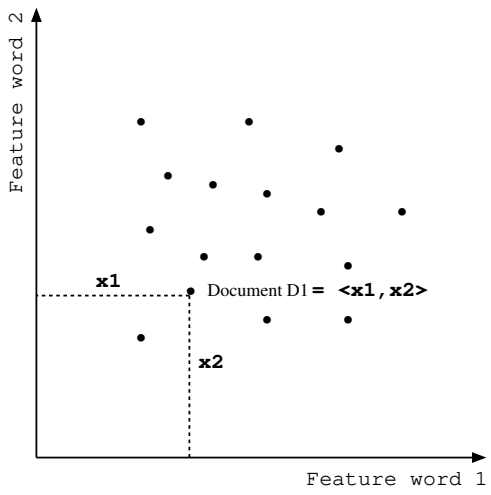

Figure 3.2: Document space (DS)

There is an opportunity to improve the document representation by representing the documents in a higher dimensional feature space with interpretable word features. Consider a feature map $\phi : WS \mapsto DS$, where WS is the Word embedding Space and DS is the Document Space, and ϕ maps the documents which are *packed* in WS as overlapping word sets into DS as position vectors. In DS, each dimension corresponds to a feature word and a given document is represented with the corresponding distances from the set of potential feature words selected from WS. As an illustrative example, consider Figure 3.2, which depicts DS with two dimensions: *Feature word 1* and *Feature word 2*. In this figure, each dot represents a document. For document $D1$ (which is a word set in WS as shown in Figure 3.1), the distance from *Feature word 1* to $D1$ is $x1$, which becomes one component, and the distance from *Feature word 2* to $D1$ is $x2$, which becomes another component. So, under the proposed DIFW model, like $D1$, a document is represented as a position vector in DS.

The document representation model under the proposed DIFW framework is as follows. Consider a set of words $\{w_1, w_2, \ldots, w_n\}$ of size n, sampled from the vocabulary of given set of documents. A given document D is modeled as a n-ary vector $< d_1, d_2, \ldots, d_n >$ s.t. $d_i \in N_i$. Here, N_i is a domain of distance values corresponding to the feature word w_i, and d_i is a distance value in domain N_i, which represents the distance between w_i and D.

In word embedding space, a word selected as a feature word acts as a representative of the concept that is conveyed by its neighboring words and the distance represents the degree of semantic dissimilarity. So under the DIFW framework, the document representations expressed in terms of distances from multiple feature words possess high representational power along with the advantage of feature interpretability.

Given the dataset of documents and pre-trained word embeddings, the two main steps in the DIFW model to represent the given document are as follows: (i) Selection of feature words from the vocab-

ulary, and (ii) Measuring the distance between feature words and the given document. We present the corresponding approaches in the following subsections.

3.2.2 Approaches to select feature words

We present two approaches for the selection of feature words from the vocabulary formed by the given dataset. First, we explain the approach based on the words frequency distribution. Next, we present the approach based on the words spatial distribution.

3.2.2.1 Words frequency distribution based approach

In natural language, words occur according to Zipf's law [104]. As a result, the frequency distribution of *words* is a long tail distribution [11]. As an example, consider a sample frequency distribution of *words* from $20Newsgroups$ dataset in Figure 3.3. High-frequency words are positioned in the *Head* part of the curve and the rare words are positioned in the *Long tail* of the curve. A rudimentary strategy to select feature words from the vocabulary is a random selection. A word drawn independently from the vocabulary at random is more likely to come from the long tail part of the distribution. So, most of the feature words will be *rare words*. The *rare words* possess greater specificity so collectively as features they can cover very few documents [87]. On the other end, *very high frequent words* posses low discriminative power and are the candidates for stop-words [81]. The words which are moderately frequent have both high discriminative power and high coverage of the documents. So, in this approach, the feature words are selected by choosing *moderately frequent words* from the vocabulary of the dataset.

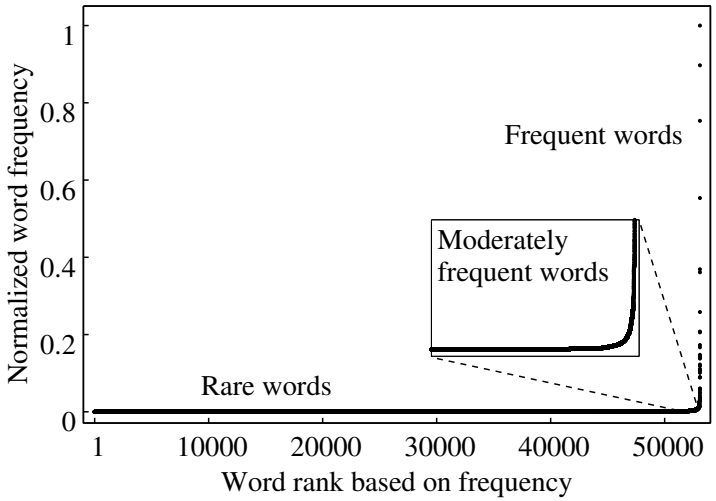

Figure 3.3: Words frequency distribution for $20Newsgroups$ dataset

3.2.2.2 Words spatial distribution based approach

The word embedding models encode the relatedness of words as the spatial distances among them in word embedding space. So, if we consider a sample of domain-related words (DRWs) from different domains in word embedding space, the words from the same domain will be positioned closer to each other, but the words belonging to different domains will be positioned farther from each other.

In this section, along with the domain-related words (DRWs), we analyze the positioning of generic words (GWs) in the word embedding space. Here, a generic word is a word that is particularly not related to any domain but commonly occurs in all the domains.

To understand the positioning of words based on their domain-relatedness, consider n DRWs $\overline{x}_1, \overline{x}_2, \overline{x}_3, ..., \overline{x}_n$ and an ideal GW $\overline{x}_{n+1}$ which co-occurs with all the DRWs uniformly. We can operationalize the distributional hypothesis over these words by defining the most general mean squared error cost function E as shown in Eq. 3.1.

$$E = \frac{2}{n(n+1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n+1} (-1)^r \|\overline{x}_i - \overline{x}_j\|^2 \tag{3.1}$$

The cost function E is defined over the pairwise distances of all the words from $\overline{x}_1$ to $\overline{x}_{n+1}$. In E, $\|\overline{x}_i - \overline{x}_j\|$ is the euclidean distance between two words $\overline{x}_i$ and $\overline{x}_j$. Here, for related words $r = 0$, and for unrelated words $r = 1$. Thus, by minimizing E, we can minimize the distance between the related words and maximize the distance between unrelated words.

Let's say we use a gradient descent optimization algorithm to minimize E. In the gradient descent algorithm, all the word vectors (parameters) will be initialized at random and then iteratively updated by the gradients which minimize E. The gradient of E at the generic word $\overline{x}_{n+1}$ can be calculated as follows.

$$\begin{aligned}
\frac{\partial E}{\partial \overline{x}_{n+1}} &= \frac{4}{n(n+1)} \sum_{i=1}^{n} (\overline{x}_i - \overline{x}_{n+1}) \\
&= \frac{4}{n(n+1)} \Big(\sum_{i=1}^{n} \overline{x}_i - \sum_{i=1}^{n} \overline{x}_{n+1} \Big) \\
&= \frac{4}{n+1} \Big(\frac{1}{n} \sum_{i=1}^{n} \overline{x}_i - \overline{x}_{n+1} \Big) \\
&= k(\mu - \overline{x}_{n+1})
\end{aligned} \tag{3.2}$$

Here, k is a constant, and μ is the mean of all DRWs. From Eq. 3.2, we can say that to minimize E, the generic word vector $\overline{x}_{n+1}$ should be updated towards the mean of the DRWs. Similarly, if we consider an ideal DRW which is unrelated to most of the words in the vocabulary, its word vector will be updated away from the mean of unrelated words.

Alternatively, in E, if we consider mean absolute error instead of mean square error, the GW will be updated towards the geometric median [30] of DRWs. Here, note that both mean and median are

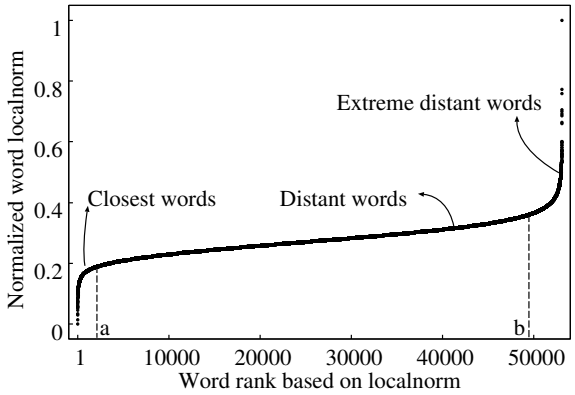

Figure 3.4: Words spatial distribution for $20Newsgroups$ dataset

the centrality measures. From this, we can say that to minimize E, GWs will be positioned near the centrality measure, which we call as the $center$ and DRWs will be positioned away from the center. For a given word, its distance from the center quantifies its degree of domain specificity.

Since the cost function E that we considered is very generic, we can expect similar phenomena for the other word embedding models such as *Word2vec* [63] and GloVe [70], which also operationalize the distributional hypothesis as explained in [76]. For simplicity, for the rest of the thesis, we consider the mean of all the words in the vocabulary as the center and the distance between a word and the center as the word's $localnorm$[1].

Based on the value of localnorm, we can divide the words into three groups: closest words, distant words, extreme distant words. Closest words are GWs and they possess low discriminative power. Distant words are DRWs and have relatively better discriminative power. Extreme distant words are also DRWs, but these words are loosely related to all the other words. So under this approach, the feature words are selected by choosing *DRWs excluding extreme distant words* from the vocabulary of the dataset.

As an example, consider the sample spatial distribution of words for $20Newsgroups$ dataset in Figure 3.4. Here, X-axis represents the word rank based on the localnorm (in ascending order), and Y-axis represents the normalized localnorm of the word. Based on the contour patterns of the localnorm curve, it is divided into three groups. Words ranked below a are closest words, words ranked between a and b are distant words in DRWs, and words ranked above b are extreme distant words in DRWs.

3.2.3 Word-document distance function

In this section, we present an approach to compute the spatial distance between a given word and a document.

[1]It is to be noted that the norm of an embedding is calculated from the origin of the embedding space, whereas the localnorm is calculated from the centroid of all the word embeddings.

The distance between any two entities in a vector space quantifies the similarity (or dissimilarity) between them. A distance measure can be defined in multiple ways but the most important property of a distance function is to have high discriminative power [26]. In the literature regarding agglomerative hierarchical cluster analysis [39] and object matching [26], there have been many distance measures defined to find the distance between any two point sets. By considering the word as a point and the document as a point set, there is an opportunity to define new distance measures in the word embedding space. We first discuss the distance measures in cluster analysis and object matching and explain the proposed distance measures.

In agglomerative hierarchical clustering, at each step, the two most similar clusters are merged into a single cluster. Here, a cluster is a set of data point vectors. The similarity (proximity) between the two clusters can be computed by using one of the following measures: single-linkage, complete-linkage, and average-linkage. In single-linkage clustering, the distance between two clusters is defined as the distance between the two closest pair of points where each point belongs to a different cluster [84, 39]. In complete-linkage clustering, the distance between two clusters is defined as the distance between the two farthest pair of points where each point belongs to a different cluster [49, 39]. In average-linkage clustering, the distance between the clusters is defined as the average of all distances between the members of a cluster to all the members of other cluster [86].

It can be observed that single-linkage clustering merges two clusters into one if a member of a cluster is highly similar to at least one member of other cluster [14]. As the clusters merging criterion in single-linkage clustering is local [60], it favors long chain-like clusters. On the other hand, complete-linkage clustering merges two clusters into one if all the members of a cluster are highly similar to all the members of other cluster [14]. As the clusters merging criterion in complete-linkage clustering is non-local [60], it favors spherical and compact clusters. The average-linkage clustering strikes the compromise between the chaining tendency of single-linkage clustering and the compacting tendency of complete-linkage clustering.

Similar to the cluster analysis, in the case of object (image) matching also an object is identified as a set of edge points. The purpose of object matching is to find the similarity between the two binary object images. The similarity between two object shapes i.e., two edge point sets is computed by finding the distance between them. Hausdorff distance is a popular measure to find the spatial distance between two point sets. It is defined as the maximum of the minimum distance between two sets of objects [36, 69]. This measure is sensitive to noise so in [26], authors proposed modified distance definitions based on the Hausdorff distance by considering the mean and median of the minimum distances between two sets of objects.

The adaptation of preceding distance measures for the word-document case is as follows. By considering word and document as a single-element set and multi-element set, respectively, the distance measures discussed in the case of cluster analysis and object matching can be adapted to word-document case majorly in four ways. Let's say w and D are the given word and document, respectively. Now, in D, consider words w_c, w_m, and w_f such that they are the closet word, middle (median) word, and

farthest word, respectively from w. Given D and w, words w_c, w_m, and w_f can be easily identified by computing the distances of all words in D from w and sorting the words in the ascending order based on the distance values.

Now, the distance between w and D can be defined by four distance measures: $\overline{ww}_c$, $\overline{ww}_m$, $\overline{ww}_f$, and d_μ which are adapted from single-linkage, modified Hausdroff distance, complete-linkage, and average-linkage respectively. The measure $\overline{ww}_c$ defines the distance between the word w and document D as the distance between word w and the closest word w_c in D. The measure $\overline{ww}_m$ defines the distance between the word w and document D as the distance between word w and the middle word w_m in D. The measure $\overline{ww}_f$ defines the distance between the word w and document D as the distance between the word w and the farthest word w_f in D. The measure d_μ defines the distance between the word w and document D as the mean of all the distances from w to every word in D.

$$\overline{ww}_c = dist(\overline{w}, \overline{w}_c) \tag{3.3}$$

$$\overline{ww}_m = dist(\overline{w}, \overline{w}_m) \tag{3.4}$$

$$\overline{ww}_f = dist(\overline{w}, \overline{w}_f) \tag{3.5}$$

$$d_\mu = \frac{1}{l}\sum_{i=1}^{l} dist(\overline{w}, \overline{w}_i) \tag{3.6}$$

Here, l is the size of the document and $dist(\overline{w}, \overline{w}_i)$ is a spatial distance measure (such as L_1-norm distance or L_2-norm distance) to find the distance between two word vectors $\overline{w}$ and $\overline{w}_i$ in word embedding space.

Even though these distance definitions are simple and parameter-free, they suffer from the following limitations. In the cases of $\overline{ww}_c$, $\overline{ww}_m$, and $\overline{ww}_f$ (Eq. 3.3, 3.4, and 3.5), much information is lost due to selecting a single word in D and finding the distance between that word and w and also these measures are sensitive to outliers and noisy words [97]. In the case of d_μ (Eq. 3.6), all distances are averaged together by ignoring the fact that the words with different semantic meanings are positioned at different distances from w. Overall, the preceding distance measures have low discriminative power as they carry very low information content.

We propose improved distance measures to overcome the limitations of word-level distances (Eq. 3.3,3.4, 3.5, 3.6). It can be observed that in word embedding space, words belonging to the same neighborhood represent the same concept or topic [43, 44]. A topic, as compared to a word, can carry higher information content and it is more immune to outliers. By extending the word-level distances, we propose new distance definitions by considering the notion of topic.

Consider T_c, T_m, and T_f, as the closest, middle, and farthest topics in the given document D, respectively. Notably, computing words belonging to T_c/T_m/T_f is simple, once we know the corresponding w_c/w_m/w_f. The topics T_c, T_m, and T_f are the collection of words neighbouring w_c, w_m, and w_f respectively. The distance between w and T_c/T_m/T_f can be calculated by simply averaging the distances from w to the words in T_c/T_m/T_f.

The proposed distance function measures the distance between word w and document D as the overall deviation of D from the w. The deviation of document D from the word w can be computed as the root mean squared distances from w to T_c, T_m, and T_f in D. The formal definition of the proposed distance function to find the distance between a feature word and document is as follows. Consider a feature word $\overline{w}$ and a document D which consists of l words. Let $D = \langle \overline{w}_1, \overline{w}_2, \overline{w}_3, ..., \overline{w}_l \rangle$ represents the ordered list of l words such that the words in D are arranged in the ascending order of their distance from $\overline{w}$, i.e., $dist(\overline{w}, \overline{w}_i) \leq dist(\overline{w}, \overline{w}_j), \forall\, i < j,\ 1 \leq i, j \leq l$. Also, let T_c, T_m, and T_f be the closet topic, median topic, and farthest topic, respectively. The sizes of these topics are $|T_c| = \frac{\alpha}{l00} \times l$, $|T_m| = \frac{\beta}{100} \times l$, and $|T_f| = \frac{\gamma}{100} \times l$, where α, β, and γ are positive real values.

Let DC (distance between $\overline{w}$ and the closest topic T_c), DM (distance between $\overline{w}$ and the middle topic T_m), and DF (distance between $\overline{w}$ and the farthest topic T_f) represent distance measures to measure the distance from $\overline{w}$ and D. Also, let DA (distance based on all topics) distance measure represents the proposed distance measure. The formulas for computing DC, DM, DF, and DA are given in Eq. 3.7, 3.8, 3.9 and 3.10, respectively. Here, DA is the proposed word-document distance function. Given $\overline{w}$ and D, we need the values of α, β and γ to determining $|T_c|$, $|T_m|$ and $|T_f|$, which are dataset dependent.

$$DC = \frac{1}{|T_c|} \sum_{i=1}^{|T_c|} dist(\overline{w}, \overline{w}_i),\ where\ 1 \leq |T_c| \leq l \tag{3.7}$$

$$DM = \frac{1}{|T_m|} \sum_{i=\frac{l}{2}-\frac{|T_m|}{2}}^{\frac{l}{2}+\frac{|T_m|}{2}} dist(\overline{w}, \overline{w}_i),\ where\ 1 \leq |T_m| \leq l \tag{3.8}$$

$$DF = \frac{1}{|T_f|} \sum_{i=l-|T_f|+1}^{l} dist(\overline{w}, \overline{w}_i),\ where\ 1 \leq |T_f| \leq l \tag{3.9}$$

$$DA = RMS\big(DC, DM, DF\big) = \sqrt{\frac{DC^2 + DM^2 + DF^2}{3}} \tag{3.10}$$

The detailed definitions and characteristics of DC, DM, DF, and DA are as follows.

DC defines the distance between the word and the document as the distance between the word and the closest topic in the document. DC is the generalized version of the distance measure in Eq. 3.3. DC is also inspired by single-linkage clustering, so it inherits the properties of single-linkage clustering. Similar to single-linkage clustering, the DC measure's criterion is local (see Figure 3.5). DC measure supposes that the word and document are related if at least one topic in the document is related to the word and it doesn't pay attention to the rest of the document (see Figure 3.5). This property of DC measure makes it suitable for the multi-label document representation where multiple topics are discussed in a single document.

DF defines the distance between the word and the document as the distance between the word and the farthest topic in the document. DF is the generalized version of the distance measure in Eq. 3.5. DF is also inspired by complete-linkage clustering, so it inherits the properties of complete-linkage clustering. Similar to complete-linkage clustering, the DF measure's criterion is non-local (see Figure

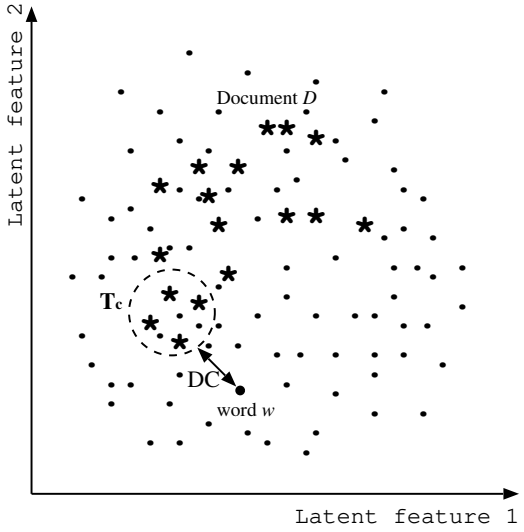

Figure 3.5: DC distance measure

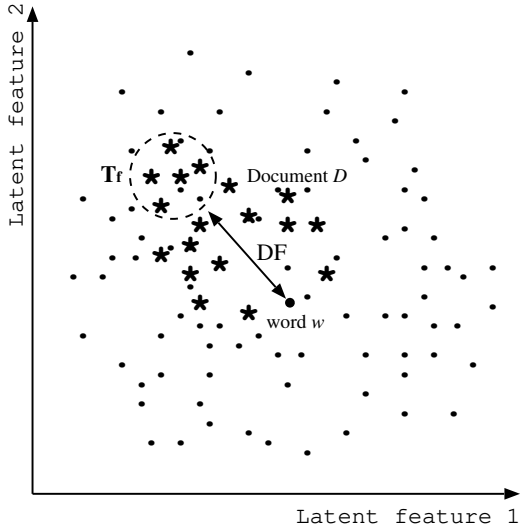

Figure 3.6: DF distance measure

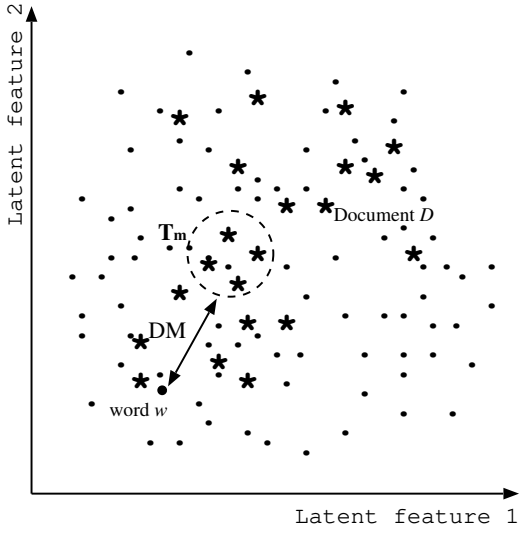

Figure 3.7: DM distance measure

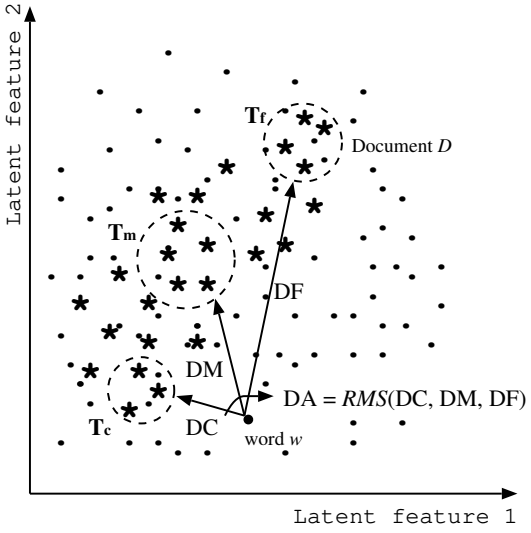

Figure 3.8: DA distance measure

3.6). DF measure supposes that the word and document are related only if all the topics in the document are related to the word (see Figure 3.6). This property of the DF measure makes it suitable for the single-label document representation where the whole document is about a single topic.

DM defines the distance between the word and the document as the distance between the word and the middle (median) topic in the document. DM is the generalized version of the distance measure in Eq. 3.4. The DM measure tries to strike a compromise between DC and DF measures. DM measure supposes that the word and document are related if the majority (at least half) of the topics are related to the word (see Figure 3.7).

The proposed DA measure defines the distance between the word and document as the root mean square (RMS) value of the distances DC, DM, and DF. It serves as an aggregator of magnitudes of DC, DM, and DF measures. Unlike DC, DM, DF, it doesn't calculate the distance based on a single aspect of the document; instead, it calculates the overall spread (deviation) of the document from the word in terms of the topical distances (see Figure 3.8). Therefore, one can notice the resemblance between the DA measure (Eq. 3.10) and the standard deviation formula.

Overall, the DA measure is simple and carries higher information content as compared to all the previously discussed distance measures. Furthermore, in contrast, to mean distance (Eq. 3.6), it captures the semantic meanings of the individual words as only semantically related word's distances are averaged together (topics) so it can be interpreted as a form of weighted averaging with word weights assigned according to their topic sizes [2]. For these reasons, the proposed distance measure has higher discriminative power over other distance measures.

3.3 Experimental Results

We have conducted experiments on 4 different text classification datasets. The $20Newsgroups$[2] dataset is a collection of news articles classified into 20 categories. We removed metadata such as headers, signatures, and quotations from the documents, which act as direct clues to the classes to make it more practical for text categorization. The $Reuters$[3] is a multi-class multi-label text classification dataset. For both $20Newsgroups$ and $Reuters$, the training set, and test set split are predefined. The BBC[4] dataset contains news stories from 2004-2005. The $AGNews$ is originally created by [101] using a large collection of titles and description fields of news articles. For computational efficiency, we randomly sampled $AGNews$ dataset. We created the training set and test set for the BBC and $AGNews$ datasets with a split ratio of 60:40.

In the experiments, for word embeddings, we used publicly available GloVe vectors[5]. These word embeddings are of 300 dimensions and are trained on a collection of Wikipedia articles. While implementing the models which require pre-trained word embeddings, the words whose pre-trained word

[2] http://qwone.com/~jason/20Newsgroups/
[3] https://archive.ics.uci.edu/ml/machine-learning-databases/reuters21578-mld/
[4] http://mlg.ucd.ie/datasets/bbc.html
[5] https://nlp.stanford.edu/projects/glove/

embeddings are not available are dropped from the vocabulary. For the models which are independent of pre-trained word embeddings, we considered the whole vocabulary.

Table 3.1 contains the details of the datasets. It contains the details such as the number of documents (N), actual vocabulary size ($|V|$), vocabulary size after dropping words whose embeddings are not available ($|V'|$), average document length (D), average document length after dropping words whose word embeddings are not available (D') and the number of classes.

Table 3.1: Datasets details

Dataset	N	$\|V\|$	$\|V'\|$	$\|D\|$	$\|D'\|$	Classes
20Newsgroups	18846	148060	53089	170	158	20
Reuters	10788	42289	27555	102	99	90
AG News	19000	31382	26514	32	31	4
BBC	2225	17746	10646	100	78	5

For all experiments, we have removed the stop-words from the documents before their feature vector generation. We used a Linear SVM classifier to perform the classification task. In all the experiments, five-fold cross-validation is employed to tune the hyper-parameters. We used computationally inexpensive L_1-norm distance measure to find the distance between two words to compute the distance for the distance measures DC, DM, DF, and DA. The default values of α, β, and γ in DA measure are 3%,70%, and 15% respectively.

We have conducted the following experiments.

- we have evaluated the performance of the proposed DIFW model using the feature words selection approach based on the words frequency distribution. We call this model as DIFW-fd model for the rest of the thesis.

- we have evaluated the performance of the proposed DIFW model using feature words selection approach based on words spatial distribution. We refer to this model as DIFW-sd model for the rest of the thesis.

- Based on experimental logs, we provided the qualitative analysis of the spatial distribution of words.

- The performance analysis of distance measures DC, DM, DF, and the proposed DA measure by varying the values of hyper-parameters α, β, and γ.

- Performance comparison of the proposed DIFW-fd and DIFW-sd models against 8 baseline methods.

- Performance analysis of hyper-parameters: the number of feature words selected from the vocabulary (n), closest topic size (α), median topic size (β), and farthest topic size (γ).

3.3.1 Performance analysis of DIFW-fd

We demonstrated the performance analysis of DIFW-fd on $20Newsgroups$ and $Reuters$ datasets. Figure 3.9 shows the quantitative analysis of DIFW-fd on $20Newsgroups$ dataset. In this experiment, the distribution curve is formed by raking the words in the vocabulary in the ascending order of their $frequency$. The distribution curve of DIFW-fd is divided into 16 equal-sized bins such that each bin contains the words approximately equal to 6.2% of the vocabulary size($|V'|$). The size of the bins is chosen such that the trends in the distribution curves are well separated by the bins. For each bin, using the feature words coupled with DA measure, the vector representations of documents in both the training set and test set are generated. Linear SVM classifier is trained on the training set vector representations and the classification task is performed on the test set. The classification accuracies over the test set for all the bins are shown in Figure 3.9. The figure contains two curves: the normalized word frequency curve corresponds to primary Y-axis and the classification accuracy step curve corresponds to secondary Y-axis. For the first curve, X-axis represents the rank of the words based on frequency, and for the second curve, X-axis represents the bin number. The bin numbers are indicated as 1 to 16 in the graph.

From Figure 3.9, it can be observed that the overall performance is increasing with the bin number. Bins 1 to 13 contains very rare words with a comparable frequency. As a result, the accuracies for these bins are relatively low and follow an arbitrary trend. From 14th bin to 16th bin, the frequency of words increases. Since the stop-words are removed, the last bin (16th) contains moderately frequent words in the dataset which have high discriminating power and high documents coverage. So the accuracies for these bins are consistently increasing and the last bin exhibits the highest performance.

The results for $Reuters$ dataset are shown in the Figure 3.10. Notably, a similar trend has been exhibited for $Reuters$ dataset. From this experiment, for the given dataset, we can conclude that the proposed approach exhibits maximum performance with moderately frequent words as feature words.

3.3.2 Performance analysis of DIFW-sd

We demonstrated the performance analysis of DIFW-sd on $20Newsgroups$ and $Reuters$ datasets. Figure 3.11 shows the quantitative analysis of DIFW-sd for $20Newsgroups$ dataset. In this experiment, the distribution curve is formed by ranking the words in the vocabulary in the ascending order of their localnorm value. Similar to the preceding experiment, the distribution curve of DIFW-sd is divided into 16 equal sized bins such that each bin contains words around 6.2% of the vocabulary size($|V'|$). For each bin, using the feature words coupled with DA measure, the vector representations of documents in both training set and test set are generated.

The classification accuracies over the test set for all the bins corresponding to $20Newsgroups$ dataset are shown in Figure 3.11. The figure contains two curves: the normalized word localnorm curve corresponds to primary Y-axis and the classification accuracy step curve corresponds to secondary Y-

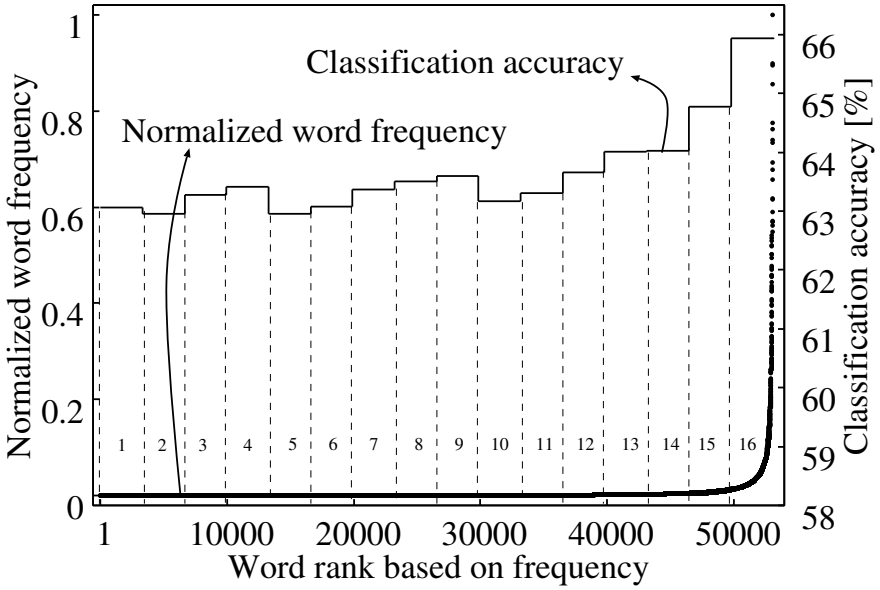

Figure 3.9: Performance of DIFW-fd on $20Newsgroups$ dataset

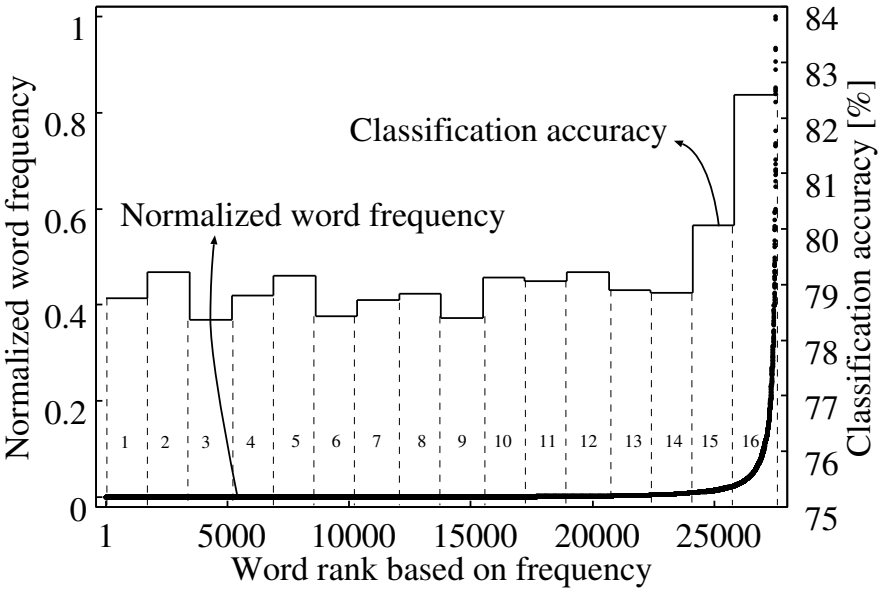

Figure 3.10: Performance of DIFW-fd on $Reuters$ dataset

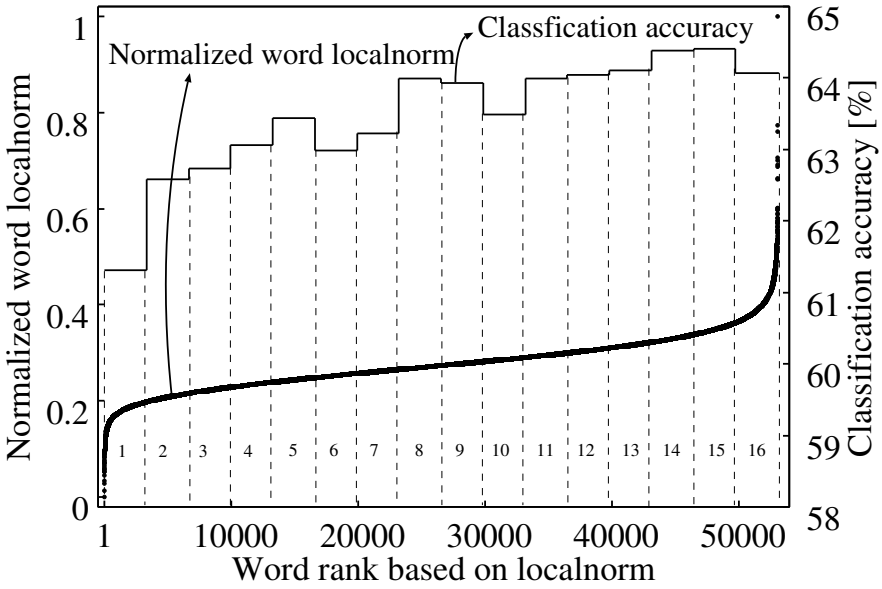

Figure 3.11: Performance of DIFW-sd on $20Newsgroups$ dataset

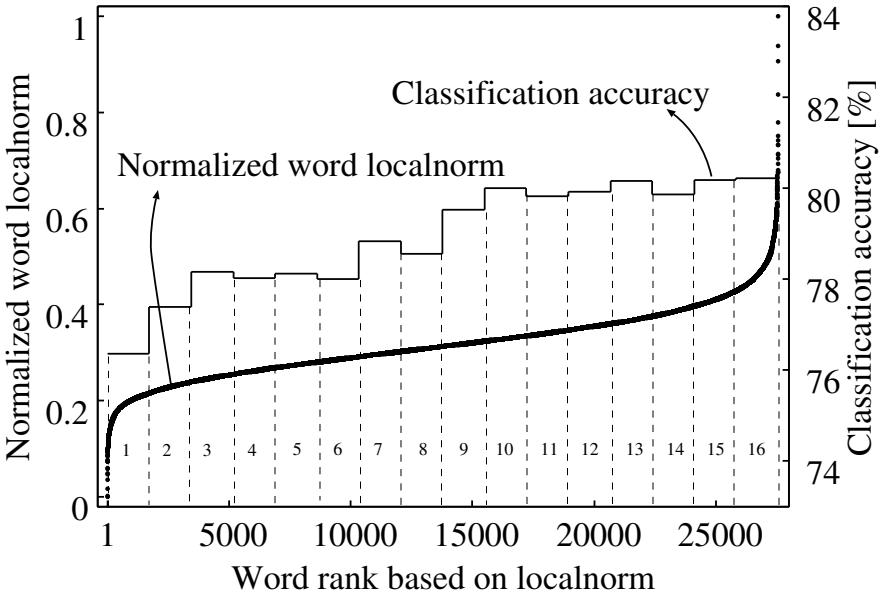

Figure 3.12: Performance of DIFW-sd on $Reuters$ dataset

axis. For the first curve, X-axis represents the rank of the words based on localnorm, and for the second curve, X-axis represents the bin number.

From the figure, it can be observed that the overall performance is increasing with the bin number. The first bin contains the GWs which have the least discriminative power so they have the lowest classification accuracy. Bins 2 to 14 contain DRWs with slightly increasing localnorm. It can be observed that the accuracies for these bins are overall exhibiting an increasing trend. The last bin, which is bin 16, also contains the DRWs. But, these are extreme distant words and loosely related to the other words. Bin 15 contains the words which are ranked before the extreme distant words and has the highest performance.

The results for $Reuters$ dataset are shown in the Figure 3.12. It can be noted that the overall performance is increasing with the localnorm. Lowest performance is exhibited by GWs in the first bin. However, notably in this case, unlike in $20Newsgroups$ dataset case, the extreme distant words in the 16th bin perform slightly better than the DRWs in the 15th bin. Normally, the extreme distant words contain noisy words, which occur out of context in the dataset. As a result, the performance of the bin containing the extreme distant words could be arbitrary. From this, we can say that it is safe to choose the words in the 15th bin which are DRWs and are ranked before the extreme distant words as feature words.

From this experiment, for a given dataset, we can conclude that the proposed approach exhibits maximum performance with farthest DRWs, which are ranked before extreme distant words as feature words.

3.3.3 Qualitative analysis of spatial distribution of words

For the qualitative analysis of spatial distribution of words, in Table 3.2, we listed 25 closest words to the center and 25 farthest words from the center along with their frequencies from $20Newsgroup$ dataset. From the lists, we can clearly observe that the words which are closest to the center are GWs and words farthest from the center are DRWs. For example, the word *lastly* is a generic word and not related to any particular domain, whereas the word *republish* is not a generic word and it is related to *literature* domain. From the lists, we can also observe that the frequencies of GWs have a wide range and they do not come under stop-words as well as rare words. So we can't filter them from the vocabulary by using the traditional frequency based trimming. However, using localnorm as a measure we can easily identify the GWs.

3.3.4 Performance analysis of distance measures

In this experiment, we analyze the performance of distance measures DC, DM, DF, and DA. The experimental analysis is presented for $20Newsgroups$ and $Reuters$ datasets using both feature words selection approaches. For each dataset, we select feature words based on one of the approaches. Next, we generate the vector representations for the documents with the distance measures DC, DM, DF, and DA. Further, we use these document representations to perform the classification task and compare the distance measures by classification accuracies. The number of feature words selected (n) for each

Table 3.2: Qualitative analysis of the spatial distribution of words

Words close to the center
lastly(8), likewise(30), interestingly(18), moreover(38), incidentally(24), furthermore(81), ironically(12), conversely(4), instance(143), indeed(270), aforementioned(9), presumably(67), importantly(19), wherein(10), evidenced(8), implying(16), consequently(11), coincidentally(3), unfortunately(236), whereupon(3), evidently(15), i.e(208), meantime(20), additionally(24), characterizing(2), paradoxically(4)
Words far away from center
republish(1), nytimes(4), daybook(1), unalienable(3), wildcards(1), distillates(1), incrimination(1), linescores(1), near-earth(4), teaspoon(2), bushel(1), advisories(1), resending(2), amortisation(1), polynomial(5), multiengine(1), affine(2), polytopes(2), projective(1), prohibitive(2), excerpted(3), autosomal(2), tensor(3), undocked(1), genus(1), backhand(4), megabits(1)

dataset may vary but it kept constant for all the distance measures comparison experiments conducted on the same dataset. The distance measures DC, DM, and DF have one hyper-parameter in their definitions, which are α, β, and γ respectively. To find the maximum performance of these distance measures, the corresponding values of α, β, and γ are varied. In the proposed distance measure DA also, we analyzed the performance by varying the three hyper-parameters α, β, and γ. However, for visualization simplicity, we reported results of DA at the default values of α, β, and γ.

Figure 3.13 shows the performance of distance measures DC, DM, and DF with feature words selected based on word frequency distribution approach on $20Newsgroups$ dataset. We selected 4500 (about 8% of $|V'|$) moderately frequent words as feature words. To analyze the performance of DC, DM, and DF, for each document, the sizes of closest topic ($|T_c|$), median topic ($|T_m|$) and farthest topic ($|T_f|$) are varied by varying the corresponding values of α, β, γ from 1% to 100% of the document size.

It can be observed that at 1% of α/β/γ, the performance of DC/DM/DF is 52%, 54%, and 57%. The performance of DC/DM/DF exhibit different trend with the increasing α/β/γ. As α increases, the performance of DC increases to a peak when α =7%, and then decreases and settles at about 61%. This indicates that the highest performance of DC could be achieved at the smaller values of α up to 10%. As β increases, the performance of DM rapidly increases untill $\beta = 70\%$ and settles at about 61%. This indicates that the highest performance of DM could be achieved with the values of β above 70%. As γ increases, the performance DF increases to peak when $\gamma = 25\%$ and then decreases and settles at about

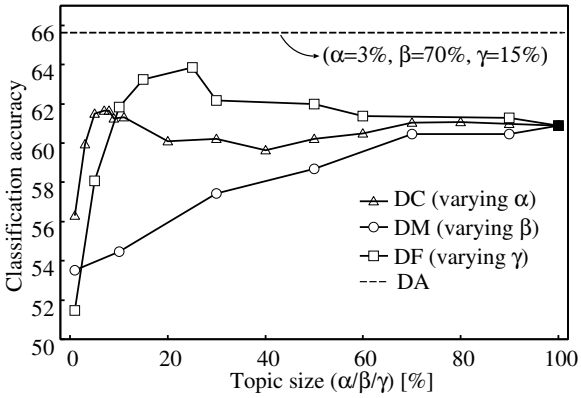

Figure 3.13: Performance of distance measures with feature words selected by word frequency distribution approach on $20Newsgroups$ dataset

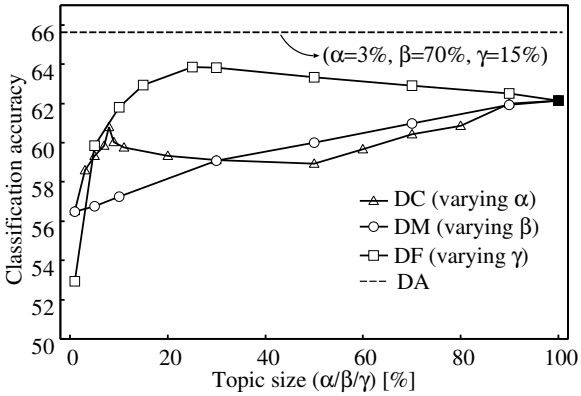

Figure 3.14: Performance of distance measures with feature words selected by word spatial distribution approach on $20Newsgroups$ dataset

61%. This indicates that the highest performance of DF could be improved with the values of γ around 25%. The performance of DC/DM/DF settles at 61% when α=β=γ=100 % because all the words of the document are covered by each approach.

Figure 3.13 also shows the performance of the distance measure DA. To analyze the performance of DA for each document, the sizes of closest topic $|T_c|$, median topic $|T_m|$ and farthest topic $|T_f|$ are fixed at α=3%, β=70%, γ=15% after conducting experiments at different values of α, β, and γ. The results show that the DA improves the performance significantly over DC, DM, and DF. It can be noted that the performance obtained by DC, DM, and DF could not achieve the maximum performance as that of DA, even though the corresponding values of α, β and γ are varied by the whole possible range.

Figure 3.14 shows the performance of distance measures with feature words selected by the words spatial distribution approach on $20Newsgroups$ dataset. We selected 4500 farthest DRWs excluding extreme distant words as feature words. Figures 3.15 and 3.16 shows the performance of distance measures with feature words selected by word frequency distribution approach and word spatial distribution

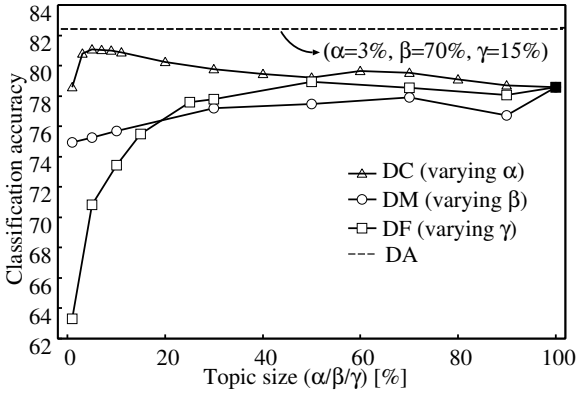

Figure 3.15: Performance of distance measures with feature words selected by word frequency distribution approach on $Reuters$ dataset

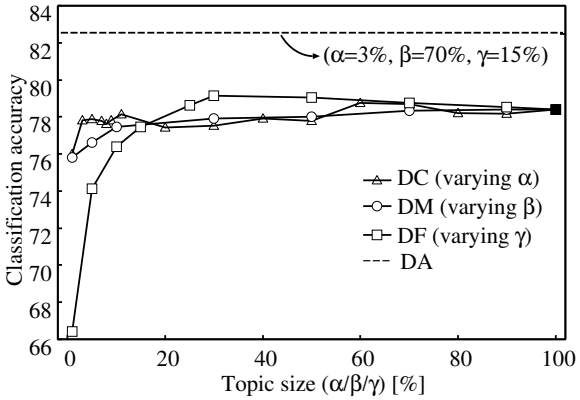

Figure 3.16: Performance of distance measures with feature words selected by word spatial distribution approach on $Reuters$ dataset

approach respectively on $Reuters$ dataset. For $Reuters$ dataset, we selected 2500 (about 9% of $|V'|$) feature words for both feature words selection approaches.

Overall, the results in Figures 3.13, 3.14, 3.15, and 3.16, demonstrate that DC performs well for small values of α (below 10%), DM performs well for very high values of β (70% to 100%), and DF performs well for medium values of γ (25% to 50%). Usually, DC and DF performs better and DM performs poorly among the three distance measures. The proposed DA measure fuses these three measures in its definition and exploits the individual powers of DC, DM, and DF. Hence, DA performs better than all of them irrespective of the feature words selection approach.

3.3.5 Performance comparison with baseline approaches

We have compared the proposed approach against 8 baseline methods on 4 different datasets. The baseline methods are Vector averaging, Min-Max concatenation [24], SIF-embeddings [9], Doc2Vec

[52], Bag of words (BOW) [35], Latent Dirichlet Allocation (LDA) [15], Bag of Concepts [43], Skip-Thought Vectors [45]. The datasets are $20Newsgroups$, $Reuters$, $AGNews$, and BBC.

We used classification accuracy as the performance metric for all the datasets. The $Reuters$ dataset is a multi-class multi-label dataset so we used F1-Score as another performance metric for this dataset.

In Vector averaging, the document vector size is equal to the number of dimensions in word embeddings which is 300. In Min-Max concatenation, the document vector size is 600 where the first 300 dimensions are from the Min vector and the rest of the 300 dimensions are from the Max vector. Min and Max's vectors are computed by performing the component-wise min and max operations on the word embeddings corresponding to the document, respectively. In SIF-embeddings, similar to Vector averaging, the document vector size is 300. The weighting scheme in SIF-embeddings has a hyper-parameter a. The values of weighting parameter a are chosen from $[10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]$ for hyper-parameter tuning.

In Doc2Vec[6], we employed the distributed memory (DM) model to generate the document vectors. To tune the document vector size, its value is varied from 100 to 500 with step size 100. The number of negative samples drawn, window size, and the minimum count of words are 5, 8, and 5 respectively. The number of epochs hyper-parameter is tuned by choosing its values from [10, 50, 100, 200]. In Bag of words[7], the number of dimensions is equal to the vocabulary size. In LDA[8], the number of topics is hyper-parameter. To tune the number of topics, its values are varied from 100 to 600 with step size 100. In Bag of Concepts, the number of concepts (or clusters) is a hyper-parameter and its values are varied from 1000 to 5000 with a step size of 500. For Skip-Thought vectors, we used publicly available encoder model[9] which is pre-trained on large external book corpora. It takes the text documents as input and produces a 4800-dimensional fixed-length vector for each sentence in a document which are then averaged to produce a document vector. In DIFW-fd and DIFW-sd, the number of feature words (n) is varied from 5% to 15% of vocabulary size, and the values of α, β, and γ in DA measure fixed at their default values.

The comparative results with baseline methods are shown in Table 3.3. The results show that the proposed DIFW-fd and DIFW-sd approaches consistently improve the performance over other approaches on all the datasets.

The performance gain of the proposed model over the Vector averaging is significant for $Reuters$ and $20Newsgroups$ datasets. The number of classes in both these datasets is relatively high and very closely related to each other. It signifies that the proposed document representation framework (Document representation model + Feature words selection approaches + Distance measure) is able to capture multiple aspects of the document in an effective manner. As a result, the proposed framework is exhibiting improved performance. Vector averaging is comparatively more insensitive to noisy words in

[6] https://radimrehurek.com/gensim/models/doc2vec.html#gensim.models.doc2vec.Doc2Vec

[7] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

[8] https://pypi.org/project/lda/

[9] https://github.com/ryankiros/skip-thoughts

Table 3.3: Experimental results against baseline approaches

	20Newsgroups	Reuters Accuracy	Reuters F1-Score	AG News	BBC
DIFW-fd	65.61	82.49	47.35	90.03	93.22
DIFW-sd	64.21	81.06	45.52	89.26	93.89
Vector averaging	62.39	77.70	37.54	88.97	92.42
Min-Max concatenation	62.50	73.91	42.09	85.53	90.50
SIF-embeddings	63.02	77.77	37.37	89.17	92.65
Doc2Vec	54.72	49.15	18.99	73.51	89.49
BOW	56.39	78.96	48.51	87.19	86.29
LDA	62.73	73.76	35.58	87.17	92.13
Bag of concepts	58.93	79.10	43.89	88.22	89.60
Skip-Thought	53.87	72.04	34.32	81.96	92.62

the vocabulary than the Min-Max concatenation method. So, Vector averaging is performing slightly better than Min-Max concatenation. SIF-embeddings assigns weights to word embeddings based on their frequency so it is performing consistently better than Vector averaging. The rest of the baselines are performing comparably to each other. The Doc2Vec model comes under the class of neural language models. Neural language models require huge amounts of data to perform effectively. The Doc2Vec model is performing poorly as it attempts to co-learn both word embeddings and document embeddings using only data at hand [51].

The following observations can be made from DIFW-fd and DIFW-sd. In DIFW-fd, a feature word is selected based on its frequency, i.e., the property of the word derived from the given dataset. In DIFW-sd, a feature word is selected based on its localnorm, i.e., the property of the word derived from the word embeddings spatial distribution, which is independent of the dataset. DIFW-sd performed better than all the baseline methods and was able to perform closely as to DIFW-fd even though the methods of selecting feature words in DIFW-sd are completely different from DIFW-fd. So, the spatial distribution based methodology provides an alternative avenue to improve the performance of document related tasks.

3.3.6 Performance analysis of hyper-parameters

In our model, there are 4 hyper-parameters: the number of feature words (n) from the feature words selection approaches, the sizes of the closest topic (α), median topic (β), farthest topic(γ) in terms of document length from the DA distance measure. We presented an empirical analysis of these parameters on $20Newsgroups$ dataset. The default values for n, α, β, and γ are 4500, 3%, 70%, and 15% respectively. To analyze a parameter, we vary that parameter and fix the rest of the parameters at their default values.

Figure 3.17 shows the cross-validation score (cv-score: average of scores from each fold) and test score for both DIFW-fd and DIFW-sd by varying the number of feature words(n) from 2000 to 5000. While increasing the n value, at each step, the document representations are generated in a computa-

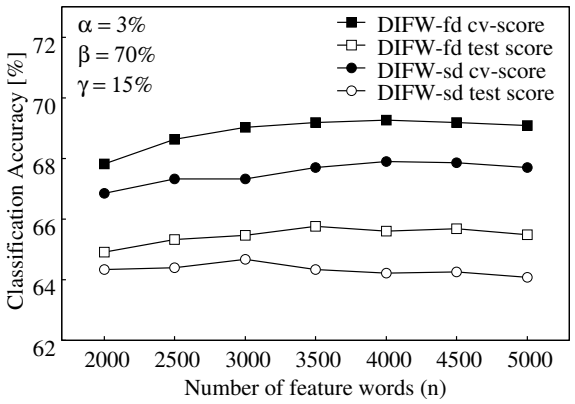

Figure 3.17: Analysis of hyper-parameter: n

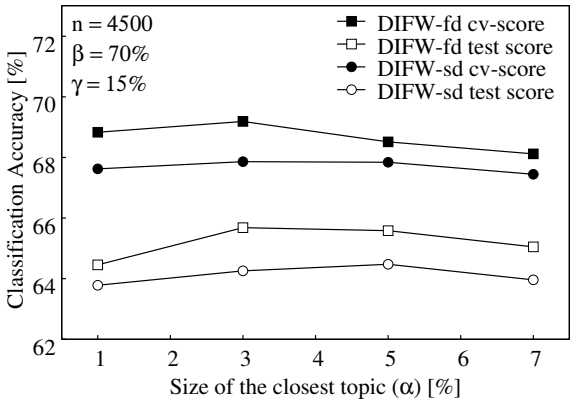

Figure 3.18: Analysis of hyper-parameter: α

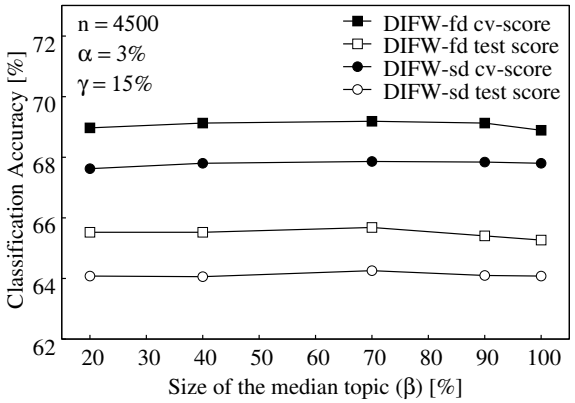

Figure 3.19: Analysis of hyper-parameter: β

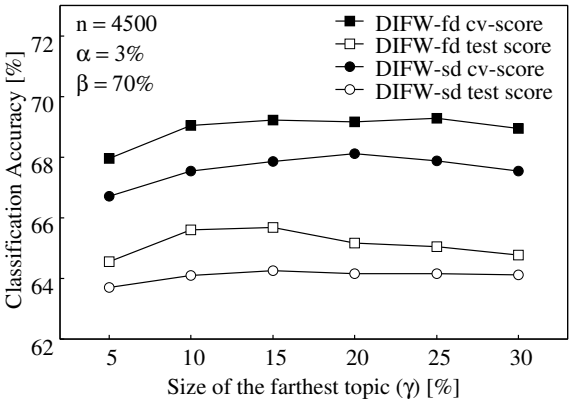

Figure 3.20: Analysis of hyper-parameter: γ

tionally effective manner by simply appending the next 500 dimensions (corresponding to the next 500 most frequent feature words) to the previous vector representations. A similar procedure is followed while generating document vectors in DIFW-sd. With the increasing n the performance also increases and reaches the peak at a point and then slightly drops afterward. For both DIFW-fd and DIFW-sd, the cv-score increases to 4000 (about 7.5% of the vocabulary size) and drops afterward. The performance follows this trend because when the n value is small the number of feature words is not sufficient enough to represent the documents effectively, as the n increases the expressive power of representations increases, after reaching the peak, as n increases the non-discriminative words are added to the features and affects the performance negatively.

Figures 3.18, 3.19, and 3.20 show the cv-score and test score for both DIFW-fd and DIFW-sd by varying α, β, and γ respectively. From Figure 3.18, we can observe that for the values of α the cv-score of DIFW-fd increases till 3% and then decreases afterward and for DIFW-sd, cv-score increases till 5% and then drops from there. From Figure 3.19, we can observe that the performance of DIFW-fd and DIFW-sd increases very slightly till β is 70% and slightly drops afterward. From Figure 3.20, we can

observe that for the values of γ the cv-score of DIFW-fd monotonically increases till 15% and then decreases afterward and for DIFW-sd, cv-score increases till 20% and then drops from there.

For other datasets also the best performances of DIFW-fd and DIFW-sd are found when α, β, and γ are approximately at 3%, 70%, and 15% respectively. Based on this observation, we used the same values as the default values in all the preceding experiments.

3.4 Summary

In this chapter, we have proposed a semantic distance based document representation framework using word embeddings. In the proposed framework, a given document is modeled as a vector of distances from multiple words in a different higher-dimensional feature space. We proposed two methods for the selection of potential feature words and present a distance function to measure the distance between the feature word and the document. We empirically evaluated these feature selection approaches and the distance measure. Experimental results on multiple data sets demonstrate that the proposed framework improves the classification accuracy significantly as compared to the baseline methods.

In the next chapter, we propose a word weighting scheme based on the geometry of word embeddings.

Chapter 4

Word Weighting Scheme Based on the Geometry of Word Embeddings

In this chapter, we present the proposed weighting scheme and the sentence embedding method based on the weighted averaging composition function. After the introduction, we discuss the weighting approach and sentence embedding algorithms in Section 4.2. In Section 4.3, we present the performance evaluation, and finally, in Section 4.4, we present the summary of the chapter.

4.1 Introduction

In the previous chapters, we learned that vector averaging is the most common composition function to generate word sequence embeddings. Also, many works have shown that the simple vector averaging model, particularly in a transfer learning setting, where the sentence embeddings are formed by composing the pre-trained word embeddings, outperforms the complex supervised deep learning models [32]. The success of this compositional model has been attributed to the geometry of word embeddings learned on large volumes of data.

Despite its simplicity and effectiveness, vector averaging has the drawback of attributing equal importance to all the words in the given sentence. Whereas, in reality, different words possess different degrees of information content. For example, *the* is a very common word and does not convey any particular concept. On the other hand, the word *database* gives an impression that the content of the text is about computers and information systems. So, assigning weights to the words based on their discriminative power helps in building better sentence representations. In Chapter 3, Section 3.2.2.2, we introduced the concept of localnorm as a feature word selection criterion. In this chapter, we present more detailed work about the localnorm and propose a weighting scheme based on localnorm to produce sentence embeddings.

In the literature, there are multiple works, such as Tf-Idf [75], which derive word weights from their term frequency and document frequency statistics. Building upon these works, recently, Arora et al. [9] proposed a word weighting scheme in the context of word embeddings named Smooth Inverse Frequency (SIF) for sentence embeddings. The basic principle behind the SIF weighting scheme is: the more the frequency of the word is, the lesser the information content it possesses and, therefore, the lesser the weight of the word should be. This weighting scheme is weakly supervised as it requires

parameter tuning, and it depends upon external sources like Wikipedia corpus to gather frequency information to calculate the word weights. In a transfer learning setting, the frequency information may not always be available publicly, or it can be incomplete if the corpus on which word embeddings are trained is related to a niche domain or anonymized to conceal sensitive information [3]. Even if the corpus data is available, processing a huge data to get the frequency information is expensive. Moreover, the frequency distribution based weighting schemes assume that the words are completely independent; therefore, they consider only individual word frequency information. However, according to the distributional hypothesis, the words are semantically related to each other at different degrees, and they co-occur in different contexts accordingly. The word-contexts co-occurrence counts distribution reflects their contextual diversity properties [93].

In this work, we propose a novel word weighting approach based on the geometry of word embeddings that captures the contextual diversity properties of words in spatial terms and provide a theoretical justification for it. By utilizing the proposed weighting scheme along with a post-processing denoising step, we propose a sentence embedding method based on the weighted averaging composition function. The proposed sentence embedding method is simple, completely unsupervised, and non-parametric. As word frequency based weighting schemes and the proposed word contextual diversity capture two different properties of the words, we propose hybrid techniques for combining these approaches to leverage both frequency distribution and spatial distribution information. We have conducted extensive experiments on multiple Semantic Textual Similarity (STS) datasets to demonstrate the utility of our word weighting scheme. Experimental results show that the proposed spatial distribution based weighting scheme not only performs competitively with the frequency distribution based weighting schemes but also can be alloyed with them to improve the overall STS task performance.

In the remainder of this chapter, we present the proposed approach and explain the experimental results.

4.2 Proposed Approach

In this section, we first present the proposed word weighting scheme derived from the geometry of word embeddings, which we call localnorm weighting scheme, that leverages the contextual diversity information. To explain the weighting scheme, we first present the overview, then the mathematical analysis, and finally a toy example illustration. Next, we present the denoising techniques which improve the quality of embeddings. Finally, we present our localnorm sentence embedding algorithm and hybrid algorithms, which combine the proposed localnorm based weighting scheme with frequency based weighting schemes.

4.2.1 Proposed weighting scheme

The frequency based word weighting schemes derive word weight solely based on the number of times a word occurs in the corpus and ignore the diversity of contexts in which it occurs. However, in

addition to the raw frequency information, the contextual diversity of a word also holds important information about its discriminative power. In this work, we proposes a weighting scheme that derives the word weights based on the contextual diversity property in the word embedding space setting. Words that occur in many different contexts have high contextual diversity, and usually, they are general words. On the other hand, Words that occur consistently in similar contexts have low contextual diversity and tend to be domain related words. Previously we demonstrated that general words have low discriminative power and domain related words have higher discriminative power (refer to Chapter 3 Section 3.2.2.2 and Section 3.3.2). Based on this understanding, we propose a derivative theory of the distributional hypothesis [77] stating that the importance of the word should be characterized by the diversity of contexts it occurs in. The fundamental principle of the proposed weighting scheme is that the degree of bias or skewness in the diversity of a word's contexts serves as an indicator of its importance. In other words, our proposed weighting scheme assigns higher weightage to words that exhibit lower contextual diversity, while words with greater contextual diversity are given a lower weightage.

The contextual diversity properties of a word are captured by its word-context co-occurrence statistics [93]. Word embedding models, which are based on the distributional hypothesis, try to capture the meaning of the words by transforming this distributional co-occurrence information into geometric representations. Therefore, analysis of this geometry of word embeddings in the semantic space provides us with an opportunity to derive a special spatial measure that captures the contextual diversity.

Previously in Chapter 3 Section 3.2.2.2, we presented an analysis regarding the positioning of general words and domain related words in the word embedding space. From the analysis, we observed that the general words are kept closer to the centroid of all the word embeddings, and domain related words are arranged farther from the centroid. Based on this analysis, we conclude that as the value of the localnorm (which refers to the distance of a word embedding from the centroid of the word cloud) increases, the contextual diversity decreases, the skewness in contexts increases, the domain specificity increases, and consequently, the discriminative power increases. So, we define the weight of the word as its localnorm value.

A simplified mathematical analysis for the derivation of the localnorm property is presented in Chapter 3. However, it is derived only from the perspective of the general words. Below, we present a comprehensive analysis of the positioning of any word with respect to its related words and unrelated words. Also, we present a simulation of this theory using a toy dataset.

4.2.1.1 Analysis

The word embedding techniques first, transform the raw co-occurrence information into some measure of relatedness and then define an optimization function to map the relatedness information to the spatial positions in the embedding/semantic space. Though there are multiple optimization methods for arriving at a word embedding space, the underlying theories and assumptions are the same. The fundamental aim of the embedding models is to reflect the semantic similarity or relatedness between words in spatial terms, as proximity in semantic space, i.e., words that are related to each other should

be closer and which are unrelated should be farther from each other. Therefore, we define a generic optimization problem that captures these fundamental principles, and hence the results thus obtained are easily generalizable.

In this connection, let us consider a sample vocabulary (V) that consists of n words $\{w_1, w_2, ..., w_n\}$ coming from a general corpus. These n words are expected to be related to each other at different degrees of relatedness. However, in this analysis, for the sake of simplification, let us consider that the semantic relatedness relationship between them is binary, i.e., the association between any pair of words is defined as either related or unrelated, and the degree of relatedness is not considered. So, each word in the V will have two mutually exclusive sets of related and unrelated words.

Now, let us consider an optimization function that can operationalize the above discussed fundamental principle of any word embedding space. In this analysis, we consider a generic mean squared error based optimization function E as shown in Eq. 4.1. Here, υ_{w_i} denotes the embedding of a word w_i, $dist(\upsilon_{w_i}, \upsilon_{w_j})$ denotes the distance between any two words w_i and w_j, $R(w_i)$ denotes set of words related to word w_i and $R'(w_i)$ denotes set of words unrelated to word w_i. Here, note that $R(w_i) \cap R'(w_i) = \phi$. $|R(w_i)|$ is the cardinality of set of words related to w_i and $|R'(w_i)|$ is the cardinality of set of words unrelated to w_i. The error cost function E is defined over the pairwise distances of all the words from vocabulary V.

$$E = \sum_{i=1}^{n} \left(\frac{\sum_{w_j \in R(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2}{|R(w_i)|} - \frac{\sum_{w_j \in R'(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2}{|R'(w_i)|} \right) \tag{4.1}$$

Note that, the first term $\left(\frac{\sum_{w_j \in R(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2}{|R(w_i)|} \right)$ in E, represents the mean squared distances from word w_i to its related words $R(w_i)$. The second represents $\left(\frac{\sum_{w_j \in R'(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2}{|R'(w_i)|} \right)$, represents the mean squared distances from word w_i to its unrelated words $R'(w_i)$. It can be observed from Eq. 4.1 that minimization of the overall loss (cost) function E results in the minimization of the first term and maximization of the second term. This can be understood as the minimization of E results in the minimization of distances between all the related words in the vocabulary and the maximization of all unrelated words in the vocabulary. So, through this formulation, we seek to generate an embedding space where the related words are closer to each other, and the unrelated words are farther from each other.

To learn the vector representation of the words by minimizing the error function E, we follow the gradient descent optimization algorithm. In this approach, the embeddings of all the words in V are randomly initialized and then updated in an iterative manner. While updating the embedding of any word w_i, the embeddings of all the words in the related and unrelated sets to be constant. Therefore, while updating the embedding for word w_i, the error function E will be reduced to those terms which depend only on the embedding of word w_i. Such an error is represented as follows:

$$E(w_i|R(w_i), R'(w_i)) \;=\; \frac{\sum_{w_j \in R(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2}{|R(w_i)|} - \frac{\sum_{w_j \in R'(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2}{|R'(w_i)|} \tag{4.2}$$

Here, $E(w_i|R(w_i), R'(w_i))$ denotes the error corresponding to the embedding of word w_i given $R(w_i)$ and $R'(w_i)$.

Also, the first term in Eq. 4.2 can be expanded as below.

$$\sum_{w_j \in R(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2 \;=\; \sum_{w_j \in R(w_i)} \|\upsilon_{w_i} - \upsilon_{w_j}\|_2^2$$

$$= \sum_{w_j \in R(w_i)} \left(\upsilon_{w_i} \cdot \upsilon_{w_i} - 2\,\upsilon_{w_i} \cdot \upsilon_{w_j} + \upsilon_{w_j} \cdot \upsilon_{w_j}\right)$$

$$= |R(w_i)|\;(\upsilon_{w_i} \cdot \upsilon_{w_i}) - 2|R(w_i)| \left(\frac{1}{|R(w_i)|} \sum_{w_j \in R(w_i)} \upsilon_{w_j}\right) \cdot \upsilon_{w_i} + \sum_{w_j \in R(w_i)} \upsilon_{w_j} \cdot \upsilon_{w_j}$$

$$= |R(w_i)|\;(\upsilon_{w_i} \cdot \upsilon_{w_i}) - 2|R(w_i)|\mu \cdot \upsilon_{w_i} + \sum_{w_j \in R(w_i)} \upsilon_{w_j} \cdot \upsilon_{w_j}$$

$$= |R(w_i)|\|\upsilon_{w_i} - \mu\|_2^2 + \sum_{w_j \in R(w_i)} \upsilon_{w_j} \cdot \upsilon_{w_j} - |R(w_i)|\|\mu\|^2 \tag{4.3}$$

Here, $\mu = \frac{1}{|R(w_i)|} \sum_{w_j \in R(w_i)} \upsilon_{w_j}$, and it is interpreted as mean of all related words to w_i.

Similarly, we can show that:

$$\sum_{w_j \in R'(w_i)} dist(\upsilon_{w_i}, \upsilon_{w_j})^2 \;=$$

$$|R'(w_i)|\|\upsilon_{w_i} - \mu'\|_2^2 + \sum_{w_j \in R'(w_i)} \upsilon_{w_j} \cdot \upsilon_{w_j} - |R'(w_i)|\|\mu'\|^2 \tag{4.4}$$

Where, $\mu' = \frac{1}{|R'(w_i)|} \sum_{w_j \in R'(w_i)} \upsilon_{w_j}$, and it is interpreted as the mean of all unrelated words to w_i.

By substituting Eq. 4.3 and Eq. 4.4 in Eq. 4.2, we get:

$$E(w_i|R(w_i), R'(w_i)) \;=\; \frac{|R(w_i)|\|\upsilon_{w_i} - \mu\|_2^2 + \sum_{w_j \in R(w_i)} \upsilon_{w_j} \cdot \upsilon_{w_j} - |R(w_i)|\|\mu\|^2}{|R(w_i)|}$$
$$- \;\frac{|R'(w_i)|\|\upsilon_{w_i} - \mu'\|_2^2 + \sum_{w_j \in R'(w_i)} \upsilon_{w_j} \cdot \upsilon_{w_j} - |R'(w_i)|\|\mu'\|^2}{|R'(w_i)|} \tag{4.5}$$

To minimize the error function E, we will learn the word embedding υ_{w_i} using gradient based update

rule. The gradient for the error function E with respect to υ_{w_i} can be calculated using Eq. 4.5 as:

$$\frac{\partial E(w_i|R(w_i), R'(w_i))}{\partial \upsilon_{w_i}} = 2(\upsilon_{w_i} - \mu) - 2(\upsilon_{w_i} - \mu') = -2(\mu - \mu')$$

Therefore, update the rule for the word embedding υ_{w_i} is:

$$\begin{aligned} \upsilon_{w_i}^{new} &= \upsilon_{w_i}^{old} - \eta \frac{\partial E(w_i|R(w_i), R'(w_i))}{\partial \upsilon_{w_i}} \\ &= \upsilon_{w_i}^{old} + \eta(\mu - \mu') \end{aligned} \tag{4.6}$$

From Eq. 4.6, we can observe that with each iteration, a fraction of the mean of w_i's related words embeddings (μ) is added to the word embedding υ_{w_i} and a fraction of the mean of w_i's unrelated words embeddings (μ') is removed from the word embedding υ_{w_i}. Therefore, with each iteration, the word embedding moves closer to the mean of its related word embeddings and moves away from the mean of its unrelated word embeddings. A word with high contextual diversity is a generic word as it occurs almost uniformly in many different contexts. So, it is related to nearly all the words in the vocabulary, and it is positioned closer to the mean (centroid) of all word embeddings in the embedding space. In contrast, a word with low contextual diversity is a domain specific word as it occurs with few specific contexts. So, it is unrelated to most of the words in the vocabulary, and it is positioned far away from the centroid of all word embeddings. So, it can be observed that as the skewness in the contextual diversity of a word increases, the distance of its embedding from the centroid increases.

From the above observations, we can say that a generic word will have a smaller localnorm, and a domain specific word will have a larger localnorm. So, localnorm of a word can be considered as a measure of its discriminative power. Moreover, since the localnorm based weighting scheme is derived using a generic optimization function that considers the fundamental principles of the distributional hypothesis it generalizes well over other word embedding models such as Word2Vec and GloVe.

4.2.1.2 Toy example

To illustrate the above theory, we consider a toy dataset from [12] which contains 4 sentences: 1. *what is the time*, 2. *what is the day*, 3. *what time is the meeting*, 4. *cancel the meeting*. The vocabulary formed by these 4 sentences contains 7 words. We consider a pair of words as related if they co-occur in at least one sentence and a pair words of words unrelated if they are never used in the same sentence. Table 4.1 shows the related and unrelated sets corresponding to each word in the vocabulary.

From the table, we can observe that *the* is the most common word co-occurring (related) with all the other words. The words *what*, *is*, and *meeting* are the next most common words, which are related to all the words except one. *what* and *is* always co-occur; therefore, their related and unrelated sets are the same. *time*, *day*, and *cancel* are related to 4, 3, and 2 words, respectively. *cancel* and *day* are the most discriminative words which can identify a sentence uniquely.

Table 4.1: Related and unrelated sets of each word in the vocabulary

Word	R: Related Words	R': Unrelated Words
the	what, is, meeting, time, day, cancel	ϕ
what	the, is, meeting, time, day	cancel
is	the, what, meeting, time, day	cancel
meeting	the, what, is, time, cancel	day
time	the, what, is, meeting	day, cancel
day	the, what, is	meeting, time, cancel
cancel	the, meeting	what, is, time, day

We train a simple model on these sentences that optimizes the error function shown in Eq. 4.1. First, for all the words in the vocabulary, two-dimensional embeddings are initialized randomly and then stochastic gradient descent iterative optimization algorithm is used to learn the word embeddings with the update Eq. 4.6. As discussed in the earlier section, the update equation ensures that the word vectors are close to their related words and far from the unrelated words. Fig. 4.1 illustrates the learned word embeddings in a two-dimensional plane.

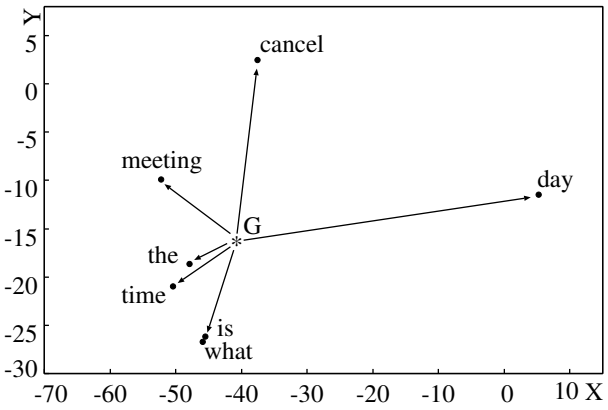

Figure 4.1: 2D-Illustration of sample word embedding space

Along with the word vectors, their mean (or centroid) vector G is also shown in Fig. 4.1. From the figure, we can observe that word *the*, which is the most generic word that occurred in all the contexts, is closest to the centroid. Since *is* and *what* have the same set of related and unrelated words; they have almost the same word vector representations. We can consider *day* and *cancel* as domain-related words as they are related to very few words and unrelated to most of the words. Since they are contextually biased and have the most discriminative power, they are placed at the boundaries far from the centroid. The word *meeting* is related to the generic words *the*, *is*, and *what* and the discriminative word *cancel*, so it is placed in the middle of these words. The word *time* occurs twice in the dataset, yet it occurs only along with the generic words. So it is placed close to the generic words *the*, *what*, *is*, *meeting* and far from the unrelated domain words *day* and *cancel*. Here, note that the words *meeting* and *time* occur an equal number of times in the corpus, and they will be treated as equally discriminative by the frequency

based weighting schemes. However, *meeting* is contextually more skewed compared to *time* since it occurs with the domain related word *cancel*. So, localnorm weighting assigns a slightly higher weight to *meeting* compared to *time*. Overall, we can observe that as the distance from the centroid increases, the skewness in contexts increases, and discriminative power increases. As per localnorm weighting scheme: *the* gets the lowest weight; *what*, *is*, *meeting*, and *time* get medium-range weights; *day* and *cancel* gets higher weights.

4.2.2 Denoising the embeddings

Noisy words are words that happen to occur in the given corpus but are actually irrelevant to the concepts discussed in the corpus. So, they possess very little relevant information content and are usually not good representation features. Word frequency distribution-based works have proved that the noise in the corpus is caused by the very low-frequency words [17]. Since the frequency based weighting schemes give higher weights to the rare words, by nature, these weighting schemes tend to exaggerate the noisy components and necessitate denoising for better representations.

In the case of word embeddings spatial distribution, since noisy words are unrelated to most of the words in the given corpus, the distributional hypothesis advocates positioning them far away from the rest of the words in the corpus. So, it is expected that words that are farthest from the center of the word cloud contain noise components. The proposed localnorm weighting scheme gives higher weights to the words as they move away from the center. So, similar to the frequency based weighting schemes, the localnorm weighting scheme also tends to give high weights to the noisy words. Since the sentence embeddings are composed by the averaging method, by its nature, it is prone to be affected by noise and outliers. So, the sentence embeddings can be purified by reducing the noise components in them.

The work in [17] shows that the noisy words cause the highest variance in the co-occurrence matrix. As the initial principal components (PCs) capture the highest variance, the effect of the noise components can be reduced by removing the initial PCs. Now, let us understand the relationship between the position of noisy words/ outliers in the embedding space and the PCs. We know that principal component analysis (PCA) tries to maximize the sum of squares of projections or minimize the sum of squares of residuals. This is achieved through the principal components passing through the centroid (or origin for mean-centered data) of the point cloud. So, the noisy words at the boundaries of the word cloud have bigger projections compared to the words nearer to the centroid. Thus, the noisy words cause the highest variance in the data. Consequently, the noisy components are captured by the initial principal components of the embedding space. So, denoising of the embeddings can be done by adapting the above-discussed technique of initial PCs removal to the embedding space as well.

However, this denoising can be done as a preprocessing step [67] and/or as post-processing step [9, 27]. In denoising as a pre-processing step, word embeddings are denoised by removing the initial PCs calculated over a stack of given word embeddings [67]. Then, the purified word embeddings are used as input for composition models. Whereas, in denoising as the post-processing step, sentence embeddings are constructed first through a composition function, and then they are denoised by removing

PCs calculated over the stack of given sentence embeddings [9, 27]. In our proposed sentence embedding method, we employ denoising as the post-processing step and consider the pre-processing step to be optional.

4.2.3 Proposed sentence embedding algorithms

4.2.3.1 Localnorm sentence embedding algorithm

The proposed sentence embedding algorithm is formally achieved by Algorithm 1. The algorithm can be divided into three basic steps: optional pre-processing step (Line 1), computing weighted sentence embeddings (Lines 2-6), and post-processing step (Line 7).

While computing weighted sentence embeddings, first, the centroid of all the word embeddings is obtained (Line 2), and then localnorm values of all the words are calculated as the distance between the corresponding word and the centroid (Lines 3-4). Finally, the embedding of each sentence is obtained as a weighted average of all word embeddings that occur in the given sentence.

The optional pre-processing step of denoising word embeddings is presented as a function in Lines 9-16. Here, first, the word embeddings are mean-centered using Lines 10-12, and then principal components over stacked word embeddings are calculated using PCA function (Line 13). Finally, the projections of word embeddings over d initial PCs are calculated and removed from the mean-centered embeddings to get the final denoised word embeddings, as shown in Line 15.

The post-processing step of denoising sentence embeddings is summarised in Lines 18-24. In this function, first, the PCs are calculated over stacked sentence embeddings, as shown in Line 19. Next, weights (λ_i) of the first m initial PCs are calculated as their normalized singular values (σ_i) (Line 21). Finally, as shown in Line 23, the weighted sum of projections of sentence embeddings over the m initial PCs are removed from the original embeddings to get the final denoised sentence embeddings.

4.2.3.2 Hybrid algorithms

From the previous discussions, we understood that frequency based weighting schemes and localnorm weighting scheme depend on two different properties of words to derive the weights. Frequency based methods use the information of word counts in the corpora, whereas the localnorm weighting scheme indirectly uses word co-occurrence information from the corpora. To understand the relationship between the word frequency and the localnorm values, we sampled 125K words from the Wikipedia corpus for which frequency information is available. To obtain localnorm values of sampled words, publicly available GloVe word embeddings are used. Fig. 4.2 shows the relationship between the spatial distribution and frequency distribution of the words. Here, X-axis represents the word index created by sorting them in ascending order of their localnorm value. On Y-axis, for each word in the index, corresponding word frequency, and localnorm values are plotted by normalizing them to the range [0-1].

From Fig. 4.2, it can be noted that many words which have the lowest localnorm value have the highest frequency. Both these properties, namely, low localnorm values and high frequencies suggest

Algorithm 1: Localnorm Sentence Embeddings

Input: Vocabulary V, Word embeddings $\{v_w : w \in V\}$, a set of sentences S
Output: Sentence Embeddings $\{v_s : s \in S\}$

/* Optional pre-processing step */
$\{v_w : w \in V\} \leftarrow Denoise\_WordEmbeddings(\{v_w : w \in V\}, d)$
$\mu \leftarrow \frac{1}{|V|} \sum_{w \in V} v_w$ // Compute the centroid
/* Compute the LocalNorm of all the words */
forall $w \in V$ **do**
  $Localnorm(w) \leftarrow dist(\mu, v_w)$
/* compute the sentence embeddings as weighed mean of words */
forall *sentence* $s \in S$ **do**
  $v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} v_w * Localnorm(w)$
/* Post-processing step */
$\{v_s : s \in S\} \leftarrow Denoise\_SentenceEmbeddings(\{v_s : s \in S\}, m)$

Def `Denoise_WordEmbeddings` $(\{v_w : w \in V\}, d)$**:**
  $\mu \leftarrow \frac{1}{|V|} \sum_{w \in V} v_w$
  forall $w \in V$ **do**
    $\tilde{v}_w \leftarrow v_w - \mu$
  $\{pc_1, pc_2, pc_3, ..., pc_d\} \leftarrow PCA(\{\tilde{v}_w : w \in V\})$
  forall $w \in V$ **do**
    $v'_w \leftarrow \tilde{v}_w - \sum_{i=1}^{d} (\tilde{v}_w {pc_i}^\top) pc_i$
  return $\{v'_w : w \in V\}$

Def `Denoise_SentenceEmbeddings` $(\{v_s : s \in S\}, m)$**:**
  $\{pc_1, pc_2, pc_3, ..., pc_m\}, \{\sigma_1, \sigma_2, \sigma_3, ..., \sigma_m\} \leftarrow PCA(\{v_s : s \in S\})$
  for *i in 1..m* **do**
    $\lambda_i \leftarrow \frac{{\sigma_i}^2}{\sum_{j=1}^{m} {\sigma_j}^2}$
  forall *sentence* $s \in S$ **do**
    $v'_s \leftarrow v_s - \sum_{i=1}^{m} \lambda_i (v_s {pc_i}^\top) pc_i$
  return $\{v'_s : s \in S\}$

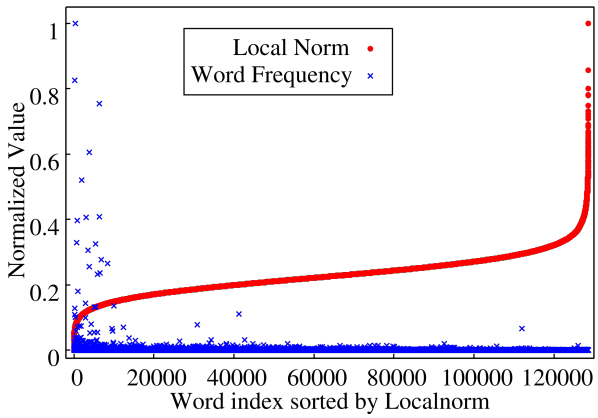

Figure 4.2: Word's localnorm and frequency relationship

that these words have low discriminative power, and hence they will get low weights in both the weighting schemes. This phenomenon can be understood as: usually, the words with the lowest localnorm values, i.e., words with the highest diversity in their contexts, tend to have the highest frequencies as well. However, for the rest of the words with medium to high localnorm values, we can observe no strong (negative) correlation between their localnorm values and the frequencies. So, it can be expected that weighting schemes based on these two different properties might assign weights in different ranges to the same words. Based on these observations, we propose employing hybrid weighting schemes to combine frequency distribution based (Tf-Idf, SIF, uSIF) and spatial distribution based (localnorm) weighting schemes as they can extract important words independently. In this work, we propose three simple hybrid weighting techniques: weights addition, vectors addition, and vectors concatenation.

In weights addition, as shown in Eq. 4.7, first, the weights from two different weighting schemes ($w1$ and $w2$) are added and multiplied with corresponding word embeddings (v_W). Finally, these intermediate sentence embeddings are denoised using the Principal Component Removal (PCR) method to get the final sentence embeddings (v_S). The proposed weights addition hybrid weighting scheme is formally summarized in Algorithm 2. Here, in Lines 1 and 2, the hybrid weight for each word is calculated as the sum of localnorm and frequency based weights. Note that both the individual weights are scaled down to the range [0-1] using the $Scale()$ function, ensuring both the weights have equal importance. Lines 3 and 4 perform weighted averaging operations to get sentence embeddings, and Line 5 performs post-processing denoising step on the sentence embeddings.

The vectors addition technique can be implemented in two ways: 1) As shown in Eq. 4.8, first, weighted averaged vectors are obtained from both the individual weighting schemes, then vector addition is performed to get the intermediate sentence embeddings, and finally, denoising is performed. 2) As shown in Eq. 4.9, first, the weighted averaged vectors from each scheme are obtained, then these vectors are denoised separately, and finally, these intermediate vectors are added to get the sentence representations.

The primary difference between these options is whether the denoising step is performed before or after

the vector addition operation. From the previous discussions, we understand that both the frequency-based and localnorm weighting schemes amplify the data's noise components. However, as they are derived from different properties of words, theoretically, they exaggerate different noise components. So, it is preferred to denoise embeddings independently first and then combine them through the vector addition operation, following the second option (Eq. 4.9). We observe the same phenomenon empirically as well. So, in this work, for vector addition technique implementation, we opt for the second option, and it is formally presented in Algorithm 3.

Weights Addition:

$$v_S = PCR(AVG((w1 + w2) * v_W)) \tag{4.7}$$

Vectors Addition:

$$v_S = PCR(AVG(w1 * v_W) + AVG(w2 * v_W)) \tag{4.8}$$

$$v_S = PCR(AVG(w1 * v_W)) + PCR(AVG(w2 * v_W)) \tag{4.9}$$

Vectors Concatenation:

$$v_S = PCR(CONCAT(AVG(w1 * v_W), AVG(w2 * v_W))) \tag{4.10}$$

$$v_S = CONCAT(PCR(AVG(w1 * v_W)), PCR(AVG(w2 * v_W))) \tag{4.11}$$

Similarly, there are two ways to implement the vector concatenation technique as well: 1) As shown in Eq. 4.10, first, the weighted averaged vectors are concatenated, and then overall denoising is performed. 2) As shown in Eq. 4.11, first, weighted averaged vectors are denoised independently first and then concatenated. For the same reason discussed above, we opt for the second option (Eq. 4.11) in vector concatenation technique implementation, and it is formally presented in Algorithm 4.

4.3 Experiments

Datasets: We empirically evaluate our sentence embedding method on the semantic textual similarity (STS) tasks. We experiment using the SemEval STS tasks datasets released for years 2012 to 2015 [4, 5, 6, 7]. Each STS task consists of 4-6 different datasets covering a wide variety of domains. We also consider SICK'14 dataset [61], Twitter'15 dataset [95] to validate our method over casual writing, and the most actively used STS-B benchmark dataset [19]. Though there are a total of 23 datasets, for simplicity of presentation, we present the average of the results over STS datasets related to each year. The STS datasets contain multiple pairs of sentences, with each pair annotated with a semantic similarity score on a scale of 0-5. The objective of the STS task is to predict the similarity score for each pair. The evaluation criterion is Pearson's coefficient between the predicted scores and the ground-truth scores.

Algorithm 2: Weights_Add(LocalNorm, Freq_Weight)

Input: Vocabulary V, Word embeddings $\{v_w : w \in V\}$, a set of sentences S
Output: Sentence Embeddings $\{v_s : s \in S\}$

forall $w \in V$ **do**
  $Hybrid\_Weight(w) \leftarrow Scale(LocalNorm(w)) + Scale(Freq\_Weight(w))$
forall *sentence* $s \in S$ **do**
  $v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} v_w * Hybrid\_Weight(w)$
$\{v_s : s \in S\} \leftarrow Denoise\_SentenceEmbeddings(\{v_s : s \in S\}, m)$

Algorithm 3: Vectors_Add(LocalNorm, Freq_Weight)

Input: Vocabulary V, Word embeddings $\{v_w : w \in V\}$, a set of sentences S
Output: Sentence Embeddings $\{v_s : s \in S\}$

forall *sentence* $s \in S$ **do**
  $v'_s \leftarrow \frac{1}{|s|} \sum_{w \in s} v_w * Scale(LocalNorm(w))$
  $v''_s \leftarrow \frac{1}{|s|} \sum_{w \in s} v_w * Scale(Freq\_Weight(w))$
$\{v'_s : s \in S\} \leftarrow Denoise\_SentenceEmbeddings(\{v'_s : s \in S\}, m)$
$\{v''_s : s \in S\} \leftarrow Denoise\_SentenceEmbeddings(\{v''_s : s \in S\}, m)$
forall *sentence* $s \in S$ **do**
  $v_s \leftarrow v'_s + v''_s$

Algorithm 4: Vectors_Concat(LocalNorm, Freq_Weight)

Input: Vocabulary V, Word embeddings $\{v_w : w \in V\}$, a set of sentences S
Output: Sentence Embeddings $\{v_s : s \in S\}$

forall *sentence* $s \in S$ **do**
  $v'_s \leftarrow \frac{1}{|s|} \sum_{w \in s} v_w * Scale(LocalNorm(w))$
  $v''_s \leftarrow \frac{1}{|s|} \sum_{w \in s} v_w * Scale(Freq\_Weight(w))$
$\{v'_s : s \in S\} \leftarrow Denoise\_SentenceEmbeddings(\{v'_s : s \in S\}, m)$
$\{v''_s : s \in S\} \leftarrow Denoise\_SentenceEmbeddings(\{v''_s : s \in S\}, m)$
forall *sentence* $s \in S$ **do**
  $v_s \leftarrow v'_s \oplus v''_s$ // $\oplus$ represents concatenation operation

We compare our method against a wide variety of both supervised methods and unsupervised methods. The details of each are as follow:

Supervised methods: All the following models are first initialized with PSL word embeddings [94], and the embeddings are then fine-tuned using paraphrase pairs dataset (PPDB) in a supervised fashion. Paragram-Phrase (PP) is an averaging method based on these supervised embeddings. PP-Proj. extends PP with an additional linear projection layer. Deep-averaging network(DAN) offers a generalization of the previous models by including multiple layers, nonlinear activation functions, and dropout techniques. Identity RNN (iRNN) is a variant of standard recurrent neural network (RNN) with identity activation function and weight matrices. Long short-term memory (LSTM) is another variant of RNN that can capture long-distance dependencies between the words in a sentence better than RNN. LSTM is experimented with and without an output gate, and the corresponding versions are denoted as LSTM (o.g.) and LSTM (no), respectively. Infersent is another supervised learning based sentence embedding method trained Stanford Natural Language Inference (SNLI) dataset and MultiGenre NLI (AllNLI) dataset.

Unsupervised methods: Skip-thought is an encoder-decoder model which is trained on BookCorpus dataset using an unsupervised learning approach to generate embeddings for sentences. DynaMax represents sentences as a fuzzy bag of words. WMD calculates the dissimilarity (distance) between two sentences based on their word alignment. BERTScore computes the similarity of two sentences as a sum of cosine similarities between their tokens' embeddings.

Averaging based unsupervised methods: ABT (All But the Top) introduces denoising of word embeddings as a pre-processing step before the simple averaging operation to find sentence embeddings. Tf-Idf (Term frequency — Inverse document frequency) weights the word embeddings before averaging based on their term frequencies and document frequencies. SIF (Smooth Inverse Frequency) derives word weights from a random walk model in the context of word embeddings. uSIF (unsupervised Smooth Inverse Frequency) uses an improved and unsupervised version of the SIF random walk model to derive the word weights. These frequency based weighting models further denoise the sentence embeddings by using PCA based post-processing techniques. The proposed localnorm sentence embedding method is implemented by deriving the weights from our novel weighting scheme along with the post-processing denoising step. Also, as discussed in Section 4.2.3.2, we implement all three variations of our hybrid weighting techniques to combine the proposed localnorm weighting approach with frequency based approaches. For pre-trained word embeddings, we use the standard GloVe[1] and Word2Vec[2] models. We have made the code for our localnorm sentence embedding algorithm publicly available[3].

[1] `https://nlp.stanford.edu/data/glove.6B.zip`

[2] `https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit?usp=sharing`

[3] The code is availabe at: `https://github.com/NarendraBabu-U/Localnorm`

Table 4.2: Experimental results (Pearson's r × 100) on textual similarity tasks.

Method	**STS'12**	**STS'13**	**STS'14**	**STS'15**	**SICK'14**	**Twitter'15**	**STS-B**
PP	58.7	55.8	70.9	75.8	71.6	52.9	-
PP-Proj.	60	56.8	71.3	74.8	71.6	52.8	-
DAN	56	54.2	69.5	72.7	70.7	53.7	-
RNN	48.1	44.7	57.7	57.2	61.2	45.1	-
iRNN	58.4	56.7	70.9	75.6	71.2	52.9	-
LSTM (no)	51	45.2	59.8	63.9	63.9	47.6	-
LSTM (o.g.)	46.4	41.5	51.5	56	59	36.1	-
InferSent (AllSNLI)	58.6	51.5	67.8	68.3	-	-	-
InferSent (SNLI)	57.1	50.4	66.2	65.2	-	-	-
Skip-thoughts	30.8	24.8	31.4	31	49.8	24.7	-
GloVe							
Average	52.7	39.66	53.64	54.22	66.53	29.62	39.57
Tf-Idf	58.29	52.81	64.16	60.81	69.23	33.35	60.83
SIF	56.95	56.31	69.43	72.26	72.19	49.34	67.56
uSIF	60.24	60.48	71.05	74.87	72.61	52.21	69.85
ABT	56.03	45.16	60.81	60.18	68.26	38.13	49.27
Dynamax	58.72	49.3	65.04	70.71	67.44	45.81	59.67
WMD	53.22	41.7	56.76	65.57	61.58	45.51	48.53
BERTScore	52.81	47.23	62.06	67.26	65.28	44.77	50.93
Localnorm	60.42	57.96	70.41	73.78	72.11	52.7	67.07
Weights_Add(Localnorm, uSIF)	61.75	62.03	73.31	75.87	**73.14**	52.59	70.74
Vectors_Add(Localnorm, uSIF)	61.72	62.2	**73.41**	75.84	73.09	52.13	70.67
Vectors_Concat(Localnorm, uSIF)	61.61	62.28	73.4	75.85	72.89	51.98	70.88
Weights_Add(Localnorm, Tf-Idf)	62.26	63.12	73.06	**76.57**	72.27	**54.68**	**71.08**
Vectors_Add(Localnorm, Tf-Idf)	**62.27**	63.23	73.05	**76.57**	72.06	54.43	70.71
Vectors_Concat(Localnorm, Tf-Idf)	62.08	**63.49**	72.96	76.56	71.51	54.66	70.84
Weights_Add(Localnorm, SIF)	60.23	60.49	71.86	74.88	73.04	53.16	69.35
Vectors_Add(Localnorm, SIF)	60.25	60.52	71.88	74.9	73.06	53.21	69.29
Vectors_Concat(Localnorm, SIF)	60.06	60.53	71.81	74.83	72.9	53.36	69.33
ABT+Localnorm	60.29	59.61	70.77	73.92	72.86	50.85	68.05
Word2Vec							
Average	54.22	51.33	65.87	67.27	71.29	35.53	58.48
Tf-Idf	59.21	57.9	68.59	71.43	70.81	32.96	67.45
SIF	56.26	57.99	69.75	72.43	72.34	40.27	68.12
uSIF	60.58	59.77	70.86	74.37	71.98	38.34	70.38
ABT	55.23	52.2	66.35	67.7	71.82	32.96	59.52
Dynamax	58.74	54.12	67.66	74.05	70.78	33.26	68.38
WMD	54.44	44.26	58.84	66.02	64.16	39.79	56.74
BERTScore	47.83	43.54	56.26	62.06	58.75	34.07	49.16
Localnorm	59.74	58.53	70.27	74.17	73.29	35.04	68.64
Weights_Add(Localnorm, uSIF)	59.36	59.58	71.45	74.53	72.14	39.72	70.19
Vectors_Add(Localnorm, uSIF)	**60.72**	59.57	71.45	74.54	72.14	39.76	70.21
Vectors_Concat(Localnorm, uSIF)	59.25	59.58	**71.47**	74.45	71.94	40.58	70.23
Weights_Add(Localnorm, Tf-Idf	59.74	60.57	70.53	**75.35**	71.61	41.85	70.33
Vectors_Add(Localnorm, Tf-Idf)	59.66	60.55	70.53	**75.35**	71.61	41.87	70.31
Vectors_Concat(Localnorm, Tf-Idf	59.6	**60.7**	70.48	75.33	71.19	**43.16**	**71.12**
Weights_Add(Localnorm, SIF)	58.57	58.59	71.1	73.88	72.04	41.26	69.17
Vectors_Add(Localnorm, SIF)	58.71	58.63	71.15	73.96	72.1	41.26	69.23
Vectors_Concat(Localnorm, SIF)	58.44	58.49	71.11	73.73	71.85	42.13	69.1
ABT+Localnorm	60.19	58.75	70.27	74.12	**73.28**	33.84	69.08

4.3.1 Results

Table 4.2 presents the results of the comparison study. First, all the deep learning based methods are presented, followed by the compositional models which use GloVe and Word2Vec embeddings.

All the deep learning based models, except skip-thought, use supervision either from the PPDB dataset (PP, PP-Proj., DAN, RNN, iRNN, LSTM (no), or LSTM (o.g.)) or the inference datasets (Infersent AllSNLI, SNLI). Therefore, these supervised deep learning based methods perform better than the unweighted averaging based method that uses unsupervised task-independent vectors. Interestingly, our simple weighted averaging method (localnorm) outperforms these complex deep learning based methods on almost all the datasets even without any task specific fine-tuning.

The proposed method beats all the baselines and approaches state-of-the-art. It outperforms unweighted averaging by large margins. In the case of GloVe, a performance improvement of over 31% (OnWN dataset- STS'13) for individual tasks and over 27% (STS-B) for datasets presented in Table 4.2 can be observed. A similar trend is observed for Word2Vec where the performance improvement is over 16% (OnWN dataset- STS'13) for individual tasks and over 7% on the presented datasets.

Localnorm yields better performance compared to other baseline methods. It shows a significant average performance gain of over 11% compared to ABT, which employs a denoising step prior to the averaging operation. Localnorm consistently outperforms the popular frequency based weighted averaging methods like Tf-Idf and SIF as well. Moreover, localnorm performs competitively compared to the current state-of-the-art uSIF model. Compared to uSIF, localnorm performs better on two datasets (STS'12, Twitter'15) and approaches the performance of uSIF for the rest of the datasets, the average difference margin being less than 1% (GloVe: 0.98% and Word2Vec: 0.94%).

Other compositional models, apart from the averaging based, such as WMD (word alignment approach) and BERTScore (word alignment approach with Idf weights) perform better than the simple averaging in case of GloVe but under-perform in the case of Word2Vec. Dynamax, a fuzzy based approach, outperforms simple averaging for both GloVe and Word2Vec embeddings. However, because of the introduction of weights, the localnorm method outperforms all these non-averaging based compositional models.

In Table 4.2, we have also presented the results obtained for 9 hybrid weighting schemes formed by combining the localnorm and frequency based weighting methods (Tf-Idf, SIF, uSIF) through the three merging techniques discussed in Section 4.2.3.2. From the table, we can observe that the hybrid methods consistently perform better than the corresponding individual methods. On average, the performance improvement of hybrid methods compared to individual methods is about 2-3%. The highest accuracies on most of the data sets are also achieved by the hybrid weighting schemes. The performance of three techniques, namely weight addition, vector addition, and vector concatenation, are very close.

4.3.1.1 Effect of denoising (ablation study)

To understand the effects of denoising techniques along with the weighting scheme, we have conducted an ablation study experiment. The pipeline of generating sentence embeddings consists of 3

components: Denoising word embeddings (DW) as pre-processing step, Weighting (W) based on localnorm measure, and Denoising sentence embeddings (DS) as post-processing step. Compared to the simple unweighted averaging, the impact of these components on the average performance gain over the 23 datasets for both GloVe and Word2Vec embeddings are presented in Fig.4.3 and Fig.4.4, respectively. In the case of GloVe, the individual performance gains of pre-processing step (DW) and localnorm weighting scheme (W) are 5.7% and 10.5%, respectively. The combination of both the pre-processing step and the weighting scheme (DW+W) yields a performance gain of 11%, which is better than their individual gains. The post processing step (DS) alone gives a performance gain of 13.2%, and along with the localnorm weighting scheme (W+DS), it gives a 15.9% gain. From these results, it can be observed that denoising as a post-processing step is more effective compared to it being a pre-processing step. When the pre-processing step is used along with the weighting scheme and post-processing step combination (DW+W+DS), the performance gain increases slightly, giving an overall gain of 16.3%. From Fig. 4.4, we can observe that the same trend is followed for Word2Vec embeddings as well.

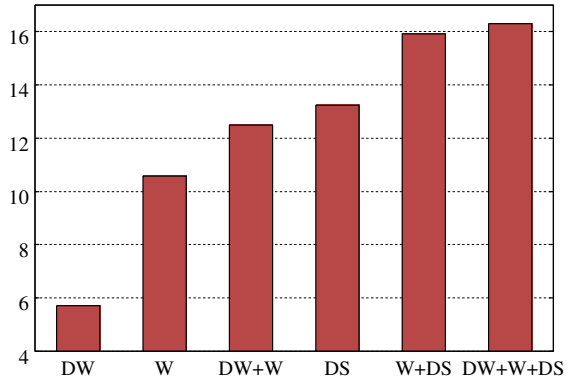

Figure 4.3: Ablation study results for GloVe

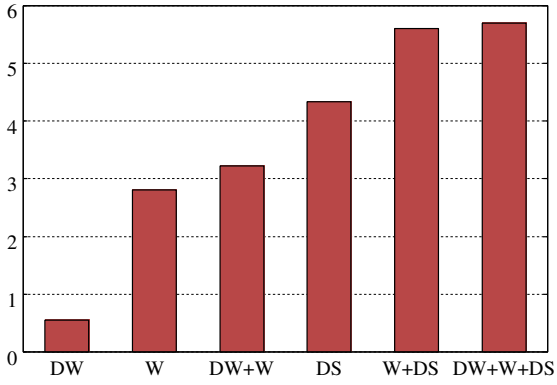

Figure 4.4: Ablation study results for Word2Vec

4.3.1.2 Qualitative analysis

For the purpose of qualitative analysis, we have presented 25 closest and farthest words from the centroid of the vocabulary from MSRpar dataset from STS'12 in Table 4.3. From the table, it can be noticed that most of the closest words to the centroid are generic words, and they lack discriminative power to distinguish one topic from another. It can also be seen that the farthest words in the dataset are mostly domain related words and have more discriminative power than the closest words. This observed phenomenon makes the working principle of localnorm weighting scheme evident over GloVe embeddings.

To understand how the words within a sentence are weighted by the localnorm weighting scheme, we have presented a sample of 10 sentences from STS-B dataset in Table 4.4. The table shows sample sentences along with localnorm weights for each word. In sentence 1, we can observe that the important words *cucumber* and *slicing* have the highest weights and period (.) has the lowest weight. In sentence

Table 4.3: Top 25 closest and farthest words from centroid for MSRpar2012 dataset

Closest words to centroid	Farthest words from centroid
meanwhile, latter, reminded, apparently, afterward, concerned, alluded, similarly, indeed, ultimately, contended, acknowledged, contend, bringing, likewise, subsequently, perhaps, contends, insisted, nonetheless, however, initially, assume, nobody, apart	12th-largest, tickets, billion, per-share, inning, chromosome, libeskind, klebold, decliners, yards, 1-11/32, biogen, kernel, non-manufacturing, nonmanufacturing, mph, reisen, de, subscription-free, pinal, advancers, enterprise, 500-stock, natsemi, ixic

2, *keyboard* and *playing* have the highest weights. Similar behaviour can be observed in the rest of the sentences as well.

From the previous discussions, it is established that frequency based approaches and localnorm approach capture different properties of the words. So, they might assign different weights to the same words and consequently give different similarity scores to the sentence pairs. Table 4.5 shows successful sample pairs from MSRvid dataset from STS'12 where localnorm scheme assigns similarity scores closer to the ground truth than uSIF does. Table 4.6 shows the failed sample pairs from MSRvid dataset from STS'12 dataset where uSIF assigns a similarity score closer to the ground truth than localnorm does. We now introduce the header notations [34] used in the Table 4.5 and 4.6 in detail below.

- GT: represents the given ground truth similarity score in a range of 0-5.
- NGT: represents the normalized ground truth similarity score. NGT is obtained by dividing the GT score by 5 so that it is in a range of 0-1.
- Localnorm_sc: represents the localnorm weighted embedding similarity score in a range of 0-1.
- uSIF_sc: represents the uSIF weighted embedding similarity score in a range of 0-1.
- Localnorm_err: represents absolute error $|Localnorm\_sc - NGT|$ between normalized ground truth similarity score and the localnorm weighted embedding similarity score.
- uSIF_err: represents the absolute error $|uSIF\_sc - NGT|$ between the ground truth similarity score and the uSIF weighted embedding similarity score.
- DIFF_err: represents absolute difference between uSIF_err and Localnorm_err. Examples where Localnorm_err performs better DIFF_err = Localnorm_err - uSIF_err (used in Table 4.5). Examples where uSIF performs better DIFF_err = uSIF_err - Localnorm_err (used in Table 4.6).

Table 4.4: 10 sample sentences with each word marked with its localnorm weight

Id	Sentences
1	a man is slicing a cucumber . 0.17 0.17 0.14 0.2 0.17 0.24 0.15
2	a man is playing a keyboard . 0.17 0.17 0.14 0.2 0.17 0.23 0.15
3	a group of men play soccer on the beach . 0.17 0.20 0.15 0.21 0.21 0.21 0.16 0.13 0.22 0.15
4	a girl is styling her hair . 0.17 0.19 0.14 0.22 0.20 0.24 0.15
5	a woman is cutting onions . 0.17 0.20 0.14 0.19 0.27 0.15
6	a man is riding an electronic bike . 0.17 0.17 0.14 0.20 0.18 0.23 0.21 0.15
7	south korea declares end to mers outbreak 0.18 0.21 0.14 0.15 0.14 0.20 0.22
8	what is your lid made of ? 0.15 0.14 0.19 0.21 0.14 0.15 0.16
9	you do not need any visa . 0.14 0.17 0.15 0.15 0.16 0.26 0.15
10	the cats are running through the grass . 0.13 0.22 0.17 0.19 0.17 0.13 0.24 0.15

Table 4.5: Examples where localnorm weighting performed better than uSIF weighting

Sentence 1	Sentence 2	GT	NGT	Localnorm_sc	uSIF_sc	Localnorm_err	uSIF_err	DIFF_err
three women are dancing .	the man is dancing .	1.6	0.32	0.572	0.732	0.252	0.412	0.16
raccoons are eating .	a man is eating .	1.5	0.3	0.373	0.51	0.073	0.21	0.137
a woman is slicing up some meat .	a woman is breading some meat .	2.25	0.45	0.538	0.672	0.088	0.222	0.134
a woman is talking on a cell phone .	a man and woman are talking on the phone .	3	0.6	0.647	0.761	0.047	0.161	0.114
a woman is deboning a fish .	a man catches a fish .	1.25	0.25	0.42	0.53	0.17	0.28	0.11
a woman is dancing and singing with other women .	a woman is dancing and singing in the rain .	3	0.6	0.611	0.72	0.011	0.12	0.108
a woman is chopping an onion .	a woman is slicing partially into half of an onion .	3.2	0.64	0.626	0.53	0.013	0.109	0.096
a woman puts flour on a piece of meat .	a woman is putting flour onto some meat .	5	1	0.794	0.699	0.205	0.3	0.095
a panda is climbing .	a man is climbing a rope .	1.6	0.32	0.39	0.472	0.07	0.152	0.081
the military officer barked at the recruits .	an officer is talking to recruits .	2.5	0.5	0.58	0.661	0.08	0.161	0.081

Table 4.6: Examples where uSIF weighting performed better than localnorm weighting

Sentence 1	Sentence 2	GT	NGT	Localnorm_sc	uSIF_sc	Localnorm_err	uSIF_err	DIFF_err
a young boy is holding a guitar .	a little boy is playing a guitar .	2.75	0.55	0.719	0.62	0.169	0.07	0.099
two puppies are playing with each other .	some men are playing soccer .	0.2	0.04	0.46	0.359	0.42	0.319	0.101
a woman is powdering her face .	the woman is tapping her fingers on the table .	0.8	0.16	0.363	0.26	0.203	0.1	0.103
a woman is filing her fingernails with an emery board .	a woman is feeding her baby with a bottle .	0.5	0.1	0.249	0.136	0.149	0.036	0.112
a man is pouring oil into a pan .	a man is pouring oil into a skillet .	4.2	0.84	0.631	0.751	0.208	0.088	0.12
a man is running on the road .	a car is driving down the road .	1	0.2	0.553	0.42	0.353	0.22	0.132
two men are talking .	two men are playing guitar .	1.5	0.3	0.47	0.27	0.17	0.029	0.141
two baby pandas are playing .	two pandas are laying together .	3	0.6	0.79	0.625	0.19	0.025	0.164
two boys are driving .	two bays are dancing .	0.6	0.12	0.32	0.084	0.2	0.035	0.165
a couple are walking .	some men are sawing .	0	0	0.319	0.143	0.319	0.143	0.176

4.4 Summary

In this Chapter, we proposed an alternative weighting scheme to frequency based weighting schemes that capture the contextual diversity in word embedding space. The proposed weighting scheme treats skewness or bias in the diversity of contexts a word occurs in, as the indicator of the word's importance. In spatial terms, the distance from the centroid of the word cloud to the given word (localnorm) is quantified as its weight. By employing this weighting scheme along with a denoising technique, a sentence embedding model is proposed. We empirically evaluated the proposed sentence embedding method against many baseline methods on semantic textual similarity (STS) tasks. The experimental results demonstrate that the localnorm based weighted averaging beats all the baseline methods and achieves performance competitive to the current frequency based state-of-the-art weighting schemes. Additionally, we developed a hybrid weighting scheme, and it has shown better performance than their corresponding individual weighting schemes and most deep learning based methods.

Chapter 5

Summary, Conclusions and Future Work

The quantity of unstructured and unlabeled textual data is expanding rapidly in the present digital age. To handle this textual data, numerous text representation models are being employed. In addition to modeling semantic relationships, the key aspects of a text representation model include high representational capacity, unsupervised learning ability, and feature interpretability. To improve in one or more of these aspects over existing models, we propose two word sequence representation models by exploiting the frequency distribution and spatial distribution properties (geometry) of the word embeddings.

Several works have shown that the simple vector averaging model, particularly in a transfer learning setting, where the word sequence embeddings are formed by composing the pre-trained word embeddings, outperforms the complex supervised deep learning models. Vector averaging composition function is simple, effective, and computationally inexpensive. However, vector averaging oversimplifies the longer texts as it represents the given word sequence in the same feature space as that of word embeddings. Also, it ignores the discriminative properties of the words and considers all the words equal.

We explored the opportunity to utilize the semantic information encoded in the word embeddings as the distances among them to develop better text representation models. In this thesis, as the first contribution, we propose an alternative unsupervised text representation framework to vector averaging in the context of longer texts such as documents. In the proposed novel document representation framework, a document is modeled as a vector of distances from multiple words in a different higher dimensional feature space. We proposed two methods for the selection of potential feature words and presented a distance function to measure the distance between the feature word and the document. We empirically evaluated these feature selection approaches and the distance measure.

We have conducted experiments on multiple document classification datasets, and the summary of the results is as follows:

- Experimental results on multiple data sets demonstrate that the proposed approaches, DIFW-fd and DIFW-sd, consistently improve the performance over baseline methods which include traditional and deep learning text representation models. Among the two proposed approaches, DIFW-fd outperforms DIFW-sd.

- Achieving better word features requires balancing between term specificity and topic coverage. We have advocated that moderately frequent words strike this balance well and possess stronger discriminative power than very frequent or rare words. Our empirical observations also support this, as the highest performance is attained with moderately frequent words.

- As the local norm value of the feature words is increasing the performance also increased consistently. This indicates that the domain related words that have high localnorm values are good features for representation. Also, this consistent performance gain inspired the localnorm based weighting scheme.

- Qualitative analysis reveals that the frequencies of general words located near the centroid vary widely and are not considered stop-words, rendering them difficult to detect through traditional frequency-based trimming methods. Nevertheless, localnorm provides a straightforward means of identifying these general words.

- Best performances are observed when the sizes of the closest topic (α), median topic (β), and farthest topic(γ) in DC, DM, and DF measures are approximately at 3%, 70%, and 15%, respectively, which are also the default values. Among the three distance measures, usually, DC and DF perform better than DM. The proposed DA measure fuses these three measures in its definition and exploits the individual powers of DC, DM, and DF. Hence, DA performs better than all of them irrespective of the feature words selection approach.

From the insights gained from above discussed work about the spatial distribution of words, as a second contribution, we propose an improved weighted average model in the context of shorter texts such as sentences. The existing weighting schemes use frequency based statistics to derive the weights of the words. These weighting schemes suffer from drawbacks such as dependency on external resources for frequency statistics. These external resources may not always be available publicly for niche domains or might have incomplete information. Also, these weighting schemes assume that the words are completely independent. These schemes judge the word importance based only on the number of times it occurred in the corpus and ignore the word-contexts co-occurrence distribution. The proposed weighting scheme treats skewness or bias in the diversity of contexts a word occurs in, as the indicator of the word's importance. In spatial terms, the distance from the centroid of the word cloud to the given word (localnorm) is quantified as its weight. Also, we propose hybrid weighting techniques to combine the localnorm (spatial property) weighting scheme with the frequency (statistical property) based weighting schemes as they capture two different properties of words.

We have conducted extensive experiments on multiple Semantic Textual Similarity (STS) datasets, and the summary of the results is as follows:

- The experimental results demonstrate that the localnorm based weighted averaging beats all the baseline methods by at least 7%-30% and achieves performance competitive to the current frequency based state-of-the-art weighting schemes.

- Hybrid weighting schemes which combine the frequency and localnorm weighting schemes perform better than their corresponding individual weighting schemes by at least 2%-3% and achieve the best performance on most of the datasets beating the deep learning models.

- The denoising step along with the weighting scheme contributes to the performance improvement. However, when it is used as a post-processing step, the performance gain is greater (15.9%) than when used it as a pre-processing step (11%). Applying denoising as both pre and post processing steps yields only a slight improvement compared to using it solely as a post-processing step (0.4% gain).

5.1 Conclusions

The conclusions are as follows:

- In the proposed semantic distance based text representation framework, the very idea of using distances to model the documents facilitated the framework with interpretability and high representational power. Even though the word embeddings themselves have uninterpretable latent features, their relative meanings can be interpreted as the proximities (or distances) between them in the embedding space. Additionally, in the embedding space, each word acts as a representative of the concept (or topic) expressed by its neighborhood. So, in the proposed framework, incorporating multiple important words as features provided the feature interpretability and the distance value quantified thematic differences between the feature word/topic and the given document. Since the document is understood from multiple perspectives through the different feature words (or topics), this approach yields a more comprehensive representation of the document's meaning resulting in higher representational power.

- The proposed localnorm weighting scheme is non-parametric and derives word weights from word embeddings themselves so it doesn't depend on any external statistical information. Also, interestingly, despite its simplicity, the proposed sentence embedding method outperformed complex deep learning based methods on almost all the datasets even without any task-specific fine-tuning.

- Both the proposed word sequence models are simple and support unsupervised settings.

- Overall, the spatial distribution based paradigm offers an alternative to traditional frequency based frameworks and opens up new research directions to develop novel text modeling approaches for improving the performance of text mining tasks.

5.2 Future work

- In this work, localnorm measure is employed as a feature selection criterion for the proposed text representation framework and also used as a weighting scheme for the sentence embedding method. However, fundamentally, the localnorm identifies the most important words (keywords) from given text content based on their spatial positions in the word embedding space. Hence, it can be used for keyword extraction. Extracting keywords is employed in a wide range of tasks such as text indexing, categorization, summarization, ranking, and search engines. Until now, frequency based approaches such as Tf-Idf weighting and its other variations dominated this direction of research. As part of our future work, we plan to extend this work and leverage localnorm as a competitive alternative and/or complementary approach to the frequency based approaches employed in the above mentioned text mining and information retrieval tasks.

- The proposed DA distance measure is a combination of DC, DM, and DF measures which are enhanced versions of existing distance measures from hierarchical clustering and object matching literature. Essentially, these DC, DM, DF, and DA measures capture different aspects while computing the similarity (or dissimilarity) between two data entities in a vector space. So, as part of future work, we want to investigate the applicability of these improved distance measures back in the methods which operate on the basis of similarity between the data points such as hierarchical clustering, object matching, and nearest neighbors.

- Following the popularity of Word2Vec and GloVe, numerous works have emerged regarding entity embeddings. Similar to words, entities whose relationships can be captured by a co-occurrence matrix, such as those found in recommendation systems and social networks (or graphs), can be represented as embeddings in a vector space. These entity embeddings also follow the similar principles of the distributional hypothesis and seek closer embeddings for similar entities. As part of future work, we will explore the potential applications of the proposed semantic distance based representation framework for entity sequences in domains such as recommendation systems and social networks. Also, we would like to test if the localnorm property holds true in these entity embedding spaces.

- In this work localnorm weighting technique is used to improve the vector averaging composition function. In our future work, we would like to develop methods to integrate the localnorm weights in other composition functions that utilize the word embeddings such as Word Mover's Distance (WMD), Dynamax, BertScore, and so on.

Related Publications

Publications related to Ph.D. thesis

1. *Narendra Babu Unnam*, P.Krishna Reddy; A document representation framework with interpretable features using pre-trained word embeddings; International Journal of Data Science and Analytics, Springer, Volumet:10, pages 49–64, 2019.

2. *Narendra Babu Unnam*, P.Krishna Reddy, Amit Pandey, Naresh Manwani; Journey to the center of the words: Word weighting scheme based on the geometry of word embeddings; 34th International Conference on Scientific and Statistical Database Management (SSDBM 2022); July 6-8, 2022 — Copenhagen, Denmark.

Other publications

1. Bhoomeendra singh sisodiya, *Narendra Babu Unnam*, P. Krishna Reddy, Apala Das, K.V.K. Santhy, V Balakista Reddy; Analyzing resourcefulness of paragraph for precedence retrieval; Proceedings of the 19th International Conference on Artificial Intelligence and Law (ICAIL) 2023.

2. Amit Pandey, Swayatta Daw, *Narendra Babu Unnam* and Vikram Pudi; Multilinguals at SemEval-2022 Task 11: Complex NER in Semantically Ambiguous Settings for Low Resource Languages; Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Association for Computational Linguistics (ACL).

3. Ravi Kumar, *Narendra Babu Unnam*, Uday Rage, Susheela Devi V, Kazuo Goda, Krishna Reddy P; A Novel Parameter-Free Energy Efficient Fuzzy Nearest Neighbor Classifier for Time Series Data; 2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2021, pp. 1-6, doi: 10.1109/FUZZ45933.2021.9494521.

4. Vyshnavi Gutta, *Narendra Babu Unnam*, and P. Krishna Reddy; An improved human-in-the-loop model for fine-grained object recognition with batch-based question answering; Proceedings of the 7th ACM IKDD CoDS and 25th COMAD. 2020. 134-142.

5. Vyshnavi Gutta, *Narendra Babu Unnam*, and P. Krishna Reddy; Fine-grained object recognition via Image and batch-based local question-answering; IEEE Access; *(Under Final Review)*

6. P.Krishna Reddy, *Narendra Babu Unnam*; A Smart Phone Based Field Diagnosis Guide for Farmers; 4th International Conference on Agriculture & Animal Husbandry, University of Hyderabad, Hyderabad, India, August 2019.

7. P.Krishna Reddy, *Narendra Babu Unnam*; Building a smart phone based field diagnosis guide for farmers by extending the concept of generalization; XIX International Plant Protection Congress (IPPC 2019), 10-14, November 2019, Hyderabad, India.

8. P.Krishna Reddy, *Narendra Babu Unnam*; Towards Building A Field Diagnosis Guide For Farmers; Asia-Pacific Federation for Information Technology in Agriculture (AFITA/WCCA); 2018.

Patents

1. P.Krishna Reddy, *Narendra Babu Unnam*, Revanth P, Srinivas Reddy A; System and Method for Classifying Plant Disorder Based on a Questionnaire that Includes Symptoms; Metayage R © IP Strategy Consulting LLP; Reference No: MY269047 INC; Application No: 202141012596; Filing/Priority Date: 23-March- 2021;, Filed the Complete Patent application with the Indian Patent Office.

Bibliography

[1] *Amazon ditched AI recruiting tool that favored men for technical jobs by Guardian News & Media Limited*. URL `https://www.theguardian.com/technology/2018/oct/10/amazon-hiring-ai-gender-bias-recruiting-engine`. [Online; Accessed 20-04-2023].

[2] Calculating a Weighted Average (Average of Averages) by Analytics Edge Inc. `https://help.analyticsedge.com/howto/calculating-the-average-of-averages/`, 2014. [Online; Accessed 20-04-2023].

[3] M. Abdalla, M. Abdalla, G. Hirst, F. Rudzicz, et al. Exploring the privacy-preserving properties of word embeddings: algorithmic validation study. *Journal of medical internet research*, Volume 22(7):pages e18055, 2020.

[4] E. Agirre, D. Cer, M. Diab, and A. Gonzalez-Agirre. Semeval-2012 task 6: A pilot on semantic textual similarity. In *SEM 2012: The first joint conference on lexical and computational semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the sixth International workshop on semantic evaluation*, pages 385–393, 2012.

[5] E. Agirre, D. Cer, M. Diab, A. Gonzalez-Agirre, and W. Guo. Semeval-2013 shared task: Semantic textual similarity. In *SEM 2013:Second joint conference on lexical and computational semantics–volume 1: Proceedings of the main conference and the shared task: semantic textual similarity*, pages 32–43, 2013.

[6] E. Agirre, C. Banea, C. Cardie, D. Cer, M. Diab, A. Gonzalez-Agirre, W. Guo, R. Mihalcea, G. Rigau, and J. Wiebe. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International workshop on semantic evaluation (SemEval 2014)*, pages 81–91, 2014.

[7] E. Agirre, C. Banea, C. Cardie, D. Cer, M. Diab, A. Gonzalez-Agirre, W. Guo, I. Lopez-Gazpio, M. Maritxalar, R. Mihalcea, et al. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th International workshop on semantic evaluation (SemEval 2015)*, pages 252–263, 2015.

[8] S. Arora, Y. Li, Y. Liang, T. Ma, and A. Risteski. A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for computational linguistics*, Volume 4:pages 385–399, 2016.

[9] S. Arora, Y. Liang, and T. Ma. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of International conference on learning representations (ICLR)*, 2017.

[10] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[11] M. Baroni. 39 distributions in text. *Corpus linguistics: An international handbook volume*, Volume 2:pages 803–822, 2005.

[12] J. R. Bellegarda. Latent semantic mapping [information retrieval]. *IEEE signal processing magazine*, Volume 22(5):pages 70–80, 2005.

[13] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, Volume 35(8):pages 1798–1828, 2013.

[14] R. K. Blashfield. Mixture model tests of cluster analysis: Accuracy of four agglomerative hierarchical methods. *Psychological bulletin*, Volume 83(3):pages 377, 1976.

[15] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine learning research*, Volume 3:pages 993–1022, 2003.

[16] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Transactions of the Association for computational linguistics*, Volume 5:pages 135–146, 2017.

[17] J. A. Bullinaria and J. P. Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, Volume 39(3):pages 510–526, 2007.

[18] J. Camacho-Collados and M. T. Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *Journal of artificial intelligence research*, Volume 63:pages 743–788, 2018.

[19] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv:1708.00055*, 2017.

[20] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv:2003.10555*, 2020.

[21] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International conference on machine learning*, pages 160–167. ACM, 2008.

[22] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, Volume 12(Aug):pages 2493–2537, 2011.

[23] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun. Very deep convolutional networks for text classification. *Proceedings of the 15th conference of the European chapter of the Association for computational linguistics*, Volume 1:pages 1107–1116, 2017.

[24] C. De Boom, S. Van Canneyt, T. Demeester, and B. Dhoedt. Representation learning for very short texts using weighted word embedding aggregation. *Pattern recognition letters*, Volume 80: pages 150–156, 2016.

[25] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018.

[26] M.-P. Dubuisson and A. K. Jain. A modified hausdorff distance for object matching. In *Proceedings of 12th International conference on pattern recognition*, pages 566–568. IEEE, 1994.

[27] K. Ethayarajh. Unsupervised random walk sentence embeddings: A strong but simple baseline. In *Proceedings of the 3rd workshop on representation learning for NLP, Association for computational linguistics (ACL)*, pages 91–100, 2018.

[28] J. R. Firth. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, 1957.

[29] P. W. Foltz, W. Kintsch, and T. K. Landauer. The measurement of textual coherence with latent semantic analysis. *Discourse processes*, Volume 25(2-3):pages 285–307, 1998.

[30] J. Friedman, T. Hastie, and R. Tibshirani. The elements of statistical learning. *Springer series in statistics*, Volume 1(10):pages 18–20, 2001.

[31] F. S. Gharehchopogh and Z. A. Khalifelu. Analysis and evaluation of unstructured data: text mining versus natural language processing. In *5th International conference on application of information and communication technologies*, pages 1–4. IEEE, 2011.

[32] J. W. M. B. K. Gimpel and K. Livescu. Towards universal paraphrastic sentence embeddings. *arXiv:1511.08198*, 2015.

[33] B. Goodman and S. Flaxman. European union regulations on algorithmic decision-making and a right to explanation. *AI magazine*, Volume 38(3):pages 50–57, 2017.

[34] V. Gupta, A. Saw, P. Nokhiz, P. Netrapalli, P. Rai, and P. Talukdar. P-sif: Document embeddings using partition averaging. In *Proceedings of the Association for the advancement of artificial intelligence (AAAI)*, volume 34, pages 7863–7870, 2020.

[35] Z. S. Harris. Distributional structure. *Word*, Volume 10(2-3):pages 146–162, 1954.

[36] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*, Volume 15(9): pages 850–863, 1993.

[37] I. Iacobacci, M. T. Pilehvar, and R. Navigli. Sensembed: Learning sense embeddings for word and relational similarity. *Proceedings of the 53rd annual meeting of the Association for computational linguistics and the 7th International joint conference on natural language processing*, Volume 1:pages 95–105, 2015.

[38] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd annual meeting of the Association for computational linguistics and the 7th International joint conference on natural language processing*, pages 1681–1691. Association for computational linguistics, 2015.

[39] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, Volume 32(3):pages 241–254, 1967.

[40] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *arXiv:1607.01759*, 2016.

[41] S. M. Julia Angwin, Jeff Larson and P. Lauren Kirchner. *Machine Bias: There's software used across the country to predict future criminals. And it's biased against blacks.* URL `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`. [Online; Accessed 20-04-2023].

[42] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. *arXiv:1404.2188*, 2014.

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

[44] Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the International conference on empirical methods in natural language processing*, pages 1746–1751. Association for Computational Linguistics, 2014.

[45] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-thought vectors. In *Proceedings of the Advances in neural information processing systems*, pages 3294–3302, 2015.

[46] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger. From word embeddings to document distances. In *Proceedings of International conference on machine learning*, pages 957–966. PMLR, 2015.

[47] S. Lai, L. Xu, K. Liu, and J. Zhao. Recurrent convolutional neural networks for text classification. *In Proceedings of Association for the advancement of artificial intelligence (AAAI)*, Volume 29 (1):pages 2267–2273, 2015.

[48] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv:1909.11942*, 2019.

[49] G. N. Lance and W. T. Williams. A general theory of classificatory sorting strategies: 1. hierarchical systems. *The computer journal*, Volume 9(4):pages 373–380, 1967.

[50] T. K. Landauer and S. T. Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, Volume 104(2):pages 211, 1997.

[51] J. H. Lau and T. Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv:1607.05368*, 2016.

[52] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International conference on machine learning (ICML)*, pages 1188–1196, 2014.

[53] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, Volume 27:pages 2177–2185, 2014.

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

[55] P. Liu, X. Qiu, and X. Huang. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the 25th International joint conference on artificial intelligence*, page 2873–2879, 2016.

[56] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv:1907.11692*, 2019.

[57] Z. Liu, Y. Lin, and M. Sun. *Word Representation*, pages 13–41. Springer, 2020. ISBN 978-981-15-5573-2.

[58] Z. Liu, Y. Lin, and M. Sun. *Document Representation*, pages 91–123. Springer, 2020. ISBN 978-981-15-5573-2.

[59] C. Manning and H. Schutze. *Foundations of statistical natural language processing*. MIT press, 1999.

[60] C. Manning, P. Raghavan, and H. Schütze. Introduction to information retrieval. *Natural language engineering*, Volume 16(1):pages 382, 2010.

[61] M. Marelli, S. Menini, M. Baroni, L. Bentivogli, R. Bernardi, and R. Zamparelli. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the 9th International conference on language resources and evaluation (LREC)*, pages 216–223. European language resources association, 2014.

[62] B. McCann, J. Bradbury, C. Xiong, and R. Socher. Learned in translation: Contextualized word vectors. *Advances in neural information processing systems*, 2017.

[63] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv:1301.3781*, 2013.

[64] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of Advances in neural information processing systems*, pages 3111–3119, 2013.

[65] T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. *In the Proceedings of the North american chapter of the Association for computational linguistics: Human language technologies (ACL:HLT)*, Volume 13:pages 746–751, 2013.

[66] J. f. Mitchell and M. Lapata. Vector-based models of semantic composition. In *Proceedings of Association for computational linguistics*, pages 236–244, 2008.

[67] J. Mu, S. Bhat, and P. Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. *arXiv:1702.01417*, 2017.

[68] M. Nickel and D. Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347, 2017.

[69] S. Nutanong, E. H. Jacox, and H. Samet. An incremental hausdorff distance calculation algorithm. *Proceedings of the very large data base endowment*, Volume 4(8):pages 506–517, 2011.

[70] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the International conference on empirical methods in natural language processing*, pages 1532–1543, 2014.

[71] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 conference of the North American chapter of the Association for computational linguistics: Human language technologies*, pages 2227–2237. Association for computational linguistics, 2018.

[72] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv:1908.10084*, 2019.

[73] S. M. Rezaeinia, R. Rahmani, A. Ghodsi, and H. Veisi. Sentiment analysis based on improved pre-trained word embeddings. *Expert systems with applications*, Volume 117:pages 139–147, 2019.

[74] S. E. Robertson. Understanding inverse document frequency: on theoretical arguments for idf. *J. Documentation*, Volume 60:pages 503–520, 2004.

[75] S. E. Robertson and K. S. Jones. Relevance weighting of search terms. *Journal of the american society for information science*, Volume 27(3):pages 129–146, 1976.

[76] X. Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.

[77] H. Rubenstein and J. B. Goodenough. Contextual correlates of synonymy. *Communications of the association for computing machinery*, Volume 8(10):pages 627–633, 1965.

[78] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. *Proceedings of 6th International conference on computer vision, IEEE*, pages 59–66, 1998.

[79] M. Sahlgren. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. PhD thesis, 2006.

[80] M. Sahlgren. The distributional hypothesis. *Italian journal of disability studies*, Volume 20:pages 33–53, 2008.

[81] G. Salton and C.-S. Yang. On the specification of term values in automatic indexing. *Journal of documentation*, Volume 29(4):pages 351–372, 1973.

[82] K. N. Singh, A. Dorendro, H. M. Devi, and A. K. Mahanta. Analysis of changing trends in textual data representation. *Recent trends in image processing and pattern recognition*, pages 237–251, 2021.

[83] P. Singh and A. Mukerjee. Words are not equal: Graded weighting model for building composite document vectors. In *Proceedings of the 12th International conference on natural language processing*, pages 11–19. NLP Association of India, 2015.

[84] P. H. Sneath. The application of computers to taxonomy. *Microbiology*, Volume 17(1):pages 201–226, 1957.

[85] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the Joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211. Association for computational linguistics, 2012.

[86] R. Sokal and P. Sneath. Principles of numerical taxonomy san francisco. *WH Friedman and Company*, 1963.

[87] K. Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, Volume 28(1):pages 11–21, 1972.

[88] R. Speer, J. Chin, and C. Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. *arXiv 1612.03975*, 2018.

[89] K. S. Tai, R. Socher, and C. D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd annual meeting of the Association for computational linguistics and the 7th International joint conference on natural language processing*, pages 1556–1566. Association for computational linguistics, 2015.

[90] A.-H. Tan et al. Text mining: The state of the art and the challenges. In *Proceedings of the Pacific-asia conference on knowledge discovery and data mining workshop on knowledge disocovery from advanced databases*, volume 8, pages 65–70, 1999.

[91] D. Tang, F. Wei, B. Qin, N. Yang, T. Liu, and M. Zhou. Sentiment embeddings with applications to sentiment analysis. *IEEE transactions on knowledge and data engineering*, Volume 28(2): pages 496–509, 2015.

[92] B. Wang, J. Xu, J. Li, C. Hu, and J.-S. Pan. Scene text recognition algorithm based on faster rcnn. In *First International conference on electronics instrumentation & information systems*, pages 1–4. IEEE, 2017.

[93] C. Wartena, R. Brussee, and W. Slakhorst. Keyword extraction using word co-occurrence. In *Workshops on database and expert systems applications*, pages 54–58. IEEE, 2010.

[94] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for computational linguistics*, Volume 3:pages 345–358, 2015.

[95] W. Xu, C. Callison-Burch, and W. B. Dolan. Semeval-2015 task 1: Paraphrase and semantic similarity in twitter. In *Proceedings of the 9th International workshop on semantic evaluation*, pages 1–11, 2015.

[96] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the north american chapter of the Association for computational linguistics: Human language technologies*, pages 1480–1489. Association for computational linguistics, 2016.

[97] P. Yildirim and D. Birant. K-linkage: A new agglomerative approach for hierarchical clustering. *Advances in electrical and computer engineering*, Volume 17(4):pages 77–88, 2017.

[98] S. Yokoi, R. Takahashi, R. Akama, J. Suzuki, and K. Inui. Word rotator's distance. In *Proceedings of the International conference on empirical methods in natural language processing*, pages 2944–2960. Association for computational linguistics, 2020.

[99] L.-C. Yu, J. Wang, K. R. Lai, and X. Zhang. Refining word embeddings for sentiment analysis. In *Proceedings of the International conference on empirical methods in natural language processing*, pages 534–539, 2017.

[100] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with BERT. In *Proceedings of 8th International conference on learning representations*. OpenReview.net, 2020.

[101] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Proceedings of advances in neural information processing systems*, pages 649–657, 2015.

[102] V. Zhelezniak, A. Savkov, A. Shen, F. Moramarco, J. Flann, and N. Y. Hammerla. Don't settle for average, go for the max: Fuzzy sets and max-pooled word vectors. In *Proceedings of International conference on learning representations*, 2019.

[103] C. Zhou, C. Sun, Z. Liu, and F. Lau. A c-lstm neural network for text classification. *arXiv:1511.08630*, 2015.

[104] G. K. Zipf. *The psycho-biology of language: An introduction to dynamic philology*. Routledge, 2013.