Subspace Representation for Natural Language Processing

March 2022

Erica Kido Shimomoto

Subspace Representation for Natural Language Processing

Graduate School of Systems and Information Engineering
University of Tsukuba

March 2022

Erica Kido Shimomoto

Abstract

Using word embeddings has become the default strategy when representing natural language data. Traditional word embeddings learn a fixed real-valued vector representation of each word in a corpus that encodes semantic information. This thesis proposes using subspaces to represent sentences and texts based on such word embeddings. The motivation behind this idea comes from a crucial aspect of these embeddings: Performing arithmetic and distance calculations between two word vectors can give us information about how their respective words relate semantically. Understanding that a sentence or a text can be represented as a set of word vectors, a natural extension is to compare two texts based on the subspaces spanned by their word vectors. We call such subspace the *word subspace*. The word subspace is a simple representation that does not require computationally intensive learning and can be derived from sentences and texts with different lengths. The basis vectors of this subspace are obtained by applying the principal components analysis (PCA) without data centering and can be regarded as the main hidden topics of the given text. Once represented as subspaces, we can efficiently compare texts with different lengths in terms of subspace similarity. Despite these appealing characteristics, this representation has not yet been explored to its full potential as, in general, subspaces are blindly vectorized (e.g., concatenation of basis vectors), so they can work with standard machine learning (ML) algorithms. We argue this might not be the best strategy as there is already a vast theory on general-purpose ML methods that can work directly with the subspace representation. Therefore, we propose solving several tasks in NLP based on such theory. We explore the geometry behind the word embeddings to perform a guided decision on the most appropriated subspace-based method for each task and demonstrate the effectiveness of such representation through experimental results.

Acknowledgements

My journey throughout this Ph.D. course had many ups and downs. As if completing a Ph.D. was not hard enough, my peers and I had to go through it amidst a pandemic. I could not have completed this thesis without the help and support of those around me. Therefore, I would like to express my sincere gratitude to everyone who has supported me during this journey.

First of all, I would like to thank the financial support of the Ministry of Education, Culture, Sports, Science, and Technology (MEXT) for providing the scholarship through which I could pursue my Master's and Doctoral degrees. I also would like to thank the University of Tsukuba for providing academic and technical support to my research.

I would like to express my deepest gratitude to my advisor, Prof. Kazuhiro Fukui, whose guidance and continuous support have given me the motivation and strength necessary to complete this course. With him, not only I learned about the fascinating theory behind the subspace representation but also learned how to be a researcher that praises research outcomes without sacrificing my own integrity by respecting my body and mind limits. Above all, thank you for your understanding through the chaos this pandemic has been.

I would also like to extend my gratitude to the rest of my thesis committee: Prof. Keisuke Kameyama, Prof. Toshiyuki Amagasa, Prof. Takashi Inui, and Prof. Hajime Nobuhara. Your insights and questions helped me understand my research's positive and negative points, allowing me to explore and improve this work.

Furthermore, exploring the field of natural language processing from a computer vision background was extremely challenging. Completing my degree requirements would not have been possible without the support of Prof. François Portet from the University of Grenoble-Alpes. I am incredibly grateful for all the inputs about my research. It was an honor to work together.

I am also deeply grateful to Prof. Hiroya Takamura for allowing me to join his team at AIST as a research assistant. Working with his team has been teaching me many fundamental skills I still lack as a researcher and allowed me to see new possibilities for future research. I'd like to extend my gratitude to Prof. Yusuke Miyao, Prof. Ichiro Kobayashi, and Dr. Edison Marrese-Taylor for the guidance and weekly discussions. I also cannot forget to thank Ms. Ai Tomobe for all the kind assistance since my start on AIST. I am pleased to work and grow together.

I also would like to thank my labmates from the Computer Vision Laboratory: Lincon Souza, for all the guidance, support, and above all, your friendship throughout my Master's and beginning of my Ph.D. course; Sogi, for always giving me a fresh perspective about my research; Suzana, for your companionship as a fellow woman researcher and your friendship; and all other members for the excellent research discussions and conversations. A warm thanks to Ms. Hiroko Sawabe for all the support with the paperwork and all the lovely treats.

Finally, I would like to thank my friends and family for their unconditional love and support. Thanks to my boyfriend, Denis, for never judging my crazy plans; Thanks to my brother, Eric, for being the anchor keeping our family steady; Thanks to my dad, Hiroshi, for being my biggest fan and always motivating me to go forward; And thank you to my mom and grandma, Yassue and Sueko, for teaching me how to be a strong and independent woman. I hope one day I can become at least half of what you were.

Contents

List of Figures

Chapter 1

Introduction

Recently, using word embeddings has become the default strategy when representing natural language data in machine learning. Word embeddings are real-valued representations of words, learned by neural networks, and can be separated into two different categories: Non-contextualized word embeddings, such as word2vec [1], GloVe [2]; and contextualized word embeddings, such as ElMo [3], BERT [4]. While both types of embeddings aim at learning a continuous vector representation for natural language data, they differ in how they achieve the embeddings.

Non-contextualized word embeddings, also referred to as traditional word embeddings, learn a fixed real-valued vector representation of each word in a corpus. They embody the distributional hypothesis of meaning [5], according to which the meaning of words is defined by contexts in which they co-occur. They can then generate a real-valued vector for each word in the corpus that encodes its semantic information. Once a model is trained, we can easily infer the word embeddings through a look-up table.

On the other hand, contextualized embeddings use mechanisms such as attention [6] and LSTMs [3] to produce vector representations for words that vary according to their context words. Therefore, these embeddings can represent the different meanings a word can carry, improving the performance in several natural language tasks. However, inference of the word embeddings can be time-consuming as the whole sentence or document has to be processed by the language model to obtain the word embeddings.

Understanding that a sentence or a text can be represented as a set of word vectors, a natural extension is to compare two texts based on the subspaces spanned by their word vectors.

In this thesis, we propose representing sentences and texts as linear subspaces from their word embeddings to perform different tasks in natural language processing. We call such subspace the *word subspace*. The word subspace is a simple representation that does not require computationally intensive learning and can be derived from sentences with different lengths.

Specifically, we focus on the subspace representation of non-contextualized word embeddings. One crucial aspect of this type of word embedding is that we can get information about the semantic relationship between two words by performing arithmetic and distance calculation between their word vectors. However, rather than looking at independent word vectors, we are interested in higher-level natural language structures such as sentences, paragraphs, and documents (i.e., set of vectors). Therefore, it is desirable to have a text representation based on a set of these word vectors, and we

achieve this through the word subspace representation.

In the following, we first present a brief overview of text models based on non-contextualized word embeddings, targeted to solve different natural language processing (NLP) downstream tasks. Then, we present our motivations for using the subspace representation and discuss the objectives of this thesis. Finally, we present the thesis structure.

1.1 Related Work

Word embeddings are neural networks trained on a large corpus targeting text representation and can be plugged into several downstream task models (e.g., text classification, text summarization, among others) to automatically improve their performance. The results can be improved by further training a model to generate sentence embeddings [7], [8].

As shown by Perone et al. [9], sentence encoding architectures based on recurrent neural networks (RNNs) [10], [3] lead to very high performance in several NLP downstream tasks. Nevertheless, transformer-based models, such as BERT [11] and GPT-2 [12] achieve state of the art in several tasks. However, as these techniques are substantially more expensive to train and apply than traditional word embeddings [13], they are limited to when high computational power is available.

In situations where such computational power is not available, an alternative is to learn sentence representations from traditional word embeddings. The distributional hypothesis of meaning [5] motivated the development of these word embeddings, such as the word2vec [1]. This hypothesis states that the meaning of words is defined by contexts in which they co-occur. Aiming at expanding this framework also to generate sentence representations, the doc2vec [14] was proposed. It follows the same architecture as the word2vec, where instead of learning word embeddings, it learns paragraph/document embeddings.

However, learning this direct estimation of the surrounding contexts of a phrase can have a significant sampling error, as phrases are far more sparse than individual words. As explained by Tian, Okazaki, and Inui [15], in a moderate size corpus such as the British National Corpus (BNC), a total of 16000 lemmatized words are observed more than 200 times, but there are only 46000 bigrams formed by them, far less than the 16000^2 possibilities for two-word combinations. In larger corpora, we might only observe rare words due to Zipf's Law, so most of the two-word combinations will always be rare or unseen.

Such understanding motivates the construction of sentence and document embeddings from combining word vectors [16]. This strategy also follows the compositional hypothesis that the meaning of sentences is composed by the meaning of their constituent words.

Considering the above discussion, non-parameterized sentence and document embeddings, i.e., a representation based on word embeddings that do not require any further learning of the word embeddings, can be more appropriate. The simplest stratagem is to take the average of the sentence's word vectors and use it as a feature to represent the whole sentence or text. This average vector can then be used to train traditional machine learning algorithms, such as support vector machines. Such a simple representation is effective and has motivated the study of the distribution of the word vectors in a text by using different types of mean.

For example, Arora et al. [17] proposed a weighted sum of the word vectors to generate the sentence representation, which outperformed many sophisticated neural network models in sentence

embeddings tasks. The weights are obtained by calculating the corresponding word's smooth inverse frequency (SIF), generated through a random walk model. Peters at al. [18] proposed a sentence embedding generated by concatenating the power mean of the sentence's word vectors. By using different power levels, we can retrieve many well-known means, such as the arithmetic mean, the geometric mean, and the harmonic mean.

Understanding that a sentence or a text can be represented as a set of word vectors, a natural extension is to compare two texts based on the subspaces spanned by their word vectors. Using subspaces along with textual data has been explored for several decades. For example, the latent semantic analysis (LSA) [19] applied the singular value decomposition to the word co-occurrence matrix of a corpus, generating a low-rank representation of words and documents. After the development of the word embeddings, works such as Yaghoobzadeh and Schcütz [20] evaluated if the word vector space generated by a word embedding framework contained the subspace necessary to represent different facets of the words. Word embeddings have also been generated by projecting the one-hot representation of words onto a latent space generated by the canonical correlation analysis, generating context-dependent word embeddings [21]. Nevertheless, methods, such as the principal components analysis (PCA), have been utilized to perform dimension reduction of the word vector space [22], which achieved similar or better results than the original embeddings in several benchmarks.

The principal components analysis has been proposed to generate a subspace representation for textual data based on traditional word embeddings, which we refer to as word subspace. This representation was first proposed to model sentences [23] and has since been demonstrated to be a powerful model for understanding and solving different NLP tasks, such as word compositionality [24], word polysemy [25], and text summarization [26]. Nevertheless, when represented as subspaces, we can easily compare textual data in terms of subspace similarity. Besides, the subspace representation generated by QR-decomposition has also been used to analyze if each new word in a sentence brings a different orthogonal basis to the subspace spanned by the previous word vectors [27]. Using a similar strategy, Sbert-Wk [28] uses QR-decomposition to analyze different information given by each of the layers in BERT to create a sentence embedding.

However, the word subspace representation has not been explored to its full potential. It is usually used as a baseline for sentence models using spectral methods, such as the dynamic mode decomposition [29], [30] and discrete cosine transformation [31], [32], where a sentence is represented as the concatenation of the first principal components of its subspace. While such approach leads to decent performance, all the interesting properties of the word subspace are lost in this process.

In the following, we discuss in more depth the properties that motivate the use of the subspace model for natural language data.

1.2 Motivations

Linear subspaces have been widely used in the classification of image sets, tackling tasks such as face [33], [34], hand shape [35], [36], and motion recognition [37]–[39]; and have also been applied in bioacoustic signal classification [40], [41]. Moreover, they have also been applied to artificially generated features, such as CNN features from images [42], [43] and graph embeddings [44],

succeeding in classification tasks in several data modalities. By this representation, sets of features representing a single entity, such as a set of images taken from different angles of a person, a set of frames from a video, lagged feature vectors from an audio signal, among others, are usually modeled as lower-dimensional linear subspaces in the original high-dimensional feature space by using PCA. A solid theoretical foundation on subspace-based methods has been developed throughout all these applications. These methods can directly work with the subspace representation, which motivates its application in other modalities.

In the natural language processing field, a great motivation to apply the subspace model is the subspace uniqueness property. While the subspace representation is a unique entity in a high-dimensional vector space, it can be spanned by different sets of basis vectors. Analogously, we can express a unique concept by using different words. This parallelism raises the possibility of modeling the same concept word subspace from different texts containing different words.

There are several advantages to working with such representation. First, since modeling a subspace requires only applying PCA to a set of features, it has a low computational cost. Moreover, the subspace-based methods work well when little data is available, as PCA can represent as much variance as possible in a small number of dimensions, represented by the principal components. Therefore, variations such as rotation and illumination in computer vision can be captured as linear combinations of the principal components.

Most importantly, the subspace model is highly interpretable. Given that we have some intuition on the features, e.g., we understand how two vectors relate to each other, it is possible to visualize and interpret the subspace's basis vectors, as the subspace-based methods explore the relationship between subspaces from a geometrical perspective. Interpretability is one of the characteristics that sets the subspace models apart from the current trend focused on deep neural models and can be a step towards more interpretable and explainable AI.

However, there is a limitation when working with subspaces. Most traditional machine learning algorithms require single vector representation on a Euclidean Space. However, subspaces exist on a Riemannian manifold called Grassmannian, and as such, subspace processing should be performed on this space. Previous works in NLP have worked around this problem by forcing the vectorization of the word subspace representation.

Taking such an approach is counter-productive since this vectorization process disregards all the interesting properties of the subspace mentioned above. While it is understandable the motivation of working with traditional machine learning algorithms, we cannot ignore the already established theory on subspace-based methods. The subspace-based methods operate with the subspace representation while preserving all the characteristics of this model.

To our knowledge, no work handles the subspaces generated by PCA from the word embeddings in a text and applies compatible subspace-based methods to the tasks proposed in this thesis. Although Mu et al. [23] used compatible subspace similarity measurement to perform semantic textual similarity, their approach consists of modeling each sentence as a subspace, and comparison was performed on a sentence level. Throughout this thesis, we provide additional insights on the subspace representation and demonstrate through our experiments how modeling word subspaces at different levels (e.g., sentence, text, and sets of texts) can help improve the results on text classification.

1.3 Objectives

This thesis aims to provide a solid foundation for applying the subspace representation to word embeddings through the word subspace model. While the subspace model has been extensively applied in the computer vision field, where subspace compatible methods have been developed, most of the work in NLP undermines the capabilities of this representation.

In this thesis, our main goal is to define the concept of the word subspace and, based on the established theory of subspace-based methods, develop tools to better understand this model from the NLP perspective. Moreover, we seek to apply the subspace representation to solve different NLP tasks, using the knowledge of word embeddings geometry and subspace theory to guide the decision of the subspace-based method.

Furthermore, we mainly focus on the word subspaces generated from non-contextualized word embeddings. While we do not go into much depth, we also demonstrate through our experiments the efficacy in modeling sets of contextualized word embeddings to solve sentiment analysis.

1.4 Contributions

The main contributions of this thesis are:

- The introduction of the concept of word subspace, which is efficient to represent natural language data based on the word embeddings.
- A simple but powerful tool called the word importance score, which allows us to interpret the basis vectors of a word subspace and the canonical vectors computed during subspace comparison.
- Empirical demonstration of the mathematical property of the subspace uniqueness on the word vector space, which shows that different sets of word vectors (i.e., different texts) belonging to the same topic span the same topic subspace.
- The incorporation of the uniqueness property to solve topic classification, which supports modeling a word subspace from sentences, texts and sets of texts.
- The geometric interpretation of the word embeddings to solve the sentiment analysis task.
- The proposal of a multi-modal framework for multimedia generation based on the word subspace model.

1.5 Thesis organization

The remainder of this thesis is organized as follows:

- **Chapter 2:** Word Subspace

  This chapter presents the word subspace model and lays the theoretical foundation behind it, explaining how we can model a word subspace from a set of word vectors and compare two

sentences or texts by using the subspace similarity. We also show how it is possible to extract the semantic difference between two sentences or texts based on the difference between their subspaces.

- **Chapter 3:** Text Analysis based on Word Subspace

  In this chapter, we propose a simple but powerful tool called the word importance score, which allows us to interpret the basis vectors of a word subspace and the canonical vectors computed during subspace comparison. We demonstrate how we can use this score by analyzing toy data.

- **Chapter 4:** Topic Classification

  This chapter discusses the application of the word subspace concept in topic classification. We focus on two different sub-tasks: Topic classification and sentiment analysis. First, we explore the geometry behind the word embeddings to perform a guided decision on the type of subspace-based method that is more suitable to solve each task. Then, based on the uniqueness property, which shows that different texts from the same topic generate almost the same word subspace, we propose solving topic classification by using the mutual subspace method (MSM) [45]. We demonstrate how incorporating the uniqueness property into the MSM leads better results through our experimental evaluation and use the tools proposed in Chapter 3 to interpret our results. For sentiment analysis, we explore the geometry behind the word vectors and show how we can compensate for the lack of sentiment information in these embeddings by using methods such as the orthogonal mutual subspace-based method (OMSM) and the Grassmann variations of MSM and OMSM, such as the Grassmann subspace method (GSM) and the Grassmann orthogonal subspace method (GOSM).

- **Chapter 5:** Multimedia generation

  In this chapter, we explore how the word subspace concept can be used as a powerful tool to compare different types of media, such as image and text, in a meme generation problem. We propose *news2meme*, a method for automatically generating memes from a news article, where we aim to match texts and images efficiently. We approach this task as two multimedia retrieval problems with the same input news text: 1) An image retrieval task where the output is a meme image; 2) A text retrieval task where the output is a catchphrase. These two outputs are combined to generate the meme for the news article. First, we represent texts and catchphrases as sets of word vectors through the *word2vec* representation. Then, to handle images similarly, we extract sets of tags from the images using a deep neural network. These tags are then translated to word vectors in the same vector space through *word2vec*. Finally, we represent the intrinsic variability of features in a set of word vectors with a word subspace. Under this framework, we can directly compare images and texts, making retrieval across media formats possible. Preliminary experiments were performed to evaluate our framework.

- **Chapter 6:** Conclusions

  Finally, in this chapter we present our conclusions and discuss several directions to be explored as future work.

Chapter 2

Word Subspace

In this chapter, we present the concept of the word subspace. We start by giving its definition and explain how a word subspace can be modeled from a sentence, a text, or a set of texts. We then explain how two word subspaces can be compared in terms of subspace similarity and the difference between subspaces. We finally explore one essential characteristic of the subspace representation, the subspace uniqueness property, and investigate if it holds for subspaces modeled from word vectors.

2.1 Definition of Word Subspace

In this formulation, words are represented as vectors in a real-valued feature vector space $\mathbb{R}^p$ generated by a word embedding model. Naturally, a sentence, text, or a set of texts can be seen as a set of word vectors, which can be modeled as a set of basis vectors spanning a linear subspace. To obtain a consistent subspace representation, which can be applied to sets of different numbers of word vectors, the principal components analysis (PCA) without data-centering is applied to the set of word vectors.

Through this representation, most of the variability of the word vectors in the set is retained, and, consequently, a word subspace can effectively and compactly represent the context of the corresponding sentence, text, or set of texts. Each direction given by the basis vectors of the word subspace represents the directions with the highest variance of the text in the embedding space and can be regarded as the main semantic meanings [26], or main hidden topics.

Figure 2.1 shows how to model a word subspace from a text. Consider a text d to be a set of N words, $d = \{w_k\}_{k=1}^{N}$. Each of these words are translated into a word vector by using a word embedding model of our choice, resulting in a set of word vectors $X_d = \{\boldsymbol{x}_k\}_{k=1}^{N}$. The word vectors in this set can then be stacked into a matrix $\boldsymbol{X}_d \in \mathbb{R}^{p \times N}$, where p is the dimension of the word vector space.

To model the word subspace for this set of word vectors, we first need to compute the following autocorrelation matrix, $\boldsymbol{R}_d$:

$$\boldsymbol{R}_d = \boldsymbol{X}_d \boldsymbol{X}_d^{\top} \tag{2.1}$$

The orthonormal basis vectors of the m-dimensional subspace $\mathcal{Y}_d$ are obtained as the eigenvectors

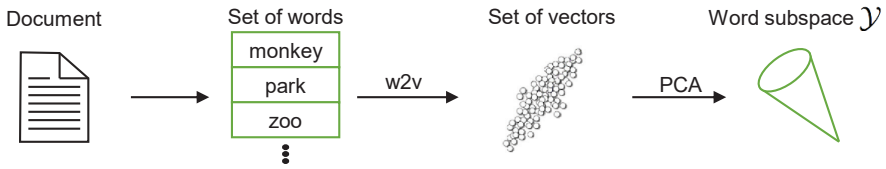


Figure 2.1: Process of modeling a text as a word subspace. Words from the text are extracted and then translated to word vectors by using a word embedding model. Then, the set of word vectors is modeled as a word subspace by using PCA

with the m largest eigenvalues $\{\lambda_l\}_{l=1}^{m}$ of the matrix $\boldsymbol{R}_d$. The subspace $\mathcal{Y}_d$ is then represented by the matrix $\boldsymbol{Y}_d \in \mathbb{R}^{p\times m}$, which has the corresponding orthonormal basis vectors as its column vectors.

Analogously, it is possible to model a word subspace from a set of documents $D = \{d_i\}_{i=1}^{|D|}$. In this case, we are modeling the distribution of the words in all of the documents in the set. To model it, we follow the same process described above. The main difference is the word vector matrix $\boldsymbol{X}_D$, which will stack all the word vectors from all the documents in the set.

In general, the dimension m of each subspace is empirically determined. For sentences, about four dimensions should suffice to retain most of the sentence's variance [23]. However, for texts or sets of texts, more dimensions will likely be necessary. The amount of variance retained by the basis vectors of the subspace can be determined by using the cumulative contribution rate $\mu(m)$. Considering that we want to keep a minimum of μ_{min} of the text variance, we can determine m by ensuring that $\mu(m)_d \geq \mu_{min}$, where:

$$\mu(m)_d = \frac{\sum_{l=1}^{m}(\lambda_l)}{\sum_{l=1}^{p}(\lambda_l)}. \tag{2.2}$$

2.2 Comparison between subspaces

In this section, we discuss how to compare two sentences or texts based on their word subspace representation. We present two different ways to compare them: Based on the subspace similarity and based on the difference subspace.

2.2.1 Subspace similarity

To measure the similarity between two word subspaces $\mathcal{Y}_1$ and $\mathcal{Y}_2$, the canonical angles between the two word subspaces are used [46]. There are several methods for calculating canonical angles [47], [48], and [34], but the simplest and most practical is the singular value decomposition (SVD). Consider two subspaces represented as matrices of bases, $\boldsymbol{Y}_1 = [\boldsymbol{\Phi}_1 \dots \boldsymbol{\Phi}_{m_1}] \in \mathbb{R}^{p\times m_1}$ and $\boldsymbol{Y}_2 = [\boldsymbol{\Psi}_1 \dots \boldsymbol{\Psi}_{m_2}] \in \mathbb{R}^{p\times m_2}$, where $\boldsymbol{\Phi}_i$ are the bases for $\mathcal{Y}_1$ and $\boldsymbol{\Psi}_i$ are the bases for $\mathcal{Y}_2$, and $m_1 \geq m_2$. Let the SVD of $\boldsymbol{Y}_1^\top \boldsymbol{Y}_2 \in \mathbb{R}^{m_1\times m_2}$ be $\boldsymbol{Y}_1^\top \boldsymbol{Y}_2 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{\Sigma} = \text{diag}(\kappa_1, \dots, \kappa_{m_2})$, $\{\kappa_i\}_{i=1}^{m_2}$ represents the set of singular values. The canonical angles $\{\theta_i\}_{i=1}^{m_2}$ can be obtained as $\{\cos^{-1}(\kappa_1), \dots, \cos^{-1}(\kappa_{m_2})\}$ $(\kappa_1 \geq \dots \geq \kappa_{m_2})$. The similarity between the two subspaces is measured by t angles as follows:

Table 2.1: Document distribution over the classes of the Reuters-8 dataset

Index	Class	Number of samples
1	acq	2292
2	crude	374
3	earn	3923
4	grain	51
5	interest	271
6	money-fx	293
7	ship	144
8	trade	326

$$S_{(\boldsymbol{Y}_1,\boldsymbol{Y}_2)}[t] = \frac{1}{t}\sum_{i=1}^{t}\cos^2\theta_i,\ 1 \leq t \leq m_2,\ m_2 \leq m_1. \tag{2.3}$$

2.2.2 Difference Subspace

To understand what are the different topics between two texts, we can use the concept of difference subspace (DS) [45]. The DS is a natural extension of the difference vector concept to a pair of subspaces. By projecting a vector or a subspace onto a DS, we can extract different components, that is, different topics between two word subspaces.

Consider two subspaces $\mathcal{Y}_1$ and $\mathcal{Y}_2$, represented as matrices of bases, $\boldsymbol{Y}_1 = [\boldsymbol{\Phi}_1 \dots \boldsymbol{\Phi}_{m_1}] \in \mathbb{R}^{p\times m_1}$ and $\boldsymbol{Y}_2 = [\boldsymbol{\Psi}_1 \dots \boldsymbol{\Psi}_{m_2}] \in \mathbb{R}^{p\times m_2}$, where $\boldsymbol{\Phi}_i$ are the bases for $\mathcal{Y}_1$ and $\boldsymbol{\Psi}_i$ are the bases for $\mathcal{Y}_2$. To compute the difference subspace $\tilde{\mathcal{D}}_2$ between subspaces $\mathcal{Y}_1$ and $\mathcal{Y}_2$, we first need to calculate the following projection matrices $\boldsymbol{P} = \boldsymbol{Y}_1\boldsymbol{Y}_1^\top$ and $\boldsymbol{Q} = \boldsymbol{Y}_1\boldsymbol{Y}_2^\top$. Then, we perform the eigenvalue decomposition of the matrix $\boldsymbol{P} + \boldsymbol{Q}$. The subspace spanned by the eigenvectors of $\boldsymbol{P} + \boldsymbol{Q}$ with eigenvalues larger than 0 and smaller than 1 span the difference subspace $\tilde{\mathcal{D}}_2$.

2.3 Word subspace uniqueness

To better understand how we can use the word subspace along with consolidated subspace-based method to solve different tasks in NLP, we analyze a basic mathematical property of the subspace representation. One attractive characteristic of the subspace representation is that, while it is a unique entity in a high-dimensional vector space, it can be spanned by different sets of basis vectors. This section seeks to understand if different sets of word vectors from the same topic can generate the same subspace by observing how the subspace similarities change when different sets of words are modeled.

For this purpose, we analyzed the word subspace generated from the Reuters-8 dataset classes without stop words [49]. We considered words in the texts as they appeared without performing stemming or typo correction. This database has eight different classes, where the number of samples varies from 51 to over 3000 documents, as can be seen in Table 2.1.

To obtain the vector representation of words, we used a pre-trained model for *word2vec*[1] [1]. For this analysis, we first chose class 1 and class 3 and defined all subspaces to have a dimension of 50.

We first observe how the similarities behave. Consider the texts in class 1 for the standard train set, D_1. We randomly divided them into two subsets of 798 texts, namely D_1^1 and D_1^2, so that there is no overlap between them. Then, for each of the subsets, we obtained the word vectors corresponding to their respective words, resulting in the word vector sets X_1^1 and X_1^2. We modeled them as word subspaces, represented by the basis vectors $\boldsymbol{Y}_1^1$ and $\boldsymbol{Y}_1^2$, and calculated the similarity between them:

$$S_{(\boldsymbol{Y}_1^1, \boldsymbol{Y}_1^2)} = 0.97,$$

which indicates that their subspaces are very close to each other.

Then, we compared $\boldsymbol{Y}_1^1$ and $\boldsymbol{Y}_1^2$ with $\boldsymbol{Y}_3$, the basis vectors of the word subspace generated by the word vectors in class 3 from train set, X_3. We obtained the following similarities:

$$S_{(\boldsymbol{Y}_1^1, \boldsymbol{Y}_3)} = 0.67,$$
$$S_{(\boldsymbol{Y}_1^2, \boldsymbol{Y}_3)} = 0.68.$$

Since the similarity between $\boldsymbol{Y}_1^1$ and $\boldsymbol{Y}_1^2$ is almost 1 and their similarities with $\boldsymbol{Y}_3$ are much lower, we can see that both subsets are very close to each other while being further apart from class 3.

Finally, we compared the word subspace $\boldsymbol{Y}_1$ generated from X_1, with the word subspace $\boldsymbol{Y}_3$, and obtained:

$$S_{(\boldsymbol{Y}_1, \boldsymbol{Y}_3)} = 0.68,$$

which is about the same as when comparing the subsets of class 1 with class 3.

Now, let us also look at the texts from classes 1 and 3 in the test set. If we model the word subspace from the texts of class 1, $\boldsymbol{Z}_1$, and compare it with the subsets from class 1 of the train set, we obtain:

$$S_{(\boldsymbol{Z}_1, \boldsymbol{Y}_1^1)} = 0.96,$$
$$S_{(\boldsymbol{Z}_1, \boldsymbol{Y}_1^2)} = 0.97,$$

which is about the same when $\boldsymbol{Y}_1^1$ and $\boldsymbol{Y}_1^2$ were compared to each other. This shows that $\boldsymbol{Z}_1$, $\boldsymbol{Y}_1^1$ and $\boldsymbol{Y}_1^2$ are very close to each other.

If we compare $\boldsymbol{Z}_1$ with $\boldsymbol{Y}_3$, we obtain:

$$S_{(\boldsymbol{Z}_1, \boldsymbol{Y}_3)} = 0.68,$$

which is almost the same as the similarity between $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_3$. Finally, to see if $\boldsymbol{Z}_1$ corresponds to $\boldsymbol{Y}_1$, we can take their similarity:

$$S_{(\boldsymbol{Z}_1, \boldsymbol{Y}_1)} = 0.96,$$

which indicates they are almost the same.

[1] https://code.google.com/archive/p/word2vec/

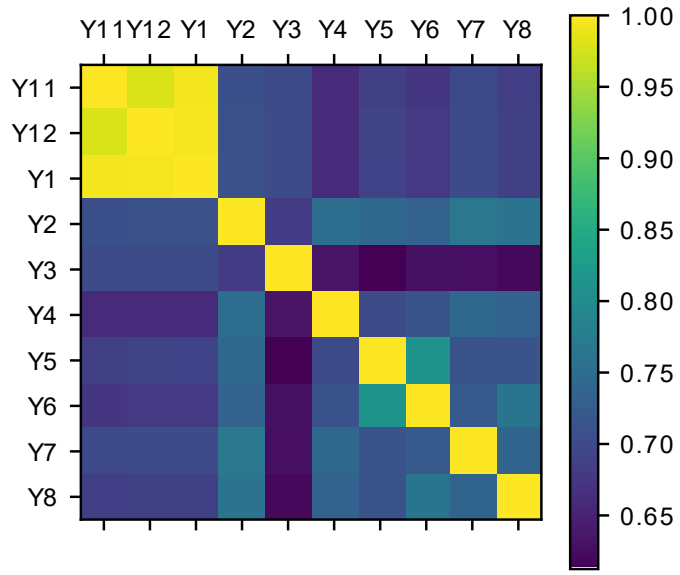


Figure 2.2: Similarity matrix between the subspaces of the classes in the train set of the Reuters-8 dataset and the subspaces $\boldsymbol{Y}_1^1$, $\boldsymbol{Y}_1^2$ and $\boldsymbol{Y}_1$. $\boldsymbol{Y}_1$ was modeled from all texts in the train set of class 1, represented by $\boldsymbol{X}_1$; $\boldsymbol{Y}_1^2$ and $\boldsymbol{Y}_1$ were modeled from mutually exclusive subsets of $\boldsymbol{X}_1$

.

On the other hand, if we compare $\boldsymbol{Z}_3$, the word subspace generated from texts of class 3 in the test set, and $\boldsymbol{Y}_3$, we obtain:

$$S_{(\boldsymbol{Z}_3,\boldsymbol{Y}_3)} = 0.92.$$

Therefore, we can understand that, if texts belong to the same semantic topic, we can obtain almost the same word subspace by considering different subsets of it.

To better understand these relationships, we can check how the subspace of each class in the train set compares to the subspaces $\boldsymbol{Y}_1^1$, $\boldsymbol{Y}_1^2$ and $\boldsymbol{Y}_1$. These similarities are represented in Fig. 2.2:

We can see that the subsets of class 1 have a high similarity between them. In contrast, they have a much lower similarity with the other classes. We can also see that the two subsets compare to the other classes very similarly to how class 1 itself compares to the other classes. This result is a strong indication that the subspaces $\boldsymbol{Y}_1^1$, $\boldsymbol{Y}_1^2$, and $\boldsymbol{Y}_1$ are almost the same, despite them being generated from different texts.

Chapter 3

Text Analysis based on Word Subspace

In this chapter, we present how to perform text analysis based on the word subspace representation. We start by introducing the Word Importance Score, which allows us to interpret the basis vectors of a word subspace. Then, we demonstrate how the word subspace and the word importance score can be used to perform text analysis on toy data.

3.1 Word Importance Score

The basis vectors of a word subspace represent the most important directions in terms of variance of the text word vectors and can be interpreted as the main hidden topics in the text. Despite the word vector space being a real-valued space, the word vectors represent discrete data, i.e., words, and therefore it is very likely that no known word vectors correspond to the basis vectors. However, we can understand which words are more important to represent the hidden topics based on the word importance score, defined as the cosine similarity between the word vectors from the text and its basis vectors.

Let $\boldsymbol{X} = [\boldsymbol{x}_1 \dots \boldsymbol{x}_N]$ be an embedding matrix, where each column is the word vector for each word in the text $d = \{w_k\}_{k=1}^{N}$, and $\boldsymbol{Y} = [\boldsymbol{\Phi_1} \dots \boldsymbol{\Phi}_m]$ be the basis vectors of the word subspace $\mathcal{Y}$ generated from $\boldsymbol{X}$. We consider all word vectors were normalized to have norm equal to 1. To calculate the word importance score of w_k with respect to basis vector $\boldsymbol{\Phi}_k$, we use the following equation:

$$I(\boldsymbol{x}_k, \boldsymbol{\Phi}_i) = \boldsymbol{x}_k{}^\top \boldsymbol{\Phi}_i. \tag{3.1}$$

The intuition behind this score is that the closest the word vector is to the basis vector of the word subspace, the more relevant this word is with regard to the hidden topic represented by the basis vector. Therefore, it is possible to understand these hidden topics by observing the closest known word vectors to them.

Furthermore, we can understand which words in the text are the most important by measuring the projection of its word vectors onto its subspace. For example, the word importance score of word w_k with respect to the word subspace $\mathcal{Y}$ is defined by the following equation:

$$I(\boldsymbol{x}_k, \boldsymbol{Y}) = \boldsymbol{x}_k{}^\top \boldsymbol{P} \boldsymbol{x}_k, \tag{3.2}$$

where $\boldsymbol{P} = \boldsymbol{Y}\boldsymbol{Y}^\top$ is the projection matrix of the subspace $\mathcal{Y}$.

Despite its simplicity, the word importance score is a powerful metric that can help us quickly understand essential information about the texts (i.e., main topics) and interpret how two texts are related to each other in terms of similarity.

Let $\boldsymbol{Y}_a = [\boldsymbol{\Phi}_1 \dots \boldsymbol{\Phi}_{m_a}] \in \mathbb{R}^{p \times m_a}$ and $\boldsymbol{Y}_b = [\boldsymbol{\Psi}_1 \dots \boldsymbol{\Psi}_{m_b}] \in \mathbb{R}^{p \times m_b}$ be the basis vectors matrices of the word subspaces $\mathcal{Y}_a$ and $\mathcal{Y}_b$, generated from the embedding matrices $\boldsymbol{X}_a = [\boldsymbol{x}_a^1 \dots \boldsymbol{x}_a^{N_a}]$ and $\boldsymbol{X}_b = [\boldsymbol{x}_b^1 \dots \boldsymbol{x}_b^{N_b}]$, respectively. To understand which word vectors from $\boldsymbol{X}_a$ and $\boldsymbol{X}_b$ were more relevant to the comparison between $\mathcal{Y}_a$ and $\mathcal{Y}_b$, we calculate their importance with respect to the canonical vectors pairs $\boldsymbol{p}_i, \boldsymbol{q}_i (i = 1, ..., t)$:

$$I(\boldsymbol{x}_a^k, \boldsymbol{p}_i) = \boldsymbol{x}_a^{k\top} \boldsymbol{p}_i \tag{3.3}$$

$$I(\boldsymbol{x}_b^k, \boldsymbol{q}_b) = \boldsymbol{x}_b^{k\top} \boldsymbol{q}_i \tag{3.4}$$

Based on these scores, we can see how two texts relate to each other in terms of subspace similarity and understand, for example, why a given input text was assigned to a determined class in a topic classification problem. However, if we seek to understand what makes two texts different, analysis based on the canonical vectors might not be effective if the texts have some similarities while having different points. In this case, we can use the importance score with regards to the basis vector of their difference subspace.

Let $\boldsymbol{D} = [\boldsymbol{\Upsilon}_1 \dots \boldsymbol{\Upsilon}_n]$ be the basis vectors of the difference subspace between $\mathcal{Y}_1$ and $\mathcal{Y}_2$. Consider $\{\boldsymbol{x}_j\}_{j=1}^N$ to be the set of word vectors from both texts. To measure the importance score of the word w_j with respect the difference subspace basis vector $\boldsymbol{\Upsilon}_k$, we use the following equation:

$$I(\boldsymbol{x}_j, \boldsymbol{\Upsilon}_k) = \boldsymbol{x}_j{}^\top \boldsymbol{\Upsilon}_k \tag{3.5}$$

3.2 Analysis of Toy Data

This section demonstrates how we can use the word importance score to analyze texts. We first show how we can easily understand the topics inside a text based on its word subspace. Then, we analyze how two texts relate to each other based on their similarity and difference subspace.

We selected three different texts for this demonstration: Text1 and Text2 talk about the same event, i.e., protests in early 2019 in Brazil due to the cuts in the public funding on education. Text3 talks about protests in 2014 against the World Cup in Brazil. We specifically chose them such that there is a common topic among them, i.e., they all talk about protests, while one differs in terms of their motivation. The full texts can be seen in Appendix A.

Then, we modeled a word subspace for each one of them. For this analysis, we pre-processed the texts by removing the stop words and defined all word subspaces to have 40 dimensions.

3.2.1 Word subspace representation

We start by analyzing the word subspaces generated from each of the texts by using the word importance score described in Equation 3.2. Then, for easier visualization, we plotted a word cloud for each text, where the bigger and darker the word is, the higher is the importance score. They can be seen in Figure 3.2.

For Text1, we can see that words such as “protest” and “thousands” are the most important, followed by words such as “Thursday”, “government”, “student”, and “demonstration”. We can

(a) Text1. (b) Text2. (c) Text3.

Figure 3.1: Word cloud representing the word importance score of the words in each text, according to their word subspaces.

see that for Text2, more attention is given to words related to education, such as “Education” and “student”, but words associated with the protests, such as “protesters” are also highlighted. Finally, for Text3, we can see that words related to the World Cup and the strikes are considered more important. They are all consistent with our prior knowledge about the texts, which shows that the basis vectors of the word subspace are aligned with the main topics in the text.

Furthermore, it is possible to visualize these words per basis vector by using Equation 3.1, where words highlighted based on the first vector are more important than words highlighted by the second basis vector and so on.

Figure 3.2 shows the words for the Text1, according to the first four basis vectors of its subspace. Here we have a clearer understanding of the main hidden topics of the text. We can see that words such as “protests”, “government”, and “education” are considered to be more important by the first basis vector. Interestingly, the first basis vector of a subspace modeled from a set of vectors by PCA without data centering is almost the same as the average of the vectors. This result might justify why taking the average of word vectors works well on many different NLP tasks.

However, the words in a text can represent different aspects of its content. For the first basis vector, we could grasp a general idea of what is happening in the text. However, if we look at the most important words according to the second basis vector, we see different words being highlighted. We can see that the second and third basis vector highlighted a different topic included in the first basis vector analysis. The second basis vector gave more importance to the education topic, highlighting words such as “education”, “students”, and “teachers”, whereas the third basis vector gave more importance to the protests, highlighting words such as “protests” and “demonstrations”.

If we extend this analysis to the fourth basis vector, we can see another highlighted aspect of the text, where the fourth basis vector mainly highlighted the name of cities, such as “Sao Paulo”, “Rio de Janeiro” and “Brasilia”, and “Brazil” itself.

By observing how the word subspace highlights the words in each of its bases, we can observe a factorization of the main topics in the text. For example, while the first basis vector was capable of capturing words related to the main topic, i.e., the protests against cuts in education in Brazil, the following basis vectors could separate it into what was happening (protests), what was it related to (education), and where did it happen (cities in Brazil). Such analysis is strong evidence that considering the whole distribution of the words in the text through the subspace representation can lead to a richer representation of the contents of the text.

(a) First basis vector.

(b) Second basis vector.

(c) Third basis vector.

(d) Fourth basis vector.

Figure 3.2: Word clouds representing the word importance score of the words in text1, according to the first four basis vector of its word subspace.

(a) Important words in Text1.

(b) Important words in Text2.

Figure 3.3: Word clouds representing the word importance score of the words in the texts about the same event, i.e., Text1 and Text2, based on the first canonical pair between their word subspaces.

(a) Important words in Text1.

(b) Important words in Text3.

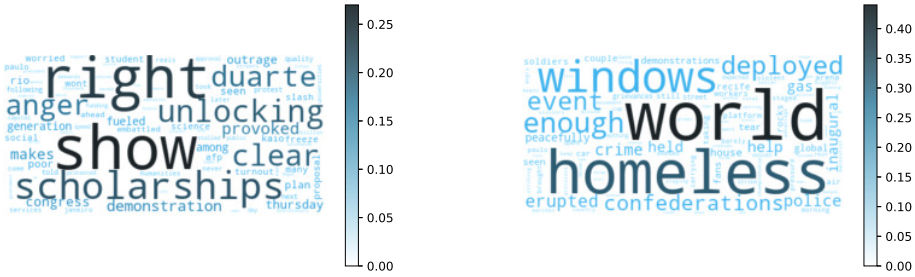
Figure 3.4: Word clouds representing the word importance score of the words in the texts that talk about different protests, i.e., Text1 and Text3, based on the first canonical pair between their word subspaces.

3.2.2 Comparison between texts - Canonical Vectors

In this section, we analyze which words are more important when comparing two texts by using Equations 3.3 and 3.4. Based on these equations, it is possible to understand which words are more important when comparing their respective subspaces, which can serve as an essential tool for understanding the results in a classification task, for example.

We first analyze how two texts talking about the same event relate to each other. Figure 3.3 shows the most important words with respect to the first pair of canonical vectors between Text1 and Text2. We can see words such as “students”, “teachers”,“thousands” highlighted in both texts. We also see words such as “protests” highlighted with lower importance scores. This result is coherent as both texts talk about protests made by students and professors in favor of education. At the same time, it reveals that what mostly connects them is the topic “education”.

Next, we perform the same analysis on two texts that talk about different protests. Figure 3.4



(a) Important words in Text1.

(b) Important words in Text3.

Figure 3.5: Word clouds representing the word importance score of the words in the texts with respect to the first basis vector of the difference subspace between Text1 and Text3.

shows the important words with respect to the first pair of canonical vectors between Text1 and Text3. Words such as “protest”, “demonstration”, and “thousands” were highlighted in Text1 and words such as “streets”, “protests” and “demonstrators” were highlighted in Text3. In this case, no words related to the motivation of the protests were highlighted, as they are different for each text.

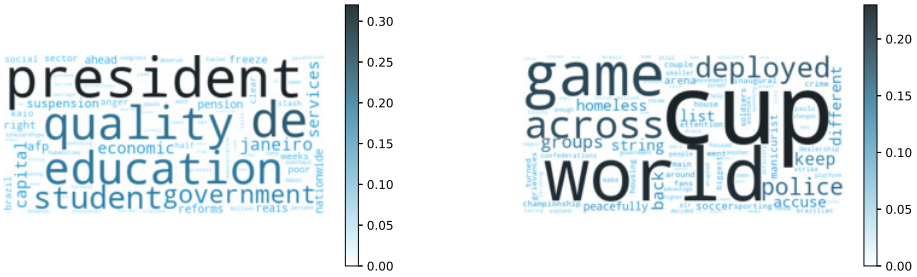
3.2.3 Comparison between texts - Difference Subspace

So far, we have focused on understanding how texts are related when comparison based on canonical vectors is performed. We could see that when texts talk about common subjects, the canonical vectors can connect them by these subjects. However, in cases where texts have some similar and different points, the analysis based only on the canonical angles might not be sufficient.

Consider Text2 and Text3. Both talk about protests happening in Brazil; however, one of them discusses the protests against education funding cuts, while the other talks about the protests against the World Cup. To understand what are the differences between these texts, we modeled the difference subspace from the word subspaces of Text1 and Text3 as described in Section 2.2.2. Then, we calculated the word importance of the words in both texts with respect to the basis vectors of this difference subspace, following Equation 3.5.

Figure 3.5 shows the most important words that differ Text1 and Text3 based on the first basis vector of their difference subspace. We can see that words such as “right”, “show”, and “scholarships” were highlighted for Text1, while words such as “world”, “homeless”, and “windows” highlighted for Text3. As expected, we have words related to education and the world cup being considered more important when looking for their differences. However, we also see words that, when isolated, do not seem to connect to the content of the texts.

Figure 3.6 shows the most important words that differ Text1 and Text3 based on the second basis vector of their difference subspace. By observing the most important words according to the second basis vector of their difference subspace, we can better understand what they mean in the context of the texts. For example, we see the word “president” highlighted in Text1. Therefore, it is likely that the word “right”, highlighted in the first text based on the first basis vector of the DS, is being used as political meaning. Furthermore, more words related to education are highlighted, such as



(a) Important words in Text1.

(b) Important words in Text3.

Figure 3.6: Word clouds representing the word importance score of the words in the texts with respect to the second basis vector of the difference subspace between Text1 and Text3.

"education", whereas words related to the world cup are highlighted for Text3.

3.3 Summary

This chapter presented a simple but powerful tool to perform text analysis based on the word subspace representation, the word importance score. It measures how important a word vector is with regards to a word subspace basis vector, canonical vectors, or the basis vector of the difference subspace taken from two texts. The importance with respect to each of these vectors can reveal important information about a text, such as the main hidden topics in a particular text or set of texts and the topics that relate two texts based on their subspace similarity or difference subspace. We demonstrated how to perform such analysis by using toy data and, throughout the remainder of this thesis, we use these tools to interpret the results given by the word subspace model along with the subspace-based methods.

Chapter 4

Word Subspace for Text Classification

This chapter proposes a novel framework for text classification based on subspace-based methods. We specifically cover two sub-tasks within text classification: topic classification and sentiment analysis. We explore the geometry behind the word embeddings to make a guided decision on the subspace-based method that is more suitable for solving each task.

Text classification aims to classify different texts into a fixed number of predefined categories, helping to organize data and making it easier for users to find the desired information. In the past years, many methods based on machine learning and statistical models have been applied to perform this task, such as latent semantic analysis (LSA), support vector machines (SVM), and multinomial naive Bayes (MNB). Recently, several non-parametric models based on word embeddings have been proposed to solve these tasks using different spectral methods [29], [31], [30], [32].

To our knowledge, no work handles the subspaces generated from the word embeddings in a text and applies compatible subspace-based methods to solve text classification, as proposed in this thesis. Although Mu et al. [23] utilized compatible subspace similarity measurement to perform semantic textual similarity, they took a naive approach by modeling word subspaces at a sentence level, which they demonstrated to be adequate to solve semantic textual similarity tasks.

However, such a naive approach might not be the best suited for text classification. Since most word embeddings are trained based on the words' co-occurrence, the generated word vectors might not necessarily discriminate towards a given classification task. Hence, it is essential to find a subspace-based method that better extracts the discriminative features necessary for each task. Under such methods, we model word subspaces at the sentence level and at the text level. More importantly, in this chapter, we demonstrate how the understanding of the subspace uniqueness property discussed in Chapter 2 supports the word subspace modeling at these different levels.

We show the validity of the proposed frameworks through experiments on four different datasets, where two of them focus on topic classification, and two focus on sentiment analysis. We demonstrate the effectiveness of the word subspace representation and the subspace-based classification methods by comparing them with the performance of recent text models.

The remainder of this chapter is organized as follows. We dedicate Section 4.1 for Topic Classification and Section 4.2. In each of them, we describe how we chose the most appropriate subspace-based method and demonstrate their performance through experimental evaluation. Finally, a summary is presented in Section 4.3.

4.1 Topic Classification

The goal in topic classification is to assign an input text to a previously modeled class of the same main topic. This main topic can be characterized by a set of word vectors $\tilde{X}_c = \{\tilde{\boldsymbol{x}}_c\}$ that correspond to words that highly co-occur in such a context. In general, this specific set of words might not be available; however, we hypothesize that the word's distribution from all texts in a class is a reasonable model of the main topic in this class.

Based on the findings in Section 2.3, we conjuncture that there should exist a word subspace spanned by these important words to represent the topic class, i.e., topic class subspace, which we can derive from a single text from the class or a combination of all texts in the class, i.e., $span(\tilde{X}_c) \approx span(X_c)$. On top of that, a word subspace modeled from a text, i.e., text subspace, that also belongs to this topic class should be almost the same as the topic class subspace, i.e., $span(X_c^i) \approx span(\tilde{X}_c)$.

Under these assumptions, the classification of an input text can be performed by comparing its text subspace with the topic class subspaces. This process can be performed under the framework of the mutual subspace method (MSM) [45], where words from texts of the same class are assumed to belong to the same context.

4.1.1 Proposed framework

Our proposed framework has two different states. In the training stage, we model one word subspace from the set of training documents of each class, following the procedure presented in Section 2. This process results in the topic class subspaces $\mathcal{Y}_c$. Since the number of words in each class may vary largely, the dimension m_c of each topic class subspace can be set to different values accordingly.

In the classification stage, for an input document d_q, we model a text subspace $\mathcal{Y}_q$ and compare it in terms of subspace similarity with the topic class subspaces. This framework can be seen in Fig. 4.1.

Finally, the class with the highest similarity with d_q is assigned as the class of d_q:

$$prediction(d_q) = argmax_c(S_{(\boldsymbol{Y}_c, \boldsymbol{Y}_q)}). \tag{4.1}$$

Fig. 4.2 shows the modeling and comparison of sets of words by MSM. This method can compare different-sized sets and naturally encode proximity between sets with related words.

It is important to note that the main difference between the approaches taken by Mu et al. [23] to solve the semantic textual similarity (STS) task is that, in their method, the word subspaces were modeled at a sentence level, and comparison was also performed between two sentences (i.e., same natural language structure level). In this work, we propose representing each class as a single word subspace, modeled from all texts in the class, and classifying the input text by comparing it the set of texts in each training class (i.e., comparison between different levels). If we were to keep the same strategy, each topic class would be represented by several text subspaces. To classify an input text, we would need to compare its text subspace to all of the reference text subspaces and assign the label of the reference text it is the closest to. We will refer to this framework as 1NN-MSM. Our experiments demonstrate that while this approach is valid, it might not be the most appropriate to solve topic classification.

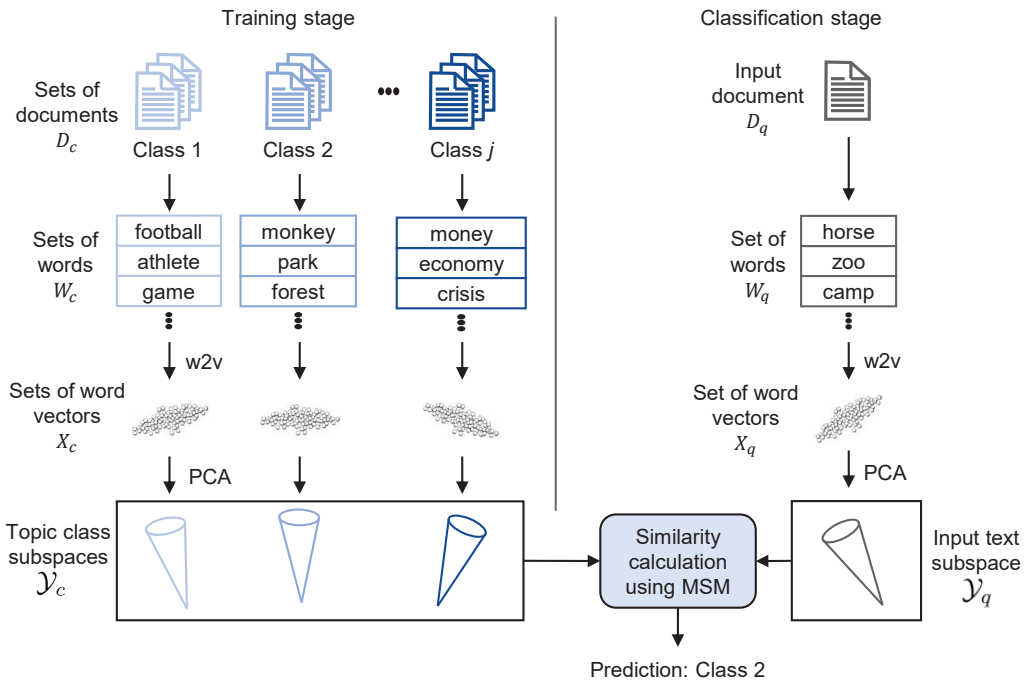


Figure 4.1: Text classification based on word subspace, under the MSM framework

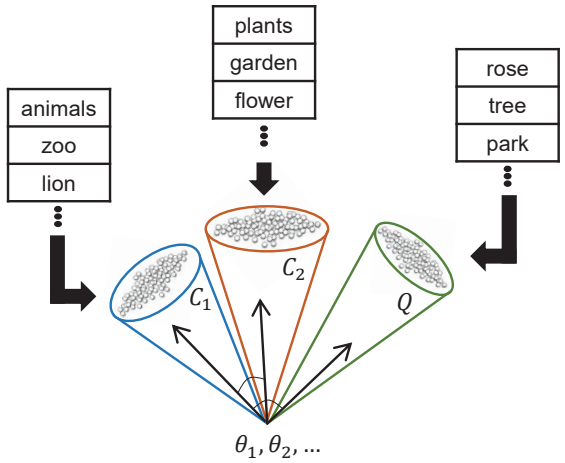


Figure 4.2: Comparison of sets of word vectors by the mutual subspace method

4.1.2 Topic class subspace modeling by weighted PCA

The word subspace formulation presented in Chapter 2 is a practical and compact way to represent sets of word vectors, retaining most of the variability of features. However, when modeling a topic class subspace, the number of word vectors in $X_c = \{\boldsymbol{x}_c^k\}_{k=1}^{N_c}$ can be large, leading to large memory

consumption. Therefore, we propose a reformulation to the word subspace modeling to reduce the memory load while achieving the same results.

In this formulation, instead of considering every instance of the word vectors in the class, we consider each unique word vector and the number of times it occurs (i.e., term-frequency). Then, we apply a weighted version of the PCA [50], [51].

Consider the set of word vectors $\{\boldsymbol{x}_k\}_{k=1}^{N_u} \in \mathbb{R}^p$, which represents each unique word in a class, and the set of weights $\{\omega_i\}_{i=1}^{N_u}$, which represents the frequencies of the words in the class. We combine these frequencies into the subspace modeling by weighting the word embedding matrix $\boldsymbol{X}$ as follows:

$$\widetilde{\boldsymbol{X}} = \boldsymbol{X}\boldsymbol{\Omega}^{1/2}, \tag{4.2}$$

where $\boldsymbol{X} \in \mathbb{R}^{p \times N_u}$ is a matrix containing the word vectors $\{\boldsymbol{x}_k\}_{k=1}^{N_u}$ and $\boldsymbol{\Omega}$ is a diagonal matrix containing the weights $\{\omega_i\}_{i=1}^{N_u}$.

We then perform PCA by solving the SVD of the matrix $\widetilde{\boldsymbol{X}}$:

$$\widetilde{\boldsymbol{X}} = \boldsymbol{A}\boldsymbol{M}\boldsymbol{B}^\top, \tag{4.3}$$

where the columns of the orthogonal matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are, respectively, the left-singular vectors and right-singular vectors of the matrix $\widetilde{\boldsymbol{X}}$, and the diagonal matrix $\boldsymbol{M}$ contains singular values of $\widetilde{\boldsymbol{X}}$.

Finally, the orthonormal basis vectors of the m_c-dimensional weighted subspace $\mathcal{W}$ are the column vectors in $\boldsymbol{A}$ corresponding to the m_c largest singular values in $\boldsymbol{M}$. This process is equivalent to modeling the word subspace as described in Chapter 2, but as we reduce the size of the embedding matrix by considering unique instances, it is possible to reduce memory and time consumption in the process of modeling the topic class subspace.

Besides the computational gain, modeling the word subspace through this formulation allows us to explore different weights for the words, such as binary weights, term-frequency inverse document-frequency weights, and others. The effects of using these different weights when modeling the word subspace are not in the scope of this thesis; however, in our experiments, we demonstrate how binary weights can be more effective, depending on the dataset.

4.1.3 Experimental Evaluation

This section describes the experiments performed to show the validity of our proposed frameworks for topic classification.

We performed four different experiments. We first compared our methods with conventional text classification methods, such as multinomial naive Bayes and latent semantic analysis. Then, we performed a simple analysis of the word importance score to investigate the validity of using the 1NN-MSM framework. We also report the computational time difference when modeling the reference classes for MSM using the generalized PCA. Then, we show how the word subspace representation and MSM compare with more recent text models.

We used two different datasets, **Reuters-8 dataset** from [49], and **The 20 newsgroups dataset**, also from [49]. To obtain the vector representation of words, we used a pre-trained model for

word2vec[1].

Comparison with Conventional Text Classification methods

We first tested the classification performance of MSM along with the word subspace representation (WSub) on the Reuters-8 dataset, using as word embedding the pre-trained word2vec model mentioned above. To understand how the frequency of the words influences the results, we also tested performing classification by considering only one occurrence of each word when modeling the word subspaces (u-WSub).

We considered the following conventional methods for comparison: multi-variate Bernoulli (MVB), multinomial naive Bayes (MNB) [52], latent semantic analysis (LSA) [19] and support vector machines (SVM). Since none of these methods work with vector set classification, we also compared a simple baseline for comparing sets of vectors, defined as the average of similarities between all vector pair combinations of two given sets. For two matrices A and B, containing the sets of vectors $\{\boldsymbol{x}_a^i\}_{i=1}^{N_A}$ and $\{\boldsymbol{x}_b^i\}_{i=1}^{N_B}$, respectively, where N_A and N_B are the number of words in each set, the similarity is defined as:

$$Sim_{(A,B)} = \frac{1}{N_A N_B} \sum_{i}^{N_A} \sum_{j}^{N_B} \boldsymbol{x}_a^{i\top} \boldsymbol{x}_b^j. \tag{4.4}$$

We refer to this baseline as similarity average (SA). We only considered one occurrence of each word in each set for this method.

Depending on the methods, we used different features. Classification with SA and MSM was performed using *word2vec* features, to which we refer as w2v. For MVB, due to its nature, only bag-of-words features with binary weights were used (binBOW). For the same reason, we only used bag-of-words features with term-frequency weights (tfBOW) with MNB. Classification with LSA is usually performed using bag-of-words features and, therefore, we tested with binBOW, tfBOW, and with the term-frequency inverse document-frequency weight, tfidfBOW. We also tested them using *word2vec* vectors. In this case, we considered each word vector from all documents in each class to be an individual sample. In addition to BOW features, we also assessed the performance of SVM using document representations generated with Latent Dirichlet Allocation [53], a topic modeling method that represents each document as a combination of a fixed number of topics.

To determine the dimensions of the class subspaces and query subspace in MSM, and the dimension of the approximation performed by LSA, we performed 10-fold cross-validation, wherein each fold, the data were randomly divided into a train (60%), a validation (20%) and a test set (20%). For LDA, we set the number of topics to be 50.

The results can be seen in Table 4.1. The simplest baseline, SA with w2v, achieved an accuracy rate of 78.73%. This result is important because it shows the validity of the *word2vec* representation, performing better than more elaborate methods based on BOW, such as MVB with binBOW.

LSA with BOW features was almost 10% more accurate than SA, where the best results with binary weights were achieved with an approximation with 130 dimensions, with TF weights were achieved with 50 dimensions, and with TF-IDF weights were achieved with 30 dimensions.

[1]https://code.google.com/archive/p/word2vec/

Table 4.1: Results on the Reuters-8 dataset, without stop words. We denote the word subspace representation as WSub and word subspace with single occurrences of words as u-WSub. 'A' stands for accuracy and 'F1' for the macro f1-score metric

Method	Text Model	W.E.	A	F1
SA	-	w2v	78.73	61.66
MSM	WSub	w2v	**92.01**	**80.62**
MSM	u-WSub	w2v	**90.62**	**80.56**
MVB	binBOW	-	62.70	42.37
MNB	tfBOW	-	**91.47**	79.71
LSA	-	w2v	34.58	20.64
	binBOW	-	86.92	71.80
	tfBOW	-	84.78	71.30
	tfidfBOW	-	82.38	74.92
SVM	-	w2v	26.61	13.02
	binBOW	-	89.23	69.25
	tfBOW	-	89.10	69.47
	tfidfBOW	-	88.78	69.18
	LDA	-	**92.00**	72.56

SVM with BOW features was about 3% more accurate than LSA, with the binary weights leading to a higher accuracy rate.

It is interesting to note that despite the reasonably high accuracy rates achieved using LSA and SVM with BOW features, they poorly performed when using *word2vec* features.

Among the baselines, the best methods were MNB with tfBOW features followed by SVM with LDA features, both achieving over 90% accuracy. They were also the only conventional methods that outperformed u-WSub with MSM in terms of accuracy. u-WSub with MSM had an accuracy rate of 90.62% and an f1-score of 80.56%. However, incorporating the frequency information in the subspace modeling resulted in significantly higher accuracy at a 95% confidence level (p-value: 2.79E-06), with WSub achieving 92.01%, and in a slight improvement of the f1-score when compared with u-WSub.

Overall, the WSub, along with MSM, achieved the best results, being significantly more accurate than MNB with tfBOW (p-value: 0.031, at a 95% significance level). On the other hand, while LDA features with SVM had similar accuracy to WSub with MSM, the f1-score was about 8% lower. This difference demonstrates the capability of the subspace model in handling classes of different sizes.

Besides, the best results when considering the frequencies of the words were achieved by using smaller word subspaces (150 to 172 dimensions for reference subspaces and 2 to 109 for input subspaces) than that when using unique occurrences (150 to 181 dimensions for reference subspaces and 3 to 217 for input subspaces).

Table 4.2: Comparison results between MSM and 1NN-MSM on the Reuters-8 database. 'A' stands for accuracy and 'F1' for the macro f1-score

Method	Text Model	W.E.	A	F1
1NN-MSM	WSub	w2v	85.97	74.72
1NN-MSM	u-WSub	w2v	88.77	76.40
MSM	WSub	w2v	**92.01**	**80.62**
MSM	u-WSub	w2v	90.62	80.56

Single Subspace vs. Multiple Subspaces per class

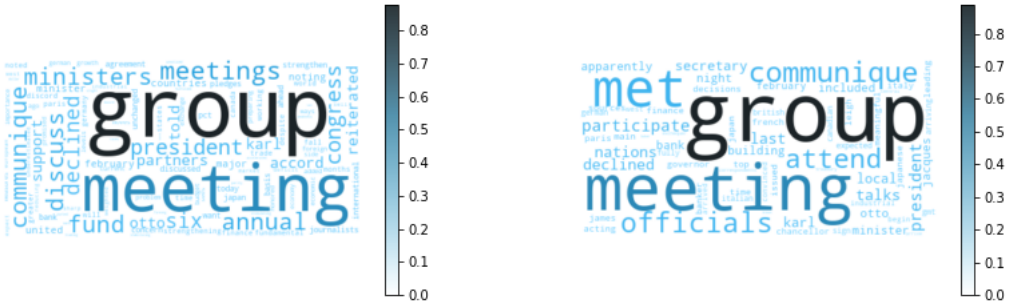
Based on the uniqueness property demonstrated in Section 2.3, we could see that modeling several texts as a single class subspace can lead to a robust representation of a topic class, and therefore MSM performs well for topic classification. However, a common approach taken by most sentence and document embeddings, such as DCT embeddings (DCT) [32], and the EigenSent [30], is to model each sentence or document as a single representation. In the context of subspaces, this means modeling each document as a subspace, resulting in each topic class being represented by several document subspaces. Classification, then, would be performed based on the nearest reference document subspace. Such an approach has some advantages over the proposed MSM, such as making it easier to update the classifier with new samples. To understand if this strategy is effective, we tested it in the same settings as the previous section. We refer to this approach as 1NN-MSM.

The results for 1NN-MSM can be seen in Table 4.2, which were achieved with subspaces dimensions varying from 1 to 15. When we compare our proposed framework utilizing the word subspace and MSM (MSM + WSub), it is clear the advantage of modeling a single reference subspace for each class over modeling several reference subspaces for each class (1NN-MSM + WSub). Even when considering single occurrences of the words, using MSM as proposed achieved better accuracy and F1-score.

To understand why such difference occurs, we use the word importance score defined in Section 3.1. As an example, we selected two texts from the class "money-fx" of the Reuters-8 dataset and modeled a word subspace for each one of them. Then, the importance score of the words in each text was calculated according to the canonical vectors between them.

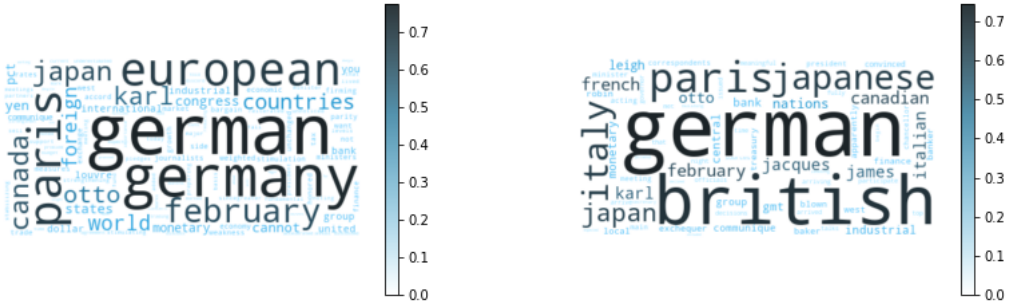
Figure 4.3 shows the word clouds, colored and sized according to the importance score based on the first pair of canonical vectors when comparing the subspaces of Text1 and Text2. Since both texts belong to the "money-fx" class, we expected that words related to money be the ones connecting the texts. However, we can see that words such as "group" and "meeting" were considered more important. This result is probably because both texts talk about meetings happening to discuss finances. The second pair of canonical vectors (Fig. 4.4) highlights words related to the name of the countries mentioned in the texts. Only in the third canonical vector (Fig. 4.5), we start to see words related to money being considered important to compare the texts. Still, these have minor importance, with the most important ones still being related to the meetings and discussion. Thus, although these two texts are very similar in terms of subspace similarity, the main reason is that they both talk about meetings.

On the other hand, if we compare the word subspace for Text1 and the word subspace for the whole class, we can see words related to money receiving more importance. When we compare the



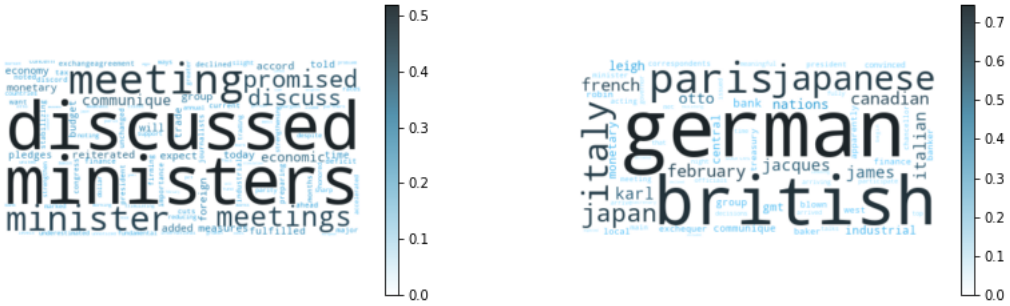
(a) Text1. (b) Text2.

Figure 4.3: Word importance score for words in texts in the class "money-fx" of the Reuters-8 dataset, according to the first pair of canonical vectors between their word subspaces. Words such as "group" and "meeting" are considered more important



(a) Text1. (b) Text2.

Figure 4.4: Word importance score for words in texts in the class "money-fx" of the Reuters-8 dataset, according to the second pair of canonical vectors between their word subspaces. Words related to the countries are considered more important



(a) Text1. (b) Text2.

Figure 4.5: Word importance score for words in texts in the class "money-fx" of the Reuters-8 dataset, according to the third pair of canonical vectors between their word subspaces. Words related to the countries are considered more important

first text with the subspace modeled from all of the remaining texts in the class, we can see that the first canonical vector pair considers words related to money as more important (Fig. 4.6).

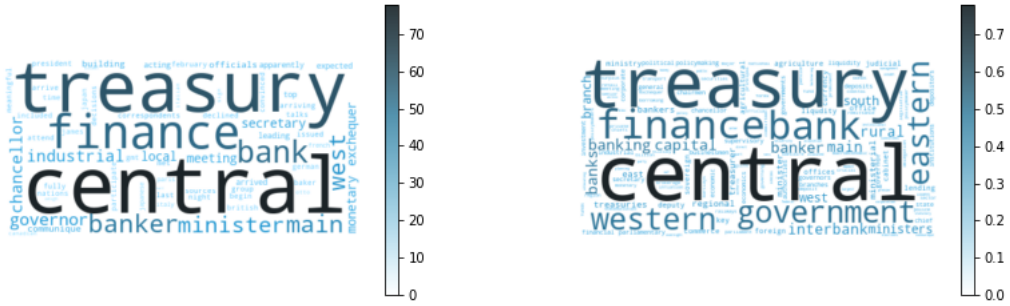
Finally, it is possible to see how the classification differs between the two approaches by visualizing the distribution of the test subspaces using 1NN-MSM and MSM on Figures 4.7a and 4.7b, respectively. As we have different classes with possible different variances, we used the t-Distributed Stochastic Neighbor Embedding (t-SNE) [54] to generate all the visualizations, as this method generates the lower dimensional embeddings in a non-linear and non-local manner. We can see that the word subspaces of seven classes are concentrated in a small region when using 1NN-MSM. In contrast, the word subspaces for all classes spread more appropriately when using MSM, improving classification. Figure 4.8 shows in more detail how the distribution of the test word subspaces of different classes compare when using 1NN-MSM and MSM.

Therefore, while 1NN-MSM can achieve reasonable results, using MSM as proposed in this thesis is more effective for topic classification. Furthermore, this analysis shows the importance of understanding the geometry behind the subspace representation, such as the uniqueness property.

Execution time experiment

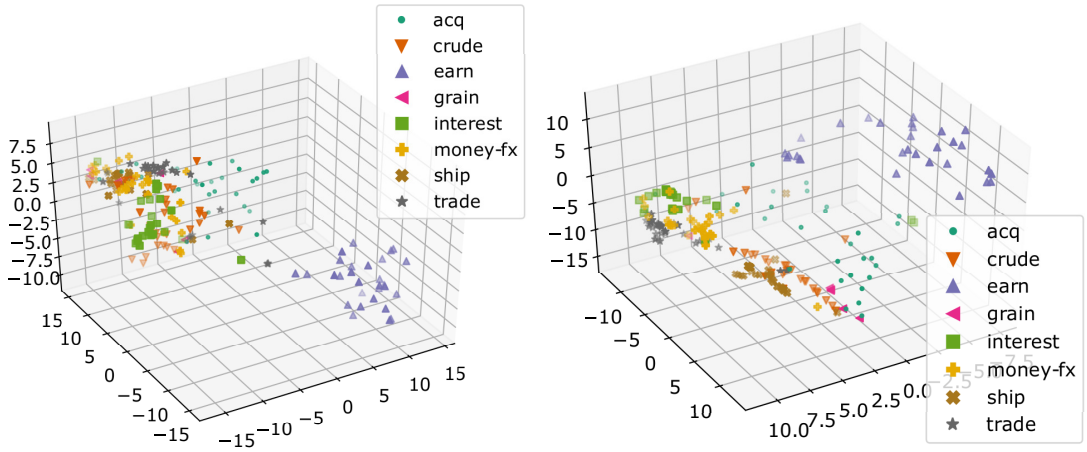
As explained in Section 4.1.2, when modeling all words from all texts into a single word subspace, the number of word vectors in the embedding matrix $X_c = \{\boldsymbol{x}_c^k\}_{k=1}^{N_c}$ can be large. A large embedding matrix can lead to large memory consumption and a slow PCA computation. By using a weighted version of PCA, we reduce the size of the embedding matrix while achieving the same results. In this section, we demonstrate this computational improvement by measuring the execution time of MSM using both the original and the proposed weighted ones. We used the subspace dimensions obtained in the previous experiment and classified the Reuters-8 dataset following the same experimental set-up.

Table 4.3 shows the results of this experiment, where the times were measured for each fold and averaged. We also included the execution times for the comparison methods tested in the experiment



(a) Text1 (b) Class "money-fx"

Figure 4.6: Word importance score in the text 1 in "money-fx", according to the first pair of canonical vectors between its subspace and the class subspace



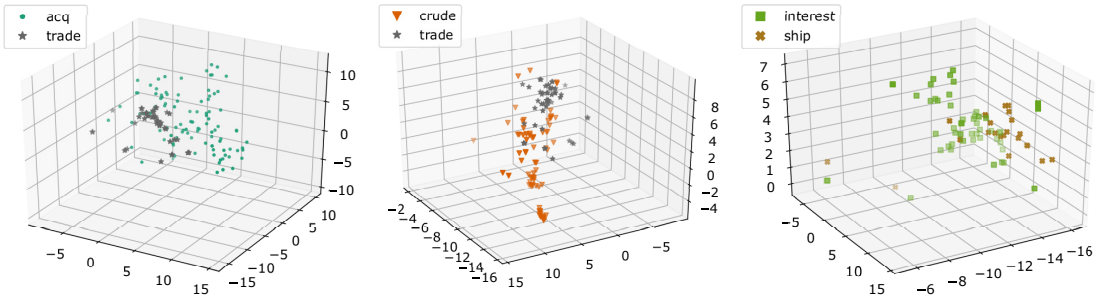
(a) 1NN-MSM (b) MSM

Figure 4.7: Distribution of the test subspaces representing each class in the R8 dataset when using (a) 1NN-MSM and (b) MSM

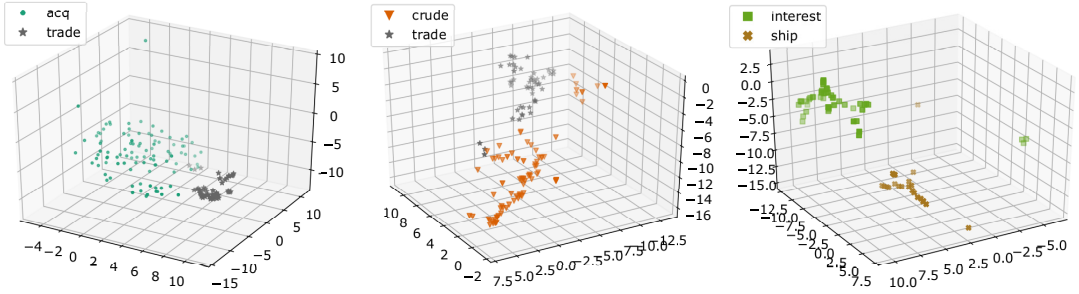
of Section 4.1.3. We can see that using the weighted PCA was more than 30% (1.5 seconds) faster than using regular PCA. Looking at the accuracy, we can see that both achieved the same performance. Nevertheless, we can see that even when using the regular PCA, the subspace-based method is significantly faster than using the traditional methods.

Comparison with recent methods

We compared our methods to more recent text models in the Reuters-8 and the 20newsgroup datasets. These models aim at creating a single vector representation for a sentence based on the word embeddings of the sentence's words. More specifically, we compared with the concatenated



(a) Using 1NN-MSM



(b) Using MSM

Figure 4.8: Comparison of the distribution of test subspaces for different classes of the R8 dataset when: (a) using 1NN-MSM; (b) using MSM. Using 1NN-MSM leads to high overlap between the classes. In contrast, when using MSM, the overlap is reduced, improving the classification performance

power mean embeddings (p-mean) [18], discrete cosine transform embeddings (DCT) [32], and the EigenSent embeddings [30]. We also include results for the PCA sentence embedding (to which we will refer as concatPCA) reported in Kayal and Tsatsaronis' work [30], which is based on the same subspace generation mechanism but represents the text as a concatenation of the basis vectors, instead of using a matrix with the basis vectors.

To allow a direct comparison with the results reported in the published literature, we used the same word2vec pre-trained model as word embedding, and for both datasets, we used the 'no-short' variation, following the standard train-test splits. We reported weighted precision, recall, and F1 metrics to match the published literature for this experiment. To determine the dimensions of the subspaces for MSM, we performed 10-fold cross-validation with the train set and reported the results for the final model on the test set.

The results for this comparison are in Table 4.4. The results for the R8 dataset were achieved with reference word subspaces of 150 dimensions for u-WSub and 120 dimensions for WSub; the input subspaces dimensions varied from 3 to 210 for u-WSub and 2 to 120 for WSub. As for the 20newsgroup dataset, the best results were achieved with reference word subspaces of 240 dimensions for both u-WSub and WSub. As for the input subspaces, they varied from 2 to 210 dimensions for u-WSub, and from 2 to 240 for WSub.

We can see that for the R8 dataset, considering all word occurrences helped increase our

Table 4.3: Results for the execution time experiment: Comparison between modeling the topic class subspaces with regular PCA and with the weighted PCA. For reference, the execution time for some of the comparison methods have also been included

Method	Text Model	Accuracy	Execution time (s)
PCA	WSub	92.01 ± 0.30	4.81 ± 0.08
Weighted PCA	WSub	92.01 ± 0.30	**3.13** ± 0.05
MNB	tfBOW	91.47 ± 0.37	19.25 ± 0.46
LSA	binBOW	86.92 ± 7.49	25.55 ± 8.30
SVM	tfBOW	89.10 ± 0.24	105.39 ± 2.20
SVM	LDA	92.00 ± 0.66	133.32 ± 2.60

Table 4.4: Comparison of the word subspace representation along with MSM with different sentence embeddings. Results for DCT were taken from Almarwani's work [32]; for PCA, p-mean and EigenSent were taken from Kayal and Tsatsaronis' work [30]. All results were based on the word2vec word embedding and standard train-test split was used for both datasets

Method	Text Model	R8			20n		
		P	R	F1	P	R	F1
MSM	u-WSub	95.00	94.83	94.81	**74.93**	**74.73**	**74.65**
MSM	WSub	95.51	95.29	95.34	74.32	73.86	73.77
SVM	concatPCA	83.83	83.42	83.41	55.43	54.67	54.77
SVM	p-mean	96.69	96.67	96.65	72.20	71.65	71.79
SVM	DCT	96.98	96.98	96.94	72.20	71.58	71.73
SVM	EigenSent	**97.18**	**97.13**	**97.14**	72.24	71.62	71.78

methods' performance by almost 1.5% in the F1-score. As for the 20newsgroup dataset, using a single occurrence of the words led to the best results. On top of that, for both datasets, we can see that using the word subspace representation with subspace-based methods led to significantly better results than concatPCA along with SVM.

When compared to p-mean, DCT, and EigenSent, our approaches performed the worst in the Reuters-8 dataset. However, DCT and EigenSent models take the word order into account, which might have helped improve the results.

As for the 20newsgroups dataset, we can see that our methods achieved the best results. Although both datasets contain texts from news articles, the texts in the 20newsgroup are longer (average of 124.69 ± 253.12 words) than the texts in the Reuters-8 dataset (76.23 ± 88.77 words). Therefore, we may assume that the order of the words for longer texts does not make much difference when the intent is to classify the text's topic. Besides, assuming that each text has only one main topic, the longer it is, the higher are the chances that different words related to that topic will appear.

It is crucial to keep in mind that the R8 dataset is highly imbalanced, and looking only at the weighted metrics can be misleading, as they favor classes with more samples. Therefore, we also present the class-wise f1-score to see if our methods perform well both on the larger classes (e.g., 'acq' and 'earn' classes) and on the smaller classes (e.g., 'grain' class).

Table 4.5: Class-wise F1-score of u-WSub and WSub with MSM on the R8 dataset

Class	Number of samples	u-WSub	WSub
acq	2292	96.33	96.74
crude	374	87.50	89.07
earn	3923	98.00	98.28
grain	51	95.23	95.23
interest	271	79.22	81.33
money-fx	293	82.54	85.40
ship	144	82.19	76.92
trade	326	85.53	85.36

These results are shown in Table 4.5. Despite the class 'grain' being the one with the smallest number of samples, it achieved one of the highest f1-scores, after 'acq' and 'earn'. The classes our method struggled the most to classify were the four last ones. Intuitively speaking, the main topic for these four classes is very related to each other, and they may share many terms that similarly co-occur, and thus, they can be harder to be distinguished from each other.

4.2 Sentiment Analysis

In this section, we tackle the task of binary sentiment analysis. The goal is to assign each input text to the class with the same sentiment, which can be positive or negative. In other words, given a training set of documents $D = \{d_i\}_{i=1}^{|D|}$, with known classes $C = \{c_j\}_{j=1}^{2}$, we wish to classify an input document d_q into one of the classes in C (positive or negative).

To solve this task, we first need to pay attention to the nature of the word embeddings. As most of them are trained based on the co-occurrence of the words, adjectives such as "good" and "bad" might end up having word vectors very close to each other, although they convey opposite sentiments. Therefore, to perform efficient sentiment analysis, it is necessary to push the word vectors apart, corresponding to these opposite sentiment words.

To tackle this problem, we propose using a discriminative version of MSM, named the orthogonal mutual subspace method (OMSM) [55]. Through the whitening process [56], this method orthogonalizes the reference subspaces, pushing them further apart.

In topic classification, we assume that words in texts from the same topic class have a common context and, therefore, can be modeled into a single topic class subspace. However, for sentiment analysis, as texts that talk about the same topic may convey opposite sentiment, modeling each sentiment class as a single word subspace might result in two main problems: First, the distribution of the word vectors of each sentiment class might be non-uniform and, therefore, PCA will not be efficient to model it; Second, there may be significant overlap between each sentiment class subspace (i.e., the overlapping topics), leading to little discrimination power.

In this case, we need to disregard each text's main topic and focus on the sentiment it conveys. To perform that, we propose modeling a text subspace for each text and then characterizing each sentiment class by the distribution of the text subspaces it contains. As a set of m-dimensional linear subspaces of $\mathbb{R}^p$ lie as a set of points on a Grassmann manifold $\mathcal{G}(m, p)$ [57], we seek to

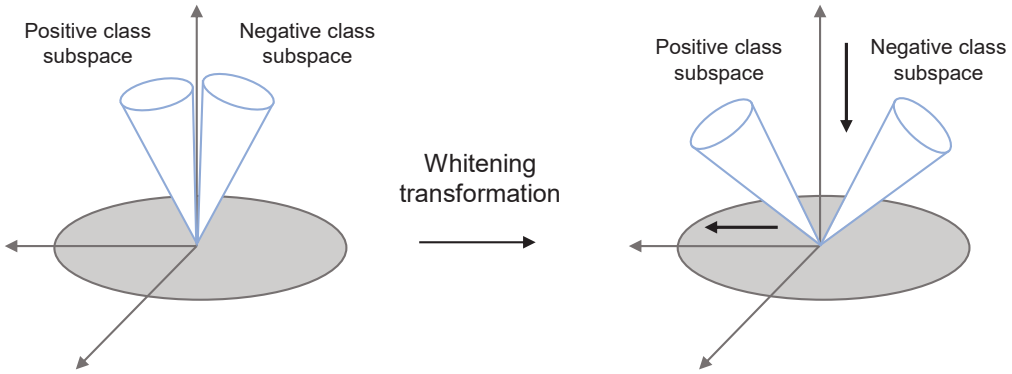


Figure 4.9: Orthogonalization of subspaces by using the whitening transformation

model each sentiment class as a set of points corresponding to its text subspaces on the Grassmann manifold. To perform this, we can use a variation of the MSM on the Grassmann Manifold, the Grassmann subspace method (GSM) [58]. To combine the discriminative power of OMSM with the representational capability of GSM, we ultimately propose using the Grassmann orthogonal subspace method (GOSM) [58] to solve sentiment analysis.

In the following, we explain how to apply these two strategies to solve sentiment analysis based on the theory of the subspace-based methods.

4.2.1 Orthogonalization by the whitening transformation

Most of the traditional word embeddings are trained based on the co-occurrence of the words. Because of that, adjectives such as “good” and “bad” might end up having word vectors very close to each other, although they carry opposite sentiments. This characteristic can lead to low accuracy when performing sentiment analysis of texts based on these word embeddings.

To solve this problem, we propose using a discriminative variant of the mutual subspace method: the orthogonalized mutual subspace method (OMSM). By applying the whitening process, it is possible to orthogonalize the reference subspaces, i.e., maximize their distance. We hypothesize that performing the orthogonalization of the reference subspaces should alleviate the lack of sentiment information of the word embeddings. Figure 4.9 depicts this behavior.

Mathematically, whitening is the process of making all eigenvalues of an autocorrelation matrix the same, i.e., it decreases the standard deviation of the eigenvalues. Moreover, under the subspace representation context, it makes the subspaces distributions uniform, increasing the angles between them [55].

To orthogonalize the subspaces of the positive and negative sentiment reference subspaces, we need to calculate a whitening matrix $\boldsymbol{O}$. First, for each class $c_j \in \{c_j\}_{j=1}^{|C|}$, we define a projection matrix $\boldsymbol{P}_j$, which takes a word vector v and creates its projection in the c_j class subspace. Such a projection matrix can be obtained by the following:

$$\boldsymbol{P}_j = \sum_{i}^{m} \boldsymbol{\Phi}_i \boldsymbol{\Phi}_i^{\top} \tag{4.5}$$

where $\{\boldsymbol{\Phi}_i\}_{i=1}^{m}$ are the basis vectors of the c_j class subspace. Then, we define the total projection matrix $\boldsymbol{G} = \sum_{j=1}^{|C|} \boldsymbol{P}_j$.

The whitening matrix $\boldsymbol{O}$ can be obtained based on the eigenvalues and eigenvectors of the matrix $\boldsymbol{G} \in \mathbb{R}^{v \times p}$ by the following:

$$\boldsymbol{O} = \boldsymbol{\Lambda}^{-1/2} \boldsymbol{E}^{\top} \tag{4.6}$$

where $v = |C| \times m$, $\boldsymbol{\Lambda} \in \mathbb{R}^{v \times v}$ is the diagonal matrix with the i-th highest eigenvalue of the matrix G as the i-th diagonal component, and $\boldsymbol{E} \in \mathbb{R}^{p \times v}$ is a matrix whose i-th column vector is the eigenvector of $\boldsymbol{G}$ corresponding to the i-th highest eigenvalue.

This matrix $\boldsymbol{O}$ whitens the matrix $\boldsymbol{G}$ so that the $|C|$ subspaces are orthogonalized, when $|C| \times m < p$.

When we utilize the whitening transformation along with MSM, we perform the orthogonal mutual subspace method (OMSM). Classification under this framework is very similar to MSM; however, we compute a whitening transformation matrix during the training stage. In the classification stage, we first apply this whitening transformation to all reference and input word subspaces through the following equation:

$$\boldsymbol{Y}_o = \boldsymbol{O}\boldsymbol{Y}, \tag{4.7}$$

where $\boldsymbol{Y}$ is the matrix whose column vectors correspond to the basis vectors of the subspace which we wish to transform. Then classification follows as described in MSM.

It is important to note that the transformed basis vectors in $\boldsymbol{Y}_o$ are not guaranteed to be orthogonal among themselves and, therefore, to calculate the subspace similarity, it is necessary to orthogonalize the transformed basis vectors through the Gram-Schmidt orthogonalization process.

This orthogonalization process is done blindly, i.e., all words from the positive class will be pushed apart from all the words of the negative class. While this can affect words with similar meanings in similar contexts (e.g., therefore and hence), we assume that such relationship changes between these words are not as crucial for sentiment analysis as words that carry sentiment information.

4.2.2 Subspace representation on a Grassmann manifold

Since texts about different topics may convey the same sentiment, we cannot model a single word subspace directly from the word vectors to represent a whole sentiment class. Instead, it is possible to model each text as a text subspace and then characterize each sentiment class by the distribution of the subspaces it contains. These subspaces lie as points on a Grassmann manifold, and, therefore, we seek to model each sentiment class as a set of points corresponding to the subspaces on the Grassmann manifold.

To perform this, we can use a variation of the MSM on the Grassmann Manifold, the Grassmann subspace method (GSM). The MSM presented in Section 4.1 is regarded as the most straightforward method on the Grassmann manifold, where the classification is performed by using the similarity

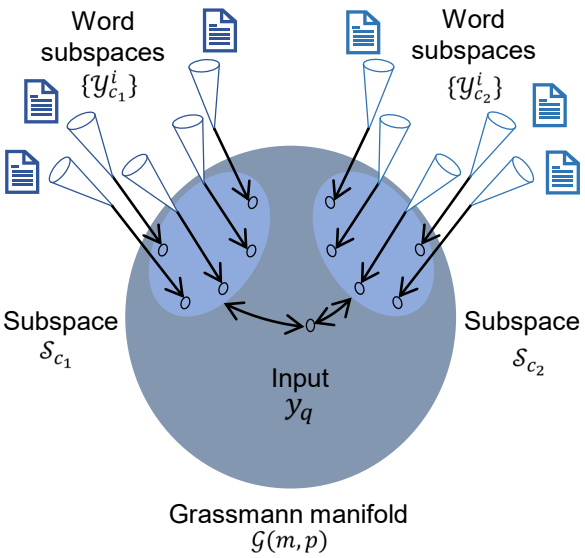


Figure 4.10: Subspace representation on a Grassmann manifold

between a reference point and an input point. In this case, classification compares a set of reference points on the Grassmann manifold with an input point.

The Grassmann manifold $\mathcal{G}(m,p)$ is defined as a set of m-dimensional linear subspaces of $\mathbb{R}^p$ [57]. A Grassmann manifold can be embedded in a reproducing kernel Hilbert space by using a Grassmann kernel [59]. In this work, we use the projection kernel, defined as the following:

$$k_p(\mathcal{Y}_1, \mathcal{Y}_2) = \frac{1}{m}\sum_{j=1}^{m} \cos^2\theta_j \tag{4.8}$$

which is homologous to the subspace similarity.

Then, a text subspace $\mathcal{Y}$ can be represented as a vector with regards to a reference subspace dictionary $\{\mathcal{Y}_q\}_{q=1}^N$ as:

$$\begin{aligned} \boldsymbol{y} &= k_p(\mathcal{Y}, \mathcal{Y}_q) \\ &= [k_p(\mathcal{Y}, \mathcal{Y}_1), k_p(\mathcal{Y}, \mathcal{Y}_2), ..., k_p(\mathcal{Y}, \mathcal{Y}_N)] \in \mathbb{R}^N \end{aligned} \tag{4.9}$$

Figure 4.10 shows a conceptual diagram of the word subspace representation on a Grassmann manifold. Consider the set of training word subspaces $T = \{\mathcal{Y}_i\}_{i=1}^{|D|}$ corresponding to the documents in the training set D, with known sentiment classes $C = \{c_j\}_{j=1}^2$. We obtain a set of vectors $\{\boldsymbol{y}_i\}_{i=1}^{|D|}$ corresponding to each training word subspace by using equation 4.9 with respect to T. Through the kernel trick using the projection kernel, we now have a set of points on the Grassmann manifold corresponding to each sentiment class. GSM models the sentiment class subspaces $\{\mathcal{S}_{c_j}\}_{j=1}^2$ based on the set of points corresponding to each class. GOSM further performs the whitening transformation to the two class subspaces, $\mathcal{S}_{c_1}$ and $\mathcal{S}_{c_2}$ on the Grassmann manifold.

4.2.3 Experimental Evaluation

This section explains the experiments performed to test the subspace-based methods and the word subspace model on the sentiment analysis task. We separate this section into two parts. In the first part, we verify if the word subspace representation is valid for sentiment analysis. We also analyze how the orthogonalization and the representation on the Grassmann manifold help to improve the results. Then, in the second part, we compare our methods with recent text models and discuss the advantages and disadvantages of the proposed methods.

In this experiment, we used two datasets: The movie review dataset v2.0 (MR)[2], proposed by [60], and the binary version of the Stanford sentiment tree dataset (SST-2)[3], proposed by [61]. Both datasets contain data extracted from movie reviews; However, the MR dataset aims at sentiment analysis on a document level, while the SST-2 dataset aims at sentiment analysis on a sentence level. For the MR dataset, we used the standard train-test split, in which we performed 10-fold cross-validation with the train set to determine the word subspaces dimensions and report the results of the final model on the test set. For the SST-2 dataset, we used the standard train and dev sets to determine the word subspaces dimensions and report the final model results on the test set.

Subspace-based methods

In this first part, we used the 1NN-MSM as a baseline. We performed an ablation study to see the effects of the representation on the Grassmann manifold and the orthogonalization process in the classification results. In addition to testing with word2vec, we also tested by using GloVe[4] and BERT[5] word embeddings. Specifically for BERT, we considered the average of the token representations given by the 4 last layers.

Table 4.6 shows the results for this experiment. First of all, we can see that better results are achieved for both datasets as better word embeddings are used. However, using the naive approach of 1NN-MSM does not perform well. For example, using MSM with w2v embeddings performed better than using 1NN-MSM with BERT, an embedding that has been consolidated as a state-of-art representation. Nevertheless, for both datasets and all embeddings, using MSM performed better than 1NN-MSM.

We can see that for the MR dataset, there is a slight improvement when applying the orthogonalization (OMSM) for w2v and Glove over MSM, which supports our assumption that the orthogonalization process can alleviate the lack of sentiment information in these word embeddings.

To illustrate how the orthogonalization helped increase the discrimination between positive and negative sentiment words, we show in Table 4.7 the cosine similarity between opposite sentiment words before and after the orthogonalization of the reference sentiment classes with the word2vec embeddings.

We can see that the orthogonalization helped to push apart these word vectors that contain opposite sentiments. On the other hand, it is interesting to see that for words such as “hence” and “therefore”, which happen in similar contexts and have similar meanings, the cosine similarity

[2]http://www.cs.cornell.edu/people/pabo/movie-review-data/

[3]https://nlp.stanford.edu/sentiment/

[4]http://nlp.stanford.edu/data/glove.42B.300d.zip

[5]https://github.com/google-research/bert (BERT-base uncased model)

Table 4.6: Results for subspace-based methods on the MR and SST-2 datasets. Best results for each dataset per word embedding are highlighted in bold

Word Embedding	Method	Movie Review			SST-2		
		A	R	P	A	R	P
w2v	1NN-MSM	61.18	61.18	61.32	60.21	60.20	60.24
	MSM	76.45	74.20	77.77	75.53	75.52	75.73
	OMSM	76.55	69.40	**80.98**	**75.69**	**75.70**	**75.92**
	GSM	79.75	81.70	78.70	74.28	74.25	78.63
	GOSM	**84.25**	**83.70**	74.66	72.91	72.90	73.09
GloVe	1NN-MSM	64.49	64.48	64.59	61.94	61.93	62.04
	MSM	76.80	72.60	79.27	**77.12**	**77.11**	77.14
	OMSM	77.05	67.80	83.20	76.71	76.70	76.75
	GSM	79.65	81.30	78.80	73.78	73.72	**78.42**
	GOSM	**85.75**	**85.20**	**86.24**	67.80	67.79	67.84
BERT	1NN-MSM	66.93	66.93	67.12	71.41	71.40	71.58
	MSM	77.44	74.87	79.00	81.59	81.59	81.63
	OMSM	73.66	74.62	73.33	83.24	83.24	83.25
	GSM	86.24	86.23	86.39	82.45	82.43	84.24
	GOSM	**91.14**	**91.14**	**91.20**	**85.51**	**85.50**	**85.53**

Table 4.7: Cosine similarity between word2vec embeddings of words with opposite sentiment before and after the orthogonalization of the reference word subspaces in the Movie Review dataset

Words		Before	After
Good	Bad	0.71	0.17
Incredible	Terrible	0.47	0.23
Excellent	Awful	0.40	0.16

almost did not change (0.70 to 0.75). This result might be because these words are not specific to any sentiment and similarly occur in both classes.

For BERT, the orthogonalization process made the results significantly worse. This result is likely to be related to the fact that BERT word embeddings are dynamically generated, where each word embedding depends on the words around it in the text. The word embeddings might represent some helpful sentiment information for this classification in this process. Therefore, the orthogonalization process might not be beneficial.

The results in the MR dataset further improved when using the representation on the Grassmann manifold, which is consistent with our assumption that mixing texts from different topics in a single reference word subspace does not lead to satisfactory results for sentiment analysis. Therefore, by modeling each text as a single word subspace, we can increase the abstraction level by projecting them onto the Grassmann manifold, where modeling the sentiment class subspaces will not be directly affected by the different topics. The best results in this dataset were achieved when combining both approaches for all the embeddings.

Table 4.8: Accuracy results for the MR and SST-2 datasets between the proposed methods and different sentence embeddings

Word Emb.	Method	Text Model	Movie Review	SST-2
w2v	MSM	WSub	76.45	75.53
	GOSM	WSub	**84.25**	72.91
	LogReg	concatPCA	65.74	71.94
	LogReg	p-mean	76.30	79.90
	LogReg	DCT	77.10	**81.00**
GloVe	MSM	WSub	76.80	77.12
	GOSM	WSub	**85.75**	67.80
	LogReg	concatPCA	63.43	50.58
	LogReg	p-mean	77.10	80.20
	LogReg	DCT	77.05	79.63
	LogReg	WR	-	82.20
	LogReg	GEM	78.80	**83.60**

For the SST-2 dataset, we see different behavior. OMSM performed better than GSM for all three embeddings. This result is probably due to the short length of the sentences in this dataset. For short sentences to express a sentiment, it is more likely that they contain more words that carry a sentiment, in which case, the classification can benefit from the orthogonalization.

Notice, however, that the word subspace with BERT was the only one to improve MSM when using GSM. While the subspace representation is very efficient at generalizing the distribution of a set of vectors, the subspace cannot create a precise representation if there is not enough variance within the set. This lack is compensated by BERT, as each word embedding depends on each sentence's words, creating more unique representations for each sentence.

Comparison with recent methods

In this section, we compared our methods against the following text representation models: the concatenated power mean embeddings (p-mean) [18], discrete cosine transform embeddings (DCT) [32], geometric embedding (GEM) [27], and the a random-walk based embedding (WR) [17]. To allow a direct comparison with the results reported in the published literature, we used the same word2vec and GloVe pre-trained models as word embeddings.

The p-mean, GEM, and WR results were taken from their respective papers. For DCT, as the results reported in the original paper for these two datasets were based on the Fast-text word embeddings, we trained a logistic regression based on w2v and GloVe embeddings. These results were achieved by considering only the c^0 coefficient. We also computed the results for the concatPCA sentence embeddings to see how they compare with our proposed methods.

Table 4.8 shows the results. For the MR dataset, we achieved the best results when using the GOSM. The concatPCA embedding, on the other hand, achieved the worst results. These results support the findings in the topic classification experiment, which shows that using a classification framework compatible with the word subspace model is much more efficient than forcing it to work

with conventional machine learning methods through concatenation.

For the SST-2, we can see that the word subspace, along with the subspace methods, did not perform well. Instead, models that consider the word order, such as DCT and GEM, achieved the best results. This result also suggests that the order of words might be more critical for shorter sentences.

4.3 Summary

This chapter proposed a new framework for text classification based on subspace-based methods. We also proposed a generalized formulation for the word subspace, which can significantly speed up the modeling of a subspace from a large number of words. We specifically tackled two sub-tasks within text classification: Topic classification and sentiment analysis.

For topic classification, as words important for the classification tend to occur in a specific context (i.e., topic), we assumed that words from texts of the same class belong to the same context. Based on this assumption and the subspace uniqueness property, we hypothesized that there should exist a word subspace spanned by these important words that can be derived from all texts in the class. Therefore, we proposed using the mutual subspace method (MSM), where words from texts of the same class are assumed to belong to the same context.

For sentiment analysis, as word vectors generated by word embeddings of words that carry opposite sentiment tend to be close to each other, we proposed using a discriminative version of MSM, the orthogonal mutual subspace method (OMSM), to reduce the similarity between them. Furthermore, to avoid the overlap between sentiment class subspaces (as texts about the same topic may convey opposite sentiment), we proposed modeling each text as a word subspace. However, it is also necessary to understand the distribution of word subspaces for each sentiment class. As these subspaces lie on a Grassmann manifold, we proposed using a variation of MSM and OMSM, the Grassmann subspace method (GSM) and the Grassnmann orthogonal subspace method (GOSM).

Our experiments demonstrated the effectiveness of the word subspace model when compared against conventional text classification methods and recent non-parameterized text models. Despite the limitation of not accounting for word order, the understanding and incorporation of the uniqueness property of the subspace representation helped improve the results compared with the framework of 1NN-MSM. Furthermore, our results showed that using MSM led to significantly better results than concatenating the basis vectors along with standard machine learning algorithm, and ultimately, our approach achieved the best results in the 20newsgroup dataset. We further demonstrated how using the weighted version of PCA helps speeding-up the modeling of a topic class subspace in more than 30% with no performance loss. Finally, combining the discriminative power of OMSM and the representational power of GSM by using the Grassmann orthogonal subspace method achieved the best results on the Movie Review dataset.

Chapter 5

Word Subspace for Multimedia Generation

This chapter presents a framework for multimedia generation, where we demonstrate a more practical application of the word subspace model. We specifically focus on the generation of memes from news articles.

Internet Memes have gained prominence due to their simplicity and comicality. This term is used to describe an activity, concept, catchphrase, or piece of media (e.g., an image, hyperlink, video, website, or hashtag) that spreads by mimicry or for humorous purposes via the Internet [62]. One of the most popular types of memes is the "image macro" meme [63], a combination of a phrase and image, which uses irony and sarcasm to depict a general opinion.

Due to their rapid spread, Internet memes have gained much attention in the past few years. They can be viewed as a form of art, as in websites such as knowyourmeme.com, memedump.com, or memebase.com, and also as solid public relations and advertising tools, with examples of memes purposely designed to create publicity for products or services [64]. Nevertheless, they spread from user to user on social networks through mimicry, commentary, or parodies, usually containing some inside joke or sarcasm. Therefore, it is not uncommon to have a rapid increase of memes surrounding an event with a significant impact on society, spread through the news.

While the study of the relationship between meme creation and news events is out of the scope of this thesis, we recognize the potential of increasing the reach of important news to a public that might not actively seek such information on news platforms and, therefore, we are interested in tackling the problem of meme generation from news articles.

Previous works have successfully generated memes from posts on Twitter and news headlines; however, most of them did not extract information automatically from both images and texts to create the memes. For example, Costa, Oliveira and Pinto [65] search for the most frequent nouns associated with a public figure and replaces them in quotes, creating new phrases. However, the image is retrieved from the internet using a search engine, with no analysis of the image context. In a different work [66], despite using common meme images, they matched the headlines based on a set of rules manually defined for each image.

Wang and Wen [67] studied the correlation among popular meme images and their wordings, retrieving meme descriptions from raw images. Their results showed that extracting information

from both image and text generates meaningful memes, with descriptions more coherent with the image context.

More recently, works using the Show and Tell model [68], an image captioning framework, to generate memes were also proposed. This framework consists of first encoding the meme image using a convolutional neural network, such as VGG-16 [69], as in Akandjani and Bouk's work [70], and Inception-v3 [71], as in Peirson and Tolunay's work [72]. Then, the encoded image is given as an input to an LSTM network that outputs the corresponding meme caption. While such a framework can generate coherent captions, the meme generation is heavily conditioned on the input image.

Finally, works such as memebot [73] generate a meme by combining a selected meme image and a transformer-generated caption, given an input sentence. Memebot is a robust model which can generate memes by leveraging inputs from different modalities of data but requires heavy supervised training.

Considering the above discussion, we propose the *news2meme*, a framework for automatic "image macro" meme generation from news articles. We attempt to leverage information automatically extracted from both text and image to generate the meme without heavy supervised training. Our input is a news text, and the output is a meme composed of an image and a catchphrase. Our problem is then formulated as two multimedia retrieval tasks where we wish to retrieve a meme image and a catchphrase that matches the content of the input news text well.

To solve our problem, we need to compare and match three different information sources: a meme image, a catchphrase, and a news text. To this end, they must be represented in a common form for direct comparison. Our basic idea to address this issue is to represent the three sources as sets of word vectors as follows: Words in the news text and catchphrase are translated to word vectors using the *word2vec* representation [1]. For the images, first, a set of tags is extracted from them by using a deep neural network. Then, these tags are then translated to word vectors by using the *word2vec*.

Under this framework, we represent each set of words compactly as a word subspace in the same vector space and calculate the similarity between two word subspaces by using the mutual subspace method (MSM) [45]. Thus, we can link and compare the different types of information sources naturally and effectively. In this way, we can realize the framework to retrieve the meme image and catchphrase from a given news text query.

The rest of this chapter is organized as follows. First, in Section 5.1, we describe our proposed meme generator, explaining how to match the three types of media. Then, we performed tests to evaluate our framework, and their main results are described in Section 5.2. Finally, we present a brief summary in Section 5.3.

5.1 Proposed Meme Generator

In this section, we first give a general overview of our framework with its basic concept. Then, we explain how to model a word subspace from texts and images and how retrieval is performed through word subspace.

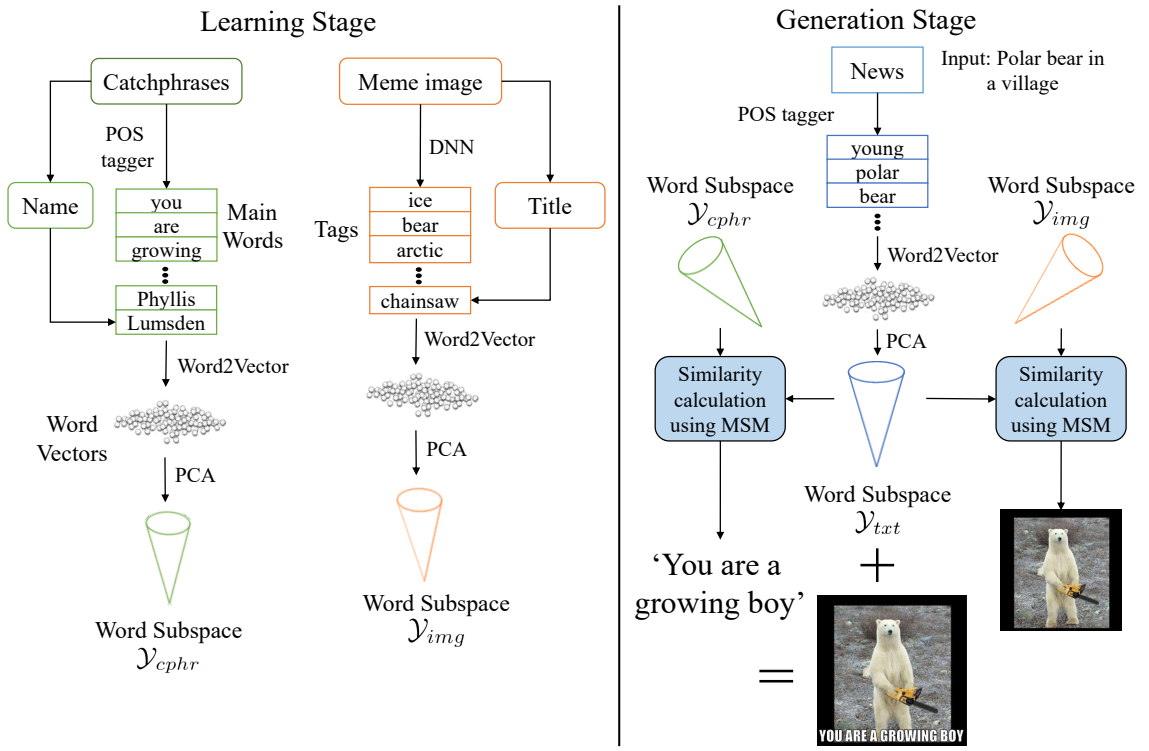


Figure 5.1: Flowchart of the proposed framework. Main words and image tags are extracted from catchphrases and news articles, using a POS tagger, and from meme images, using a DNN; Words and tags are then translated to vectors using the *word2vec* representation. Each set of word vector is modeled into a word subspace $\mathcal{Y}$, and the similarity between them is calculated using MSM.

5.1.1 Framework overview

The primary goal of *news2meme* is to generate a meme from a news text. Figure 5.1 shows our framework. To generate a meme, we find an image and a catchphrase corresponding to the news text by comparing word subspaces.

Our framework has two different stages: A learning and a generation stage. In the learning stage, we consider two different sources: $\mathcal{S}_{img}$, with meme images; and $\mathcal{S}_{cphr}$, with catchphrases. For each meme image and each catchphrase in those sources, we model a word subspace, resulting in sets of catchphrase word subspaces, $\mathcal{Y}_{cphr}$, and sets of image word subspaces, $\mathcal{Y}_{img}$. These are the reference word subspaces.

Then, in the generation stage, for a given input news text in the source $\mathcal{S}_{txt}$, we model an input word subspace $\mathcal{Y}_{txt}$. Next, we calculate the similarity between the input word subspace and the reference word subspaces (images and catchphrases) using MSM. The image and catchphrase with the highest similarity are retrieved. Finally, the selected image and catchphrase are combined to generate the meme.

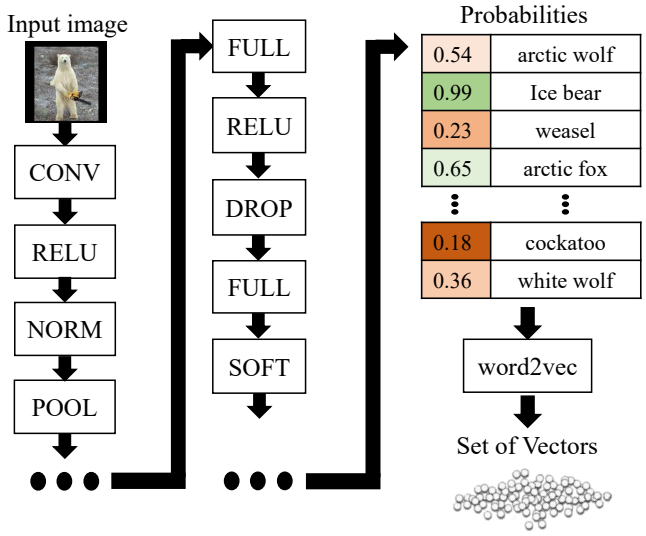


Figure 5.2: Flowchart of tag vectors extraction from imges using a deep neural network (DNN)

5.1.2 Word Subspace from different medias

To model all three sources (i.e., images, catchphrases, and news text) into word subspaces in the same vector space, we first perform the following preprocessing:

Text data: For each input news text and catchphrase, we use the Stanford part-of-speech tagger[1] [74] to extract a set of meaningful words (i.e., verbs, nouns and adjectives). Then, these words are translated to word vectors, using *word2vec*, resulting in sets of word vectors. The set of word vectors from an input text is denoted as $\{x^i_{txt}\}_{i=1}^{N_{txt}}$, while the vector set of a catchphrase is denoted as $\{x^i_{cphr}\}_{i=1}^{N_{cphr}}$.

Image data: Figure 5.2 shows our preprocessing for images. To make images compatible with text media, we represent them as sets of tags, which are extracted by using a deep neural network, the *AlexNet* [75]. Given a pre-defined set of tags, it extracts semantic information from the image in the form of a vector of probabilities among them. The N_{img} most likely tags are then converted to vectors using *word2vec*. The resulting set of tag vectors of an image is represented as $\{x^i_{img}\}_{i=1}^{N_{img}}$.

Under this setting, a set of words from a text contains syntactic information and can be seen as an ordered set, while the image tags are naturally non-ordered. Still, modeling both types of media as word subspaces can effectively represent the context of the corresponding text or image in the same vector space.

5.1.3 Retrieval based on word subspace

By representing all three types of media as word subspaces, we can compare them based on the similarity between the word subspaces. This way, it is possible to not only retrieve information from

[1]https://nlp.stanford.edu/software/tagger.shtml (3.4.1)

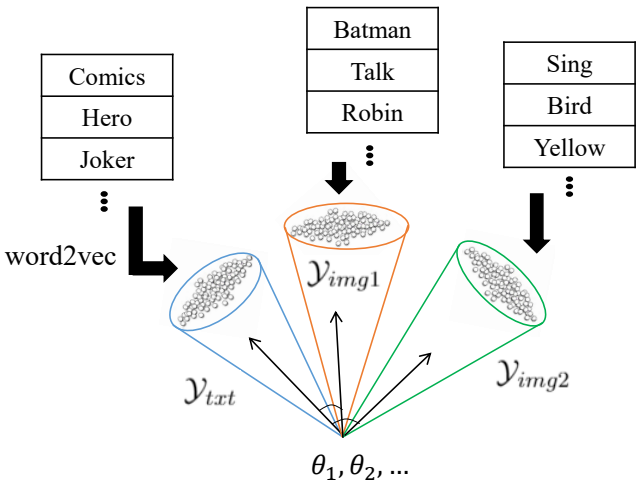


Figure 5.3: Comparison of sets of word vectors by the mutual subspace method

the same modality (e.g., a text from a text) but also retrieve information across modalities (e.g., an image from a text).

Consider an input word subspace for news text data, $\mathcal{Y}_{txt}$, and a reference word subspace for image, $\mathcal{Y}_{img}$. We can compare them by measuring the canonical angles θ between them under the framework of MSM [45]. The canonical angles are defined as the arccosine of the singular values obtained by applying SVD [34] to the matrix $\boldsymbol{Y}_{txt}^{\top}\boldsymbol{Y}_{img}$, where $\boldsymbol{Y}_{txt} \in \mathbb{R}^{p\times m_{txt}}$ and $\boldsymbol{Y}_{img} \in \mathbb{R}^{p\times m_{img}}$ are the bases matrices of $\mathcal{Y}_{txt}$ and $\mathcal{Y}_{img}$, respectively. Ultimately, the similarity between these two word subspaces is defined by using t angles as follows:

$$S[t] = \frac{1}{t}\sum_{i=1}^{t}\cos^2\theta_i,\ 1 \le t \le m_{img}. \tag{5.1}$$

Figure 5.3 shows the modeling and comparison of sets of words by MSM. This method can compare sets of different sizes and naturally encodes proximity between sets that have common words or related words. For example, the word subspace $\mathcal{Y}_{txt}$ of an input news text with the words "hero" and "comics" may be closer to an image word subspace $\mathcal{Y}_{img1}$ containing the tag "batman" than an image word subspace $\mathcal{Y}_{img2}$ containing the tags "bird" and "yellow".

5.2 Experimental Evaluation

In this section, we discuss the validity of *news2meme* through two preliminary qualitative experiments. We first describe the datasets we created to perform these experiments. Then, we describe the design of each experiment and summarize our main results.

5.2.1 Datasets

We created three different datasets:

- Meme image dataset S_{img}, with 812 images commonly used in memes, downloaded from the *Meme Generator* website [2]. For each image, we also extracted the image title.
- Catchphrase dataset S_{cphr}, with 1193 phrases taken from famous series, movies, and cartoons, downloaded from the *catchprase.info* website[3].
- News dataset S_{txt}, with 2517 news articles from the *News in Levels* website[4], from the categories 'History' (46 articles), 'Nature' (119), 'Sports' (93), 'Interesting facts' (1040), 'Funny' (61), and 'News' (1158). We chose this website over other news websites because it has short and simplified versions of news articles.

5.2.2 Meme Generation

We generated 2517 memes from news articles in the news database S_{txt} using our proposed framework. As training data, we used image tags extracted from images in the database S_{img} and main words from catchphrases in the database S_{phr}.

For the image tags, we considered the top 5 predictions by *AlexNet* for each image and added words from the image title. As for the catchphrases, we extracted the main words (nouns, adjectives, and verbs) using the POS tagger and added the name of the character who says the catchphrase.

These sets of words and tags were then translated to vectors, keeping only one occurrence of each word. PCA was then applied to each set of vectors, thus creating 812 meme image word subspaces $\{\mathcal{Y}_{img}^{i}\}_{i=1}^{812}$ and 1193 catchphrase word subspaces $\{\mathcal{Y}_{cphr}^{i}\}_{i=1}^{1193}$. We set the dimensions of the word subspaces, $\mathcal{Y}_{img}$ and $\mathcal{Y}_{cphr}$, to values ranging from 4 to 7 and from 3 to 8, respectively.

We used the news in S_{txt} as inputs, generating one meme for each news as described in section 5.1. The word subspaces $\mathcal{Y}_{txt}$ dimensions ranged from 7 to 38.

5.2.3 Meme evaluation experiment

We used 990 memes generated following the procedure in Section 5.2.2. Nine subjects were asked to read 110 news articles each and evaluate their corresponding generated memes regarding the image and phrase. Subjects voted them as 'Good' when they depicted well the news and 'Bad' when they did not. To better understand the image and phrase combination, we gave a score for each meme. Considering 'Bad' as 0 and 'Good' as 1, we summed both votes, obtaining a maximum score of 2 when both image and phrase were 'Good' and a minimum score of 0 when both were 'Bad'. Table 5.1 shows the percentage of the three possible scores each category received.

Memes with the highest scores were from the 'Sports' and 'Nature' categories. We can see an example in 5.4a, in which the news is about a young polar bear that strayed into a village in the Russian Arctic[5]. *News2meme* successfully related the news with a polar bear picture and found a

[2]https://imgflip.com/memegenerator

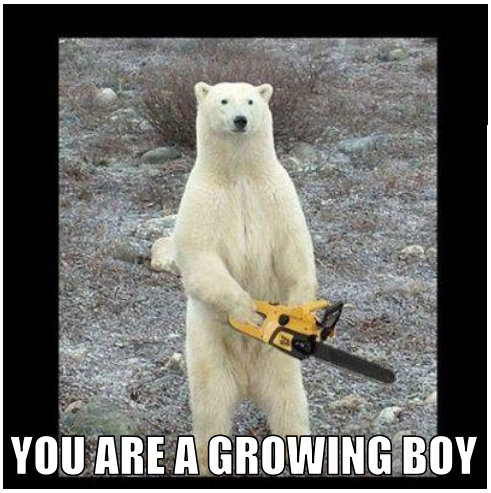
[3]http://www.catchphrases.info/

[4]http://newsinlevels.com

[5]https://www.newsinlevels.com/products/polar-bear-in-a-village-level-2/

Table 5.1: Meme Evaluation Results - General Score (%)

	General		
	Both Bad	One Good	Both Good
Sports	20.43	24.73	**54.84**
Nature	17.65	31.93	**50.42**
History	41.30	19.57	39.13
Interesting	29.45	44.36	26.18
News	71.21	16.67	12.12
Overall	30.66	37.18	32.15



(a) 'Polar Bear in a Village'.

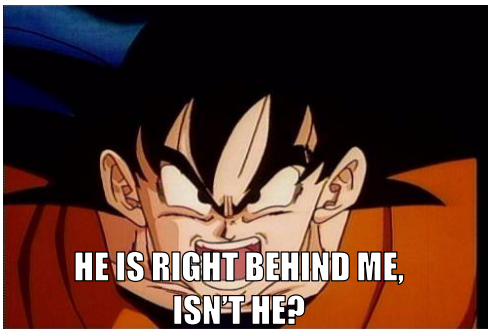
(b) 'Spider inside a man's body'.

Figure 5.4: Example of: (a) Good Meme and (b) Bad meme. Images taken from the website: *imgflip.com/memegenerator*

connection between "young" and "boy". On the other hand, memes from the 'News' category were the ones with the lowest scores. Analyzing the comments from the subjects, we noticed that most of the articles from this category reported tragic events and, therefore, subjects voted 'Bad' regardless of the image and catchphrase.

Participants also reported cases where the generated meme had a 'Good' image, but a 'Bad' catchphrase and vice-versa. One example can be seen in Fig. 5.4b, which was generated from an article about a spider inside a man's scar[6]. The phrase translates as what could be seen as the spider's point of view. However, the picture shows a man falling in the water.

(a) 'Star Wars Exhibition'.



(b) 'Human vs. Horse'.

Figure 5.5: Example of: (a) Meme with direct relation and (b) Meme with interesting relation. Images taken from the website: *imgflip.com/memegenerator*

5.2.4 Analysis of the Generated Memes

In this section, we analyze some of the generated memes by using the word importance score presented in Section 3.

For some memes, the relation between the news text and the retrieved image and catchphrase was straightforward. For example, Fig. 5.5a shows the meme generated from a news article about a Star Wars exhibition[7]. The image shows Yoda, a character of Star Wars, which shows that *news2meme* just related the character cited in the news (*"It includes an original Darth Vader suit, a **Yoda** puppet, [...]"*) with the image. As for the catchphrase, it made the connection based mainly on the word "game", which also appears in the news article (*"Visitors to the exhibition can also play a **game**, [...]"*).

On the other hand, some of the generated memes at first did not seem to have any relation to their news. Figure 5.5b shows one example. When inputting a news article about a professional sprinter that ran faster than a racehorse[8], *news2meme* generated a meme with an image of Goku, an animation character, with the catchphrase "He is right behind me, isn't he?". While "Goku" is among the tags of the image, it is unclear how this entity is related to the text. Therefore, we looked among the most similar words to "Goku" and found "dragon". Then, looking at the similar words for "dragon", we found "fast", which relates to the news text. We also analyzed the word importance scores when comparing the image with the text, noticed that the words "legs", "horse", and "faster" in the text received the highest scores, while the word "Goku" and "shoes" in the image tags received the highest score. Therefore, although "Goku" and "fast" were not directly connected, the subspace representation encoded such indirect relation.

[6]https://www.newsinlevels.com/products/spider-inside-a-mans-body-level-2/
[7]https://www.newsinlevels.com/products/star-wars-exhibition-level-2/
[8]https://www.newsinlevels.com/products/horse-vs-human-level-2/

Table 5.2: Representativeness Experiment Results - Percentage of votes

With Generated Memes		With Created Memes	
Generated	Random	Created	Random
62.55	37.45	76.86	23.14

5.2.5 Representativeness experiment

In this experiment, our main goal was to determine whether memes generated by our framework could represent news articles' content better than randomly generated memes. We designed a questionnaire where participants were asked to read ten news articles and choose among four different options which meme better represented them. One meme was generated by our framework (generated memes), and the other three were randomly generated (random memes). We randomly chose these articles from 'Sports', 'Nature', 'History', 'Funny' and 'Interesting Facts' categories of our news database S_{txt};

Because the memes were generated based on a news article input, we expected participants to prefer them over the random ones. However, there was the possibility of participants showing no preference. This could be due to a flaw in our framework or because the participants made random choices. Therefore, we also showed ten news articles with four options, one of which was created by humans from the article (created memes), while the other three were random ones. Participants were unaware of the created memes, and articles with generated and created memes were shown in random order.

This questionnaire was implemented as an online form, totaling 51 evaluations. Table 5.2 shows these results. Created memes were preferred by 76.86% of the participants, which indicates that they were not making random choices. 62.55% of the participants preferred the generated memes over the random ones. While it is clear that created memes are superior, this result shows that our memes are more meaningful than randomly generated ones.

5.3 Summary

In this chapter, we demonstrated how the word subspace can be used in a multimodal setting by proposing the *news2meme*, a framework for generating macro image memes from news articles. To solve this problem, we compared and matched three different media formats: a meme image, a catchphrase, and a news text. Our key idea is to extract tags from images using a DNN, and main words from texts, using a POS tagger. Then, we represent these sets of tags and words as word subspaces. Finally, we used the MSM to compare them and retrieve the most suitable image and catchphrase to a news text.

Our experiments showed that news articles containing tragic stories were perceived as unsuitable for memes. However, when using news articles unrelated to tragic events, participants preferred generated memes over random generations. This result shows that our framework can handle news articles and unconstrained images. Moreover, we demonstrated how to interpret the generation results of this framework by using the word importance score.

Chapter 6

Concluding remarks

This chapter summarizes the results presented in this thesis, and discusses future research directions.

6.1 Summary

In this thesis, we proposed representing a sentence or text as linear subspaces from their word embeddings, to which we refer as word subspaces, to solve different tasks in the natural language processing field. We specifically focus on the word subspaces modeled from conventional word embeddings, such as word2vec, as these embody the distributional hypothesis of meaning, where the meaning of words is defined by contexts in which they co-occur. We build on top of the linguistic intuition that the meaning of sentences is composed by the meaning of their constituent words and try to model such context with the subspace representation.

The primary motivation to apply such representation to the natural language data comes from an interesting property, the subspace uniqueness. While the subspace is a unique mathematical entity, it can be represented by different basis vectors. At the same time, in NLP, we can express a unique concept by using different words. Based on this analogy, we hypothesized that the same concept word subspace could be derived from different texts, with different words, that talk about this concept.

The word subspace representation is a simple model that does not require computationally intensive learning, can be derived from sentences with different lengths, and is highly interpretable. The basis vectors of a subspace obtained by applying the principal components analysis (PCA) without data centering can be regarded as the main hidden topics of the given text. Furthermore, once represented as subspaces, we can efficiently compare texts with different lengths in terms of subspace similarity.

While the subspace model has been extensively applied in computer vision, most of the work in natural language processing undermines the capabilities of this representation and disregards the solid theoretical foundation already developed on top of subspace-based methods. Therefore, in this thesis, we aimed at defining the concept of the word subspace, proposing simple tools grounded on the established subspace theory to understand this model from the NLP perspective.

Such definition was presented in Chapter 2. We explained how to model a word subspace from a sentence, text, or a set of texts. We also explored the subspace uniqueness property, and empirically

demonstrated how different texts belonging to the same topic class can generate almost the same word subspace.

To better understand what type of information does the word subspace represents and how to interpret results given by subspace-based methods when applied to natural language data, we propose a simple tool called the word importance score to perform text analysis based on the word subspace in Chapter 3. We demonstrate how we can perform such analysis on a toy data set.

Based on this understanding of the word subspace representation, in Chapter 4, we solved the problem of text classification. We specifically tackled two sub-tasks: Topic classification and sentiment analysis. For topic classification, we proposed using the mutual subspace method, as it embodies the uniqueness property of subspaces. For sentiment analysis, we first explored the geometrical characteristics of word embeddings and compensated for the lack of sentiment information by proposing the orthogonal mutual subspace method. We also considered the nature of the problem, where texts from similar topics can convey opposite sentiments. We proposed using the Grassmann subspace method and the Grassmann orthogonal subspace method to tackle this difference.

Lastly, in Chapter 5, we proposed the *news2meme*, a framework for automatically generating macro-image memes from news articles. Through this framework, we showed how the word subspace could be a powerful tool to compare data from different modalities, such as images and text. To generate a macro-image meme, tags from images are extracted using a CNN, and main words from texts and catchphrases are extracted using a POS parser. These tags and words are modeled into word subspaces and matched using the MSM. The best image-catchphrase match is used to create a meme representing a news article. Our qualitative experiment showed that news articles containing tragic stories were generally perceived as unsuitable for memes. When using news articles not related to tragic events, participants preferred over generated memes randomly generated memes. This result shows that our framework can handle news articles and unconstrained images without using extra preprocessing techniques.

In each application, we also performed the analysis of the results by using the tools provided in Chapter 3.

6.2 Future Work

Our experiments have demonstrated the effectiveness of using the subspace representation in several natural language processing tasks. However, our results have shown that the word subspace model is still a naive representation. For example, we saw that for some text classification datasets, such as the Reuters-8 dataset and the binary version of Stanford Sentiment Tree, even when using highly discriminative subspace-based methods, the word subspace did not perform well compared with recent text models. A possible explanation for this result might be that both datasets contain relatively shorter texts compared to the 20newsgroup dataset and the Movie Review dataset. While the subspace representation is very efficient at generalizing the distribution of a set of vectors, it will not perform well if there is not enough variance within the set to represent the context of the words.

Besides, the word subspace does not consider the order of the words, resulting in a loss of context information. To overcome this problem, we could potentially use methods that include the order information in subspaces generated from a variety of types of data, such as the randomized time

warping [76] and the Hankel subspace method [39].

Notwithstanding these limitations, understanding the subspace properties significantly improved the results. Simply changing how to model reference class subspaces to incorporate the uniqueness property resulted in significant improvement in the results for topic classification. This result opens the question of which other subspace properties we are still disregarding and how their understanding could further improve results.

Moreover, while we have focused on the word subspaces generated from conventional word embeddings, such as word2vec, our experiments showed the validity of modeling sentences and texts from their contextual word embeddings, such as BERT. Such language models have been applied in several NLP tasks, presenting state-of-the-art results. However, as it is yet unclear what type of information each layer of such model represents, the problem of how to represent a sentence or a text based on such embeddings is still open. Therefore, a natural progression of this work is to apply the word subspace on contextualized word embeddings, seeking to understand better what type of information can be represented through this model.

Furthermore, several questions remain to be answered regarding the application of the subspace representation in multimodal problems. While the subspace-based methods can work with different types of data in single-modality problems, they cannot be directly applied in multimodal settings, as they require that all data lies in the same feature space. In this thesis, we have worked around this problem using the word subspace as a hub to represent data from different modalities. In contrast, further research should be conducted to assess the possibility of directly working with multi-modal data.

Finally, we would like to expand the word subspace application to other NLP tasks. Given the proper interpretation of the word embeddings, we can perform a guided decision on which subspace-based method should be applied. Nevertheless, a better understanding of the geometry of the word embeddings can also lead to the development of new subspace-based methods. As the subspace representation is efficient for similarity calculation, we would like to explore its application to tasks such as extractive summarization and information retrieval.

Appendix A

Toy Data

A.1 Text1: Thousands protest in Brazil over education cuts

Thousands of protesters took to the streets of Brazil's capital Thursday following calls for a second nationwide demonstration in as many weeks over the government's plan to slash education spending.

Far-right President Jair Bolsonaro's government has provoked outrage among students and teachers over its proposal to freeze 30 percent of discretionary spending for public universities in the second half of this year.

A suspension of post-graduate scholarships for students in science and the humanities has also fueled anger. Tens of thousands protested across Brazil on May 15, but Thursday's turnout could be lower after the government said it would free up 1.59 billion reais in funding (about 400 million US dollars) for the sector.

Protests began in Brasilia ahead of demonstrations in Sao Paulo, Rio de Janeiro and other cities later in the day.

"I'm here for those who are poor and deserve the right to quality public education," social services student Kaio Duarte told AFP in the capital. "I'm worried that the next generation won't have all of the rights to education that I have had. This protest makes clear that students will never be silenced."

Thursday's protests come after thousands of pro-Bolsonaro protesters marched in cities across Brazil on Sunday in a show of support for the embattled leader. Among their demands was for Congress to speed up approval of the government's stalled pension overhaul, seen as key to unlocking other much-needed economic reforms.

Source: https://www.france24.com/en/20190530-thousands-protest-brazil-over-education-cuts

A.2 Text2: Brazil's students protest education cuts

Tens of thousands of students and teachers from across Brazil have demonstrated in "defense of education" after spending freezes were announced by the government of far-right President Jair Bolsonaro. Classes at universities and colleges were suspended in Sao Paulo, Rio de Janeiro, Brasilia and 17 of the country's states on Wednesday, local media reported. Government soldiers were seen guarding the Education Ministry in the capital, Brasilia. Protesters said Education Minister

Abraham Weintraub's decision to cut federal university subsidies by 30% would undermine the ability of universities to fulfill their mandate. The Education Ministry also announced last week that it would suspend scholarship payments to postgraduate students in the sciences. "Secondary school pupils, university students, researchers, teachers and other education employees will take to the streets in every state," the National Student Union (UNE) said ahead of the demonstrations. Bolsonaro, who was attending an event in Dallas, Texas, called the protesters "useful idiots, imbeciles, who are being used as the maneuvering mass of a clever little minority who make up the nucleus of many federal universities in Brazil." Bolsonaro fired Weintraub's predecessor, Ricardo Velez, in April after he vowed to stamp out "cultural Marxism" and gender-identity "ideology."

Source: https://www.dw.com/en/brazils-students-protest-education-cuts/a-48753837

A.3 Text3: Strikes, violent protests hit Brazil ahead of World Cup

With barely a month until the World Cup opens in Brazil, violent protests and strikes are breaking out across the country by groups angry about the changes the sporting event has brought – and what it hasn't.

Demonstrations were held in 18 cities Thursday. The biggest and most violent was in Sao Paulo, where police shot tear gas and protesters threw rocks and smashed the windows of a car dealership and a bank. While thousands of people took part in the protests, they were still much smaller than the massive marches seen during the Confederations Cup last year when tens of thousands took to the streets.

In the morning, the Homeless Workers Movement blocked main avenues across the city and about 4,000 people marched on the Arena Sao Paulo where the inaugural game of the World Cup will be held on June 12.

Demonstrators accuse the government of spending billions on new stadiums and not enough on low-income housing. "The World Cup has done nothing to help us," said Diana, a manicurist who has been on a list for a government-subsidized house for a decade. "So we decided to use it as a platform to make our voices heard."

Taking advantage of the global attention focused on the country for the world soccer championship, other groups are staging protests to air their grievances. Across the country in Recife, also a World Cup venue, soldiers were deployed to rein in crime and looting after police went on strike there.

In Sao Paulo, more than 5,000 striking teachers marched to demand higher wages. In the evening, a string of anti-World Cup protests were staged in different cities. In Sao Paulo, activists turned out carrying banners that said "FIFA go home" and "A World Cup without the people means we're back on the street again!" About 1,500 people marched peacefully for a couple of blocks before clashes erupted.

Anti-World Cup protesters and homeless activists vowed to keep up the pressure through the global event that ends on July 13. A total of 600,000 foreign visitors are expected for the cup and another three million Brazilian fans are expected to travel around the country.

Source: https://edition.cnn.com/2014/05/16/world/americas/brazil-world-cup-protests/index.html

Bibliography

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

[2] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[3] M. E. Peters, M. Neumann, M. Iyyer, *et al.*, "Deep contextualized word representations," in *Proc. of NAACL*, 2018.

[4] J. D. M.-W. C. Kenton and L. K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.

[5] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[6] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[7] R. Kiros, Y. Zhu, R. R. Salakhutdinov, *et al.*, "Skip-thought vectors," in *Advances in neural information processing systems*, 2015, pp. 3294–3302.

[8] L. Logeswaran and H. Lee, "An efficient framework for learning sentence representations," *arXiv preprint arXiv:1803.02893*, 2018.

[9] C. S. Perone, R. Silveira, and T. S. Paula, "Evaluation of sentence embeddings in downstream and linguistic probing tasks," *arXiv preprint arXiv:1806.06259*, 2018.

[10] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *arXiv preprint arXiv:1705.02364*, 2017.

[11] D. Cer, Y. Yang, S.-y. Kong, *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.

[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," 2019.

[13] F. Hill, K. Cho, and A. Korhonen, "Learning distributed representations of sentences from unlabelled data," *arXiv preprint arXiv:1602.03483*, 2016.

[14] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188–1196.

[15] R. Tian, N. Okazaki, and K. Inui, "The mechanism of additive composition," *Machine Learning*, vol. 106, no. 7, pp. 1083–1130, 2017.

[16] J. Mitchell and M. Lapata, "Composition in distributional models of semantics," *Cognitive science*, vol. 34, no. 8, pp. 1388–1429, 2010.

[17] S. Arora, Y. Liang, and T. Ma, "A simple but tough-to-beat baseline for sentence embeddings," in *5th International Conference on Learning Representations, ICLR 2017*, 2019.

[18] A. Rücklé, S. Eger, M. Peyrard, and I. Gurevych, "Concatenated power mean word embeddings as universal cross-lingual sentence representations," *arXiv preprint arXiv:1803.01400*, 2018.

[19] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, no. 6, p. 391, 1990.

[20] Y. Yaghoobzadeh and H. Schütze, "Intrinsic subspace evaluation of word embedding representations," *arXiv preprint arXiv:1606.07902*, 2016.

[21] P. S. Dhillon, D. P. Foster, and L. H. Ungar, "Eigenwords: Spectral word embeddings," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 3035–3078, 2015.

[22] V. Raunak, V. Gupta, and F. Metze, "Effective dimensionality reduction for word embeddings," in *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, 2019, pp. 235–243.

[23] J. Mu, S. Bhat, and P. Viswanath, "Representing sentences as low-rank subspaces," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2017, pp. 629–634.

[24] H. Gong, S. Bhat, and P. Viswanath, "Geometry of compositionality," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[25] J. Mu, S. P. Bhat, and P. Viswanath, "Geometry of polysemy," in *5th International Conference on Learning Representations, ICLR 2017*, 2019.

[26] H. Gong, T. Sakakini, S. Bhat, and J. Xiong, "Document similarity for texts of varying lengths via hidden topics," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 2341–2351. DOI: 10.18653/v1/P18-1218. [Online]. Available: https://www.aclweb.org/anthology/P18-1218.

[27] Z. Yang, C. Zhu, and W. Chen, "Parameter-free sentence embedding via orthogonal basis," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 638–648.

[28] B. Wang and C.-C. J. Kuo, "Sbert-wk: A sentence embedding method by dissecting bert-based word models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2146–2157, 2020.

[29] S. Le Clainche and J. M. Vega, "Higher order dynamic mode decomposition," *SIAM Journal on Applied Dynamical Systems*, vol. 16, no. 2, pp. 882–925, 2017.

[30] S. Kayal and G. Tsatsaronis, "Eigensent: Spectral sentence embeddings using higher-order dynamic mode decomposition," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4536–4546.

[31] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[32] N. Almarwani, H. Aldarmaki, and M. Diab, "Efficient sentence embedding using discrete cosine transform," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3672–3678.

[33] O. Yamaguchi, K. Fukui, and K. Maeda, "Face recognition using temporal image sequence," in *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, IEEE, 1998, pp. 318–323.

[34] K. Fukui and O. Yamaguchi, "Face recognition using multi-viewpoint patterns for robot vision," *Robotics Research, The Eleventh International Symposium, ISRR*, pp. 192–201, 2005. DOI: 10.1007/11008941_21.

[35] Y. Ohkawa and K. Fukui, "Hand shape recognition using the distributions of multi-viewpoint image sets," *IEICE Transactions on Information and Systems*, vol. E95-D, no. 6, pp. 1619–1627, 2012.

[36] R. Yataka, L. S. Souza, and K. Fukui, "Feature point extraction using 3d separability filter for finger shape recognition," in *Frontiers of Computer Vision (FCV) 2017*, 2017.

[37] Y. Iwashita, H. Sakano, and R. Kurazume, "Gait recognition robust to speed transition using mutual subspace method," in *International Conference on Image Analysis and Processing*, Springer, 2015, pp. 141–149.

[38] L. S. Souza, B. B. Gatto, and K. Fukui, "Enhancing discriminability of randomized time warping for motion recognition," in *Machine Vision Applications (MVA), 2017 Fifteenth IAPR International Conference on*, IEEE, 2017, pp. 77–80.

[39] B. B. Gatto, A. Bogdanova, L. S. Souza, and E. M. dos Santos, "Hankel subspace method for efficient gesture representation," in *Machine Learning for Signal Processing (MLSP), 2017 IEEE 27th International Workshop on*, IEEE, 2017, pp. 1–6.

[40] B. B. Gatto, J. G. Colonna, E. M. dos Santos, and E. F. Nakamura, "Mutual singular spectrum analysis for bioacoustics classification," in *Machine Learning for Signal Processing (MLSP), 2017 IEEE 27th International Workshop on*, IEEE, 2017, pp. 1–6.

[41] L. S. Souza, B. B. Gatto, and K. Fukui, "Grassmann singular spectrum analysis for bioacoustics classification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2018, pp. 256–260.

[42] T. Nakayama and K. Fukui, "The effectiveness of cnn feature for mutual subspace method," *IEICE Technical Report; IEICE Tech. Rep.*, vol. 117, no. 238, pp. 49–54, 2017.

[43] A. Sakai, N. Sogi, and K. Fukui, "Gait recognition based on constrained mutual subspace method with cnn features," in *2019 16th International Conference on Machine Vision Applications (MVA)*, IEEE, 2019, pp. 1–6.

[44] J. Ishikawa, H. Shiokawa, and K. Fukui, "Subspace representation for graphs," *IEICE Technical Report; IEICE Tech. Rep.*, vol. 119, no. 476, pp. 51–57, 2020.

[45] K. Fukui and A. Maki, "Difference subspace and its generalization for subspace-based methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 11, pp. 2164–2177, 2015.

[46] F. Chatelin, *Eigenvalues of Matrices: Revised Edition*. SIAM, 2012.

[47] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3,4, pp. 321–377, 1936.

[48] S. N. Afriat, "Orthogonal and oblique projectors and the characteristics of pairs of vector spaces," in *Mathematical Proceedings of the Cambridge Philosophical Society*, Cambridge Univ Press, vol. 53, 1957, pp. 800–816.

[49] A. Cardoso-Cachopo, "Improving Methods for Single-label Text Categorization," Ph.D. dissertation, Instituto Superior Tecnico, Universidade Tecnica de Lisboa, 2007.

[50] M. J. Greenacre, *Theory and applications of correspondence analysis*. London (UK) Academic Press, 1984.

[51] I. Jolliffe, *Principal Component Analysis*. Springer Science & Business Media, 2006.

[52] A. McCallum, K. Nigam, *et al.*, "A comparison of event models for naive bayes text classification," in *AAAI-98 workshop on learning for text categorization*, Madison, WI, vol. 752, 1998, pp. 41–48.

[53] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[54] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[55] T. Kawahara, M. Nishiyama, T. Kozakaya, and O. Yamaguchi, "Face recognition based on whitening transformation of distribution of subspaces," in *Proc. ACCV07 Workshop Subspace*, 2007, pp. 97–103.

[56] K. Fukunaga and W. L. Koontz, "Application of the karhunen-loeve expansion to feature selection and ordering," *IEEE Transactions on computers*, vol. 100, no. 4, pp. 311–318, 1970.

[57] Y. Chikuse, "Statistics on special manifolds," *Springer, Lecture. Notes in Statistics*, vol. 174, 2013.

[58] R. Yataka and K. Fukui, "Three-dimensional object recognition via subspace representation on a grassmann manifold.," in *ICPRAM*, 2017, pp. 208–216.

[59] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: A unifying view on subspace-based learning," in *Proceedings of the 25th international conference on Machine learning*, ACM, 2008, pp. 376–383.

[60] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, Association for Computational Linguistics, 2004, p. 271.

[61] R. Socher, A. Perelygin, J. Wu, *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[62] M. Knobel and C. Lankshear, "Online memes, affinities, and cultural production," in *A new literacies sampler*, vol. 29, New York, 2007, pp. 199–227.

[63] P. Davison, "The language of internet memes," in *The social media reader*, JSTOR, 2012, pp. 120–134.

[64] C. Bauckhage, "Insights into internet memes," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 5, 2011.

[65] D. Costa, H. G. Oliveira, and A. M. Pinto, "In reality there are as many religions as there are papers-first steps towards the generation of internet memes.," in *Proc. of 6th International Conference on Computational Creativity, ICCC*, 2015, pp. 300–307.

[66] H. G. Oliveira, D. Costa, and A. Pinto, "One does not simply produce funny memes!–explorations on the automatic generation of internet humor," in *Proc. of 7th International Conference on Computational Creativity, ICCC*, 2016, pp. 238–245.

[67] W. Y. Wang and M. Wen, "I can has cheezburger? a nonparanormal approach to combining textual and visual information for predicting and generating popular meme descriptions," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 355–365.

[68] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.

[69] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[70] G. O. Alandjani and A. H. Bouk, "Meme generation using deep neural network to engage viewers on social media," *Yanbu Journal of Engineering and Science*, vol. 18, no. 1, pp. 84–90, 2021.

[71] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[72] A. L. Peirson V and E. M. Tolunay, "Dank learning: Generating memes using deep neural networks," *arXiv preprint arXiv:1806.04510*, 2018.

[73] A. Sadasivam, K. Gunasekar, H. Davulcu, and Y. Yang, "Memebot: Towards automatic image meme generation," *arXiv preprint arXiv:2004.14571*, 2020.

[74] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proc. of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, Association for Computational Linguistics, 2003, pp. 173–180.

[75] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[76] C. H. Suryanto, J.-H. Xue, and K. Fukui, "Randomized time warping for motion recognition," *Image and Vision Computing*, vol. 54, pp. 1–11, 2016.

List of publications

1. Erica K. Shimomoto, François Portet, Kazuhiro Fukui ,“Text classification based on the word subspace representation”, Pattern Analysis and Applications, vol.24, issue 3, pp. 1075–1093, 2021.

2. Erica K. Shimomoto, Lincon S. Souza, Bernardo B. Gatto, Kazuhiro Fukui,“News2meme: An Automatic Content Generator from News Based on Word Subspaces from Text and Image”, Proc. Sixteenth IAPR International Conference on Machine Vision Applications (MVA 2019), pp. 1-6, 2019.

3. Erica K. Shimomoto, Lincon S. Souza, Bernardo B. Gatto, Kazuhiro Fukui,“Text Classification Based On Word Subspace With Term-Frequency”, Proc. 2018 International Joint Conference on Neural Networks (IJCNN18), pp. 1-8, 2018.