University of Padua

Department of Mathematics 'Tullio Levi-Civita'

Master Thesis in Data Science

Unveiling Biases in Word Embeddings: An Algorithmic Approach for Comparative Analysis Based on Alignment

Supervisor
Professor Giovanni Da San Martino
University of Padua

Co-supervisor
Professor Francesco Rinaldi
University of Padua

Master Candidate
Pietro Maria Sanguin

Academic Year
2022-2023

To my dear parents, who with love, support and dedication have guided my journey of growth and education

Abstract

Word embeddings are vectorial representation of words with the goal of preserving semantic similarity. They are the state-of-the-art representations of machine learning algorithms applied to natural language. They are the result of specific learning algorithms trained on usually large corpora. Consequently, they inherit all biases of the corpora on which they have been trained on. The goal of the project is to devise an efficient algorithm to compare two different word embeddings in order to automatically highlight the biases they are subjected to. Specifically, we look for an alignment between the two vector spaces, corresponding to the two word embeddings, that minimises the difference between the stable words, i.e. the ones that have not changed in the two embeddings, thus highlighting the differences between the ones that did changed. In this work, we provide an efficient implementation to run the alignment algorithm over multiple cores in a HPC framework, specifically using SLURM and then we test our technique on a corpus of text taken from Italian newspapers in order to automatically identify which words are more subject to change among the different pairs of corpora.

Contents

Listing of figures

Listing of tables

Listing of acronyms

CBOW . . . . . Continuous Bag-Of-Words

SGNS . . . . . Skip-gram with Negative Sampling

NLP . . . . . Natural Language Processing

MSE . . . . . Mean Squared Error

SVD . . . . . Singular Value Decomposition

CSLS . . . . . Cross domain Similarity Local Scaling

LLM . . . . . Large Language Model

AI . . . . . Artificial Intelligence

ML . . . . . Machine Learning

OOV . . . . . Out Of Vocabulary

FW . . . . . Frank-Wolfe

Introduction

In the pursuit of harnessing the power of word embeddings for natural language processing, this thesis has explored a critical and urgent concern: the pervasive biases that sometimes can hide in the very foundations of these linguistic representations. It has become evident from our research that while word embeddings stand as formidable tools, they are far from impervious to the biases entrenched within the corpora upon which they are forged. Our primary objective throughout this project has been to craft an effective algorithm capable of scrutinizing and uncovering these biases by aligning the vector spaces of distinct word embeddings. As we navigate through the outcomes of our method, applied to real-world data sources like newspaper articles in Italian. Our experiments are aimed at searching for gender, racial and political bias in Italian newspapers. Various Italian newspaper sources were considered, including openly right-wing or left-wing sources. The results do not always mirror our initial expectations as experimenters, leaving us to confront a complex interplay of factors. These factors span from the subjective categorization of newspapers as either left or right-leaning, to the precise selection of words under scrutiny. This intricacy underscores the multifaceted nature of bias in natural language, demanding nuanced approaches for its detection and mitigation. Within the realm of the Italian language, our study offers a compelling glimpse into the differential susceptibility of certain words to bias. Through the scrutiny of embeddings across various corpora, we identify words like 'negro,' 'africano,' and 'schiavo' as striking examples. In right-leaning newspapers, these words exhibit pronounced discriminatory associations, while their left-leaning counterparts showcase a conscious avoidance of such biased terminology. In our commitment to advancing the field and fostering transparency, we have laid the foundation for future research endeavors. We introduce the 'Bias On Newspapers' (BONs), a shared and open-source GitHub repository. This repository serves as a valuable resource for the meticulous analysis of bias within monolingual texts, the alignment of embeddings through the use of the MUSE library, and the replication of similar experiments in diverse linguistic contexts. However, it is crucial to acknowledge the intricate nature of our alignment algorithm, adapted from linear programming. While we had anticipated more straightforward alignment results for words commonly associated with bias, various factors have introduced complexity into the process. These include potential misclassifications of texts as left or right-leaning, the constraints imposed by

the limited resulting vocabulary (comprising approximately 4300 words), and the underlying intricacies inherent in bias detection. As we chart the course for future investigations, several avenues beckon us. Expanding the scope of our inquiry to encompass a broader spectrum of words promises a more comprehensive grasp of bias in diverse contexts. The exploration of alternative similarity measures, such as CSLS, holds the potential to yield more nuanced insights into bias detection, mitigating some of the limitations inherent in cosine similarity.

Moreover, it is imperative that we subject Italian embeddings to rigorous evaluation against established benchmarks and assess their performance across a spectrum of linguistic tasks. Such a rigorous evaluation not only serves to validate the efficacy of our alignment algorithm but also extends its applicability to a multitude of languages and contexts.

In summation, this thesis stands as a contribution to the ongoing discourse surrounding bias within word embeddings. While challenges persist in refining and enhancing the alignment process, our research represents a critical stride towards the creation of fairer and more equitable AI systems. It is a testament to the enduring commitment to unraveling the intricacies of bias in language and fostering a future where technology and language coexist harmoniously.

1

Word Embeddings: A Deep Dive into Semantic Representations

The term word embedding denotes the vectorial representation of words made up of particularly dense vectors. This expression is widely used in the field of NLP. Mathematically [5], it can be seen as a mapping e from a set of N words also called a vocabulary $\mathcal{D} = \{w_i\}_{i=i,\dots N}$, to a d-dimensional vector space $\mathcal{V} = \{\vec{w}_i\}_{i=i,\dots N}$.

$$e : \mathcal{D} \xrightarrow{w \mapsto e(w) = \vec{w}} \mathcal{V} \tag{1.1}$$

As a result, $\mathcal{V} \cong \mathbb{R}^d$, where d can take different values: the most common are 50,100,200,300. Usually, $\mathcal{V} \cong \mathbb{R}^{300}$, since 300 is the most commonly used dimensionality in various studies [6]. Word embeddings are built on vector semantics create a bridge between two situations that look different: words and vectors. It is worth mentioning that the idea of embedding words into a vector space is not recent, as shown in the paper [7] of 1973 where a group of cognitive psychologists proposed a theory of analogical reasoning in which the elements of a set of concepts, e.g., animals, are represented as points in a multidimensional Euclidean space. They then empirically tested it on human beings evaluating the validity of the model. In recent years, due to the availability of data and the improvement of computational resources, an efficient algorithm was proposed in order create this representation of word starting from digitally written text.

1.1 Text Representations and Lexical Ambiguity

Text Representations To understand text, the starting point is learning its representations. Leveraging the existing text sequences from large corpora, self-supervised learning has been extensively used to pretrain text representations. One of the aims was to predict some hidden part of the text using some other part of their surrounding text. In this way, models learn through supervision from massive text data without expensive labeling efforts. All this theoretical idea has its roots in the distributional hypothesis which will be inquired in section 1.2. As we will see, when treating each word or subword as an individual token, the representation of each token can be pretrained using embedding algorithms such as: word2vec, GloVe, or subword embedding models (fastText). After pre-training, the representation of each token is encrypted in a vector, however, it remains the same no matter what the context is. For example, the vector representation of 'bank' is the same in both 'go to the bank to deposit some money' and 'go to the bank to sit down' [8]. Thus, many more recent pre-training models adapt the representation of the same token to different contexts. Among them there are GPT and BERT, much deeper self-supervised models based on the Transformer encoder, however these models will not be take into account in our work. In this chapter, we will focus on how to pre-train such representations for text, as highlighted in figure 1.1. Before going on, it is important to highlight some terminology.

Lexical Ambiguity refers to the property of a word or phrase having multiple possible meanings or interpretations. It encompasses the broader concept of words or phrases being associated with more than one distinct meaning, whether due to *homonymy* (same form, different meaning), *polysemy* (related meanings), *homophony* (same sound, different meaning), or *homography* (same spelling, different meaning). These linguistic phenomena elucidate the inherent complexity and multifaceted nature of language. Since all these language nuances are relevant to our analysis, we recap briefly their linguistic definitions with the help of examples [9].

Homonyms are words which are either *homographs* or *homophones*, or both. Using this definition, the words row (propel with oars), row (a linear arrangement) and row (an argument) are homonyms because they are homographs (though only the first two are homophones): so are the words see (vision) and sea (body of water), because they are homophones (though not homographs). These homonymous instances underscore the importance of context in distinguishing between divergent meanings.

Polysemes are words with the same spelling and distinct but related meanings. The distinction between polysemy and homonymy is often subtle and subjective, and not all sources con-

sider polysemous words to be homonyms. Words such as mouth, meaning either the orifice on one's face or the opening of a cave or river, are polysemous and may or may not be considered homonyms. Polysemy and homonymy are traditionally described in the context of paradigmatic lexical relations. Unlike monosemy, in which one meaning is associated with one form, and unlike synonymy, in which one meaning is associated with several forms, in polysemy and homonymy several meanings are associated with one form.

It is essential to acknowledge that certain lexical items may exhibit an amalgamation of these linguistic phenomena, as their usage and context interplay to confer distinct meanings.

1.2 The Distributional Hypothesis and

The distributional hypothesis is an idea coming from the research field of distributional semantics which is a part of linguistic. Distributional semantics develops and studies theories and methods for quantifying and categorizing semantic similarities between linguistic items based on their distributional properties in large samples of language data. The distributional hypothesis can be sum up to the fact that: *'words that are used and occur in the same contexts tend to purport similar meanings'* [10]. J. R. Firth was one of the firsts drawing attention to the context-dependent nature of meaning with his notion of 'context of situation', and his work on collocational meaning is widely acknowledged in the field of distributional semantics. In particular, he is known for the famous quotation: 'You shall know a word by the company it keeps' [11]. This hypothesis underlies the architectures of the models used to describe, classify and generate 'human language'. One of the first models taking inspiration from this hypothesis, but used when the embedding technique was not still spread, is the N-Gram model which estimate the probability of the last word of a phrase/expression given the previous words, and also to assign probabilities to entire sequences. These models that assign probabilities to sequences of words are called language models or LMs [12]. Another fundamental idea underpinning NLP models that increases the scope of the distributional hypothesis is the monolingual invariance.

Monolingual Invariance, indeed, is the hypothesis that certain NLP models or techniques should work consistently and effectively across different monolingual languages. In other words, it's the concept that the performance or behavior of an NLP system should remain relatively stable and reliable when applied to different languages individually, without requiring language-specific tuning or customization. As we will see in 2, there are some constraints in order to preserve monolingual invariance while stretching the embeddings during the translation.

1.3 The Word2vec Algorithm

Natural language is a complex system used to express meanings. In this system, words are the basic unit of the meaning. As the name implies, word vectors are vectors used to represent words, and can also be considered as feature vectors or representations of words. The technique of mapping words to real vectors is called word embedding. In recent years, word embedding has gradually become the basic knowledge of natural language processing. Now we focus specifically on the so-called pre-trained text representations, as shown in Figure 1.1 and we analyze the technical details of the word2vec algorithm. The resulting embeddings can be fed to a variety of deep learning architectures for different downstream natural language processing applications. In the big picture of NLP applications, we are now focusing on the pre-training part.

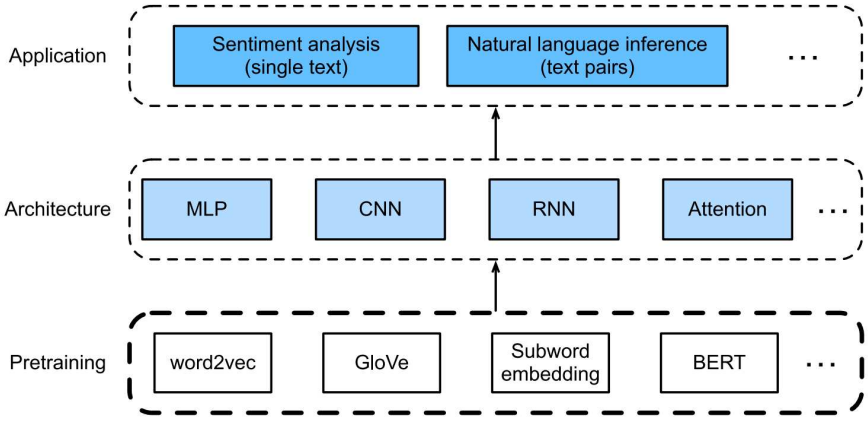


Figure 1.1: How pre-trained text representations can be fed to various deep learning architectures for different downstream natural language processing applications, taken from [8]

Why One-Hot Vectors Are a Bad Choice

The idea of encapsulating word meaning into vectors was initially proposed in the form of one-hot encoding. A one-hot encoding is a vector whose length is given by the size of the vocabulary N, where all entries are set to zero, except for the entry corresponding to our token, which is set to one. In this way one-hot vectors represent words and characters are words. To make it more concrete, suppose that the number of different words in the dictionary (the dictionary size) is N, and each word corresponds to a different integer (index) from 0 to $N-1$. To obtain the one-hot vector representation for any word with index i, we create a length N vector with all

zeros and set the element at position i to one. In this way, each word is represented as a vector of length N.

Although one-hot word vectors are easy to construct, they are usually not a good choice. A main reason is that one-hot word vectors cannot accurately express the similarity between different words, as the cosine similarity does. Since the cosine similarity between one-hot vectors of any two different words is 0, one-hot vectors cannot encode similarities among words. This fundamental property of the embeddings is further discussed in the next section 1.3.1.

1.3.1 Semantic and Syntactic Properties of Embeddings

Cosine Similarity Measure In a vector space, beyond the usual euclidean distance, it is possible to define the measure of cosine similarity, which, based on the way we compute word embeddings (see Eq. 1.4), highlights both the distributional and the semantic similarity between two vectors. Let $w_t, w_c \in \mathcal{D}$ be two words and let $\mathbf{u}, \mathbf{v} \in \mathcal{V}$ be the corresponding vectors. The cosine similarity between $\mathbf{u}$ and $\mathbf{v}$ is the cosine of the angle between the vectors, and it can be computed as:

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|_2 \|\mathbf{v}\|_2} \in [-1, 1] \tag{1.2}$$

When two words are similar, the cosine is larger, which implies that the angle between them is smaller. For example, considering a pre-trained embedding given by 50-dimensional GloVe word vectors taken from [4] we have that:

$$\cos(\overrightarrow{\text{father}}, \overrightarrow{\text{mother}}) = 0.89$$
$$\cos(\overrightarrow{\text{ball}}, \overrightarrow{\text{crocodile}}) = 0.27$$

Indeed, the words father and mother are semantically more similar than ball and crocodile.

Analogy is another semantic property of embeddings worth mentioning which is able to capture relational meanings. In the seminal work 'A Model for Analogical Reasoning' [7], previously cited by Rumelhart and Abrahamson, it is also shown that sets of concepts, represented by points in a multidimensional Euclidean space, can be related by analogical relationships. These analogies can be solved using the parallelogram model through simple vectorial operations. In more recent times, this model is applied to pretrained embeddings as word2vec [3] or GloVe vectors [4] and turned out to be very effective in bringing out relations of different nature among words.

Definition If we consider three words a, b, c the analogy is defined as $a : b = c : x$ (a is to b as c is to x) where the analogy's solution x is computed solving:

$$\text{argmin}_x \, d(\vec{x}, \vec{b} - \vec{a} + \vec{c}) \tag{1.3}$$

where d is a distance [12]. Given the optimal result for d, we can also write $\vec{b} - \vec{a} + \vec{c} \approx \vec{x}$. Let us consider again the pretrained word embedding given by 50-dimensional GloVe word vectors. We can observe:

$$\overrightarrow{\text{woman}} - \overrightarrow{\text{man}} + \overrightarrow{\text{son}} \approx \overrightarrow{\text{daughter}}$$

The analogy in this case captures the semantic relation of the pair (man, woman) which is of the type (male, female), and transposes it on new words.

$$\overrightarrow{\text{China}} - \overrightarrow{\text{Beijing}} + \overrightarrow{\text{Tokyo}} \approx \overrightarrow{\text{Japan}}$$

Now, the semantic relation represented by this operation is of the type (country, capital).

$$\overrightarrow{\text{worst}} - \overrightarrow{\text{bad}} + \overrightarrow{\text{big}} \approx \overrightarrow{\text{biggest}}$$

On the other hand, in this last example, it is possible to observe the interpolation of the syntactic relation of the type (basic form, superlative).

This property is particularly useful in many applications related to text bias and debiasing techniques. Bolukbasi et al. in [13] show that using word embeddings for simple analogies surfaces many gender stereotypes. For example, the word embedding they use, word2vec embedding trained on the Google News dataset, answer the analogy: 'man is to computer programmer as woman is to x' with 'x = homemaker'. However, they also introduced a metric (Eq. 3.1) to evaluate bias when identifying the gender subspace. Caliskan et al. further demonstrate association between female/male names and groups of words stereotypically assigned to females/males (e.g. arts vs. science).[14].

The scope of our word will be to analyse bias in language (c.f. 3), therefore debiasing techniques will not be investigated.

1.3.2 Self-Supervised word2vec

The word2vec tool was proposed to express the similarity among different words. This technique maps each word to a fixed length vector, and these vectors can better express the similarity and analogy relationship among different words. The word2vec tool contains two models, namely skip-gram and continuous bag of words (CBOW) [3]. For semantically meaningful representations, their training relies on conditional probabilities that can be viewed as predicting some words using some of their surrounding words in corpora following the intuition of the distributional hypothesis. Since supervision comes from the data without labels, both skip-gram and continuous bag of words are self-supervised models. This algorithm surpasses, in terms of efficiency and semantic/syntactic accuracy, other previous methodologies such as the neural network language model (NNLM), the Latent Semantic Analysis (LSA) and the Latent Dirichlet Allocation (LDA) [3]. From now on we will say that word vectors are a parametrization for text words.

In the following two models and their training methods are introduced.

1.3.3 The Skip-Gram Model

The skip-gram model assumes that a word can be used to generate its surrounding words in a text sequence. Take the text sequence 'the', 'man', 'loves', 'his', 'son' as an example. Choosing 'loves' as the center word and set the context window hyperparameter size to 2. As shown in figure 1.2, given the center word 'loves', the skip-gram model considers the conditional probability for generating the context words: 'the', 'man', 'his', and 'son', which are no more than two words away from the center word:

$$P(\text{ 'the', 'man', 'his', 'son' } | \text{ 'loves'}).$$

Assuming that the context words are independently generated given the center word (i.e., conditional independence). In this case, the above conditional probability can be rewritten as

$$P(\text{ 'the' } | \text{ 'loves' }) \cdot P(\text{ 'man' } | \text{ 'loves'}) \cdot P(\text{ 'his' } | \text{ 'loves' }) \cdot P(\text{ 'son' } | \text{ 'loves'}).$$

In the skip-gram model, each word has two d-dimensional-vector representations for calculating conditional probabilities. More concretely, for any word with index i in the dictionary, denote by $\mathbf{v}_i \in \mathbb{R}^d$ and $\mathbf{u}_i \in \mathbb{R}^d$ its two vectors when used as a center word and a context word, respectively. The conditional probability of generating any context word w_o (with index o in

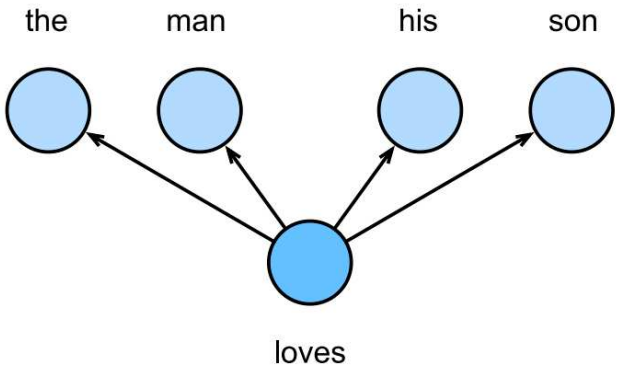


Figure 1.2: Graphical example of the skip-gram model, taken from [8]

the dictionary) given the center word w_c (with index c in the dictionary) can be modeled by a softmax operation on vector dot products:

$$P\left(w_o \mid w_c\right) = \frac{\exp\left(\mathbf{u}_o^\top \mathbf{v}_c\right)}{\sum_{i \in \mathcal{V}} \exp\left(\mathbf{u}_i^\top \mathbf{v}_c\right)} \tag{1.4}$$

where the vocabulary index set is $\mathcal{V} = \{0, 1, \ldots, |\mathcal{V}| - 1\}$. Given a text sequence of length T the word at time step t is denoted as $w^{(t)}$. For context window of size m, the likelihood function of the skip-gram model is the probability of generating all context words given any center word and assuming that context words are independently generated given any center word we have:

$$\prod_{t=1}^{T} \prod_{-m \leq j \leq m, j \neq 0} P\left(w^{(t+j)} \mid w^{(t)}\right) \tag{1.5}$$

where any time step that is less than 1 or greater than T can be omitted.

Training The skip-gram model parameters are the center word vector and context word vector for each word in the vocabulary. In training,the model parameters are learnt by maximizing the likelihood function (i.e., maximum likelihood estimation). This is equivalent to minimizing the following loss function:

$$-\sum_{t=1}^{T} \sum_{-m \leq j \leq m, j \neq 0} \log P\left(w^{(t+j)} \mid w^{(t)}\right) \tag{1.6}$$

Optimization techniques like stochastic gradient descent can be used to minimize the loss

and update the model parameters. The gradients of the log conditional probability with respect to the center word vector and the context word vector are calculated according to 1.4 and obtained:

$$\log P\left(w_o \mid w_c\right) = \mathbf{u}_o^\top \mathbf{v}_c - \log\left(\sum_{i\in\mathcal{V}} \exp\left(\mathbf{u}_i^\top \mathbf{v}_c\right)\right) \tag{1.7}$$

Through differentiation, the gradient with respect to the center word vector $\mathbf{v}_C$ results in:

$$\begin{aligned}\frac{\partial \log P\left(w_o \mid w_c\right)}{\partial \mathbf{v}_c} &= \mathbf{u}_o - \frac{\sum_{j\in\mathcal{V}} \exp\left(\mathbf{u}_j^\top \mathbf{v}_c\right) \mathbf{u}_j}{\sum_{i\in\mathcal{V}} \exp\left(\mathbf{u}_i^\top \mathbf{v}_c\right)} \\ &= \mathbf{u}_o - \sum_{j\in\mathcal{V}} \left(\frac{\exp\left(\mathbf{u}_j^\top \mathbf{v}_c\right)}{\sum_{i\in\mathcal{V}} \exp\left(\mathbf{u}_i^\top \mathbf{v}_c\right)}\right) \mathbf{u}_j \\ &= \mathbf{u}_o - \sum_{j\in\mathcal{V}} P\left(w_j \mid w_c\right) \mathbf{u}_j.\end{aligned} \tag{1.8}$$

This calculations requires the conditional probabilities of all words in the dictionary with w_c as the center word. The gradients for the other word vectors can be obtained in the same way.

After training, for any word with index i in the dictionary, we obtain both word vectors $\mathbf{v}_i$ (as the center word) and $\mathbf{u}_i$ (as the context word). In NLP applications, the center word vectors of the skip-gram model are typically used as the word representations. To sum up the skip-gram model considers the conditional probability of generating the surrounding context words given a center word.

1.3.4 The Continuous Bag of Words Model

The major difference between the skip-gram model and the CBOW is that the CBOW model assumes that a center word is generated based on its surrounding context words in the text sequence. For example, in the same text sequence 'the', 'man', 'loves', 'his', and 'son', with 'loves' as the center word and the context window size being two, the CBOW model considers the conditional probability of generating the center word 'loves' based on the context words 'the', 'man', 'his' and 'son' which is

$$P(\text{ 'loves' } \mid \text{ 'the', 'man', 'his', 'son'}).$$

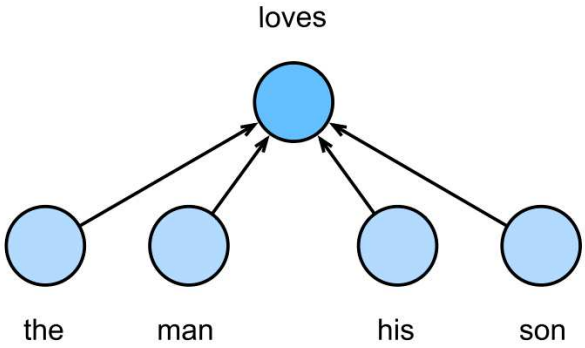


Figure 1.3: Graphical example of CBOW model, taken from [8]

Since there are multiple context words in the CBOW model, these context word vectors are averaged in the calculation of the conditional probability. Specifically, for any word with index i in the dictionary, denote by $\mathbf{v}_i \in \mathbb{R}^d$ and $\mathbf{u}_i \in \mathbb{R}^d$ its two vectors when used as a context word and a center word (meanings are switched in the skip-gram model), respectively. The conditional probability of generating any center word w_c (with index c in the dictionary) given its surrounding context words $w_{o_1}, \ldots, w_{o_{2m}}$ (with index $o_1, \ldots, o_{2m}$ in the dictionary) can be modelled by:

$$P\left(w_c \mid w_{o_1}, \ldots, w_{o_{2m}}\right) = \frac{\exp\left(\frac{1}{2m}\mathbf{u}_c^\top \left(\mathbf{v}_{o_1} + \ldots + \mathbf{v}_{o_{2m}}\right)\right)}{\sum_{i \in \mathcal{V}} \exp\left(\frac{1}{2m}\mathbf{u}_i^\top \left(\mathbf{v}_{o_1} + \ldots + \mathbf{v}_{o_{2m}}\right)\right)}. \tag{1.9}$$

For brevity, let $\mathcal{W}_o = \{w_{o_1}, \ldots, w_{o_{2m}}\}$ and $\overline{\mathbf{v}}_o = \left(\mathbf{v}_{o_1} + \ldots + \mathbf{v}_{o_{2m}}\right)/(2m)$ Then it results in:

$$P\left(w_c \mid \mathcal{W}_o\right) = \frac{\exp\left(\mathbf{u}_c^\top \overline{\mathbf{v}}_o\right)}{\sum_{i \in \mathcal{V}} \exp\left(\mathbf{u}_i^\top \overline{\mathbf{v}}_o\right)} \tag{1.10}$$

Given a text sequence of length T, where the word at time step t is denoted as $w^{(t)}$. For context window size m, the likelihood function of the continuous bag of words model is the probability of generating all center words given their context words:

$$\prod_{t=1}^{T} P\left(w^{(t)} \mid w^{(t-m)}, \ldots, w^{(t-1)}, w^{(t+1)}, \ldots, w^{(t+m)}\right) \tag{1.11}$$

Training Training CBOW is almost the same as training skip-gram models. The maximum likelihood estimation of the continuous bag of words model is equivalent to minimizing

the following loss function:

$$-\sum_{t=1}^{T} \log P\left(w^{(t)} \mid w^{(t-m)}, \ldots, w^{(t-1)}, w^{(t+1)}, \ldots, w^{(t+m)}\right) \tag{1.12}$$

Notice that:

$$\log P\left(w_c \mid \mathcal{W}_o\right) = \mathbf{u}_c^\top \overline{\mathbf{v}}_o - \log\left(\sum_{i \in \mathcal{V}} \exp\left(\mathbf{u}_i^\top \overline{\mathbf{v}}_o\right)\right) \tag{1.13}$$

Through differentiation, we can obtain its gradient with respect to any context word vector $\mathbf{v}_{o_i} \quad (i = 1, \ldots, 2m)$ as:

$$\frac{\partial \log P\left(w_c \mid \mathcal{W}_o\right)}{\partial \mathbf{v}_{o_i}} = \frac{1}{2m}\left(\mathbf{u}_c - \sum_{j \in \mathcal{V}} \frac{\exp\left(\mathbf{u}_j^\top \overline{\mathbf{v}}_o\right)\mathbf{u}_j}{\sum_{i \in \mathcal{V}} \exp\left(\mathbf{u}_i^\top \overline{\mathbf{v}}_o\right)}\right) = \frac{1}{2m}\left(\mathbf{u}_c - \sum_{j \in \mathcal{V}} P\left(w_j \mid \mathcal{W}_o\right)\mathbf{u}_j\right). \tag{1.14}$$

The gradients for the other word vectors can be obtained in the same way. Unlike the skip-gram model, the continuous bag of words model typically uses context word vectors as the word representations.

1.3.5 Approximate Training

The main idea of the skip-gram model is using softmax operations to calculate the conditional probability of generating a context word w_o based on the given center word w_c, Eq. (1.4). Due to the nature of the softmax operation, since a context word may be anyone in the dictionary $\mathcal{V}$, the opposite of eq. 1.7 contains the summation of items as many as the entire size of the vocabulary. Consequently, the gradient calculation for the skip-gram model and that for the continuous bag-of-words model both contain the summation. Unfortunately, the computational cost for such gradients that sum over a large dictionary (often with hundreds of thousands or millions of words) is huge.

In order to reduce the aforementioned computational complexity, this section will introduce two approximate training methods: negative sampling and hierarchical softmax. Due to the similarity between the skip-gram model and the CBOW, just the skip-gram model is taken as an example to describe these two approximate training methods.

Negative Sampling

Negative sampling modifies the original objective function. Given the context window of a center word w_c, the fact that any (context) word w_o comes from this context window is considered as an event with the probability modeled by

$$P\left(D=1 \mid w_c, w_o\right)=\sigma\left(\mathbf{u}_o^\top \mathbf{v}_c\right) \tag{1.15}$$

where σ uses the definition of the sigmoid:

$$\sigma(x)=\frac{1}{1+\exp(-x)} \tag{1.16}$$

Let's begin by maximizing the joint probability of all such events in text sequences to train word embeddings. Specifically, given a text sequence of length T, denote by $w^{(t)}$ the word at time step t and let the context window size be of size m, consider maximizing the joint probability:

$$\prod_{t=1}^{T} \prod_{-m \leq j \leq m, j \neq 0} P\left(D=1 \mid w^{(t)}, w^{(t+j)}\right) \tag{1.17}$$

However, this formula only considers those events that involve positive examples. As a result, this joint probability is maximized to 1 only if all the word vectors are equal to infinity. Of course, such results are meaningless. To make the objective function more meaningful, negative sampling adds negative examples sampled from a predefined distribution.

Denote by S the event that a context word w_o comes from the context window of a center word w_c. For this event involving w_o, from a predefined distribution $P(w)$ sample K noise words that are not from this context window. Denote by N_k the event that a noise word $w_k(k=1, \ldots, K)$ does not come from the context window of w_c. If these events involving both the positive example and negative examples $S, N_1, \ldots, N_K$ are mutually independent, then negative sampling rewrites the joint probability (involving only positive examples) as:

$$\prod_{t=1}^{T} \prod_{-m \leq j \leq m, j \neq 0} P\left(w^{(t+j)} \mid w^{(t)}\right) \tag{1.18}$$

where the conditional probability is approximated through events $S, N_1, \ldots, N_K$:

$$P\left(w^{(t+j)} \mid w^{(t)}\right) = P\left(D=1 \mid w^{(t)}, w^{(t+j)}\right) \prod_{k=1, w_k \sim P(w)}^{K} P\left(D=0 \mid w^{(t)}, w_k\right) \quad (1.19)$$

Denote by i_t and h_k the indices of a word $w^{(t)}$ at time step t of a text sequence and a noise word w_k, respectively. The logarithmic loss with respect to the previous conditional probabilities is:

$$\begin{aligned}
-\log P\left(w^{(t+j)} \mid w^{(t)}\right) &= -\log P\left(D=1 \mid w^{(t)}, w^{(t+j)}\right) - \sum_{k=1, w_k \sim P(w)}^{K} \log P\left(D=0 \mid w^{(t)}, w_k\right) \\
&= -\log \sigma\left(\mathbf{u}_{i_{t+j}}^\top \mathbf{v}_{i_t}\right) - \sum_{k=1, w_k \sim P(w)}^{K} \log\left(1-\sigma\left(\mathbf{u}_{h_k}^\top \mathbf{v}_{i_t}\right)\right) \\
&= -\log \sigma\left(\mathbf{u}_{i_{t+j}}^\top \mathbf{v}_{i_t}\right) - \sum_{k=1, w_k \sim P(w)}^{K} \log \sigma\left(-\mathbf{u}_{h_k}^\top \mathbf{v}_{i_t}\right).
\end{aligned} \quad (1.20)$$

We can see that now the computational cost for gradients at each training step has nothing to do with the dictionary size, but linearly depends on K. When setting the hyperparameter K to a smaller value, the computational cost for gradients at each training step with negative sampling is smaller.

Hierarchical Softmax

As an alternative approximate training method, hierarchical softmax uses the binary tree, where each leaf node of the tree represents a word in dictionary $\mathcal{V}$.

Denote by $L(w)$ the number of nodes (including both ends) on the path from the root node to the leaf node representing word w in the binary tree. Let $n(w, j)$ be the j^{th} node on this path, with its context word vector being $\mathbf{u}_{n(w,j)}$. For example, $L\left(w_3\right) = 4$ in Fig. 1.4. Hierarchical softmax approximates the conditional probability in Eq. (1.4) as:

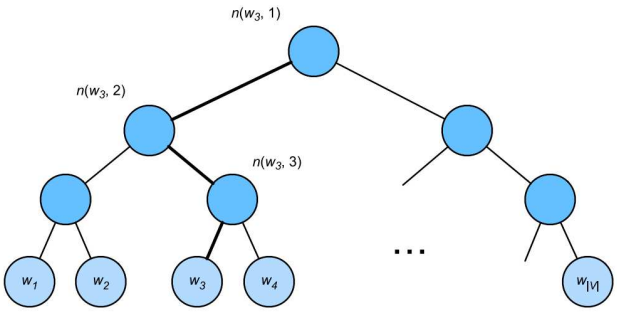


Figure 1.4: Hierarchical softmax for approximate training, where each leaf node of the tree represents a word in the dictionary, taken from [8]

$$\begin{cases} P\left(w_o \mid w_c\right) = \prod_{j=1}^{L(w_o)-1} \sigma\left(\mathbf{u}_{n(w_o,j)}^{\top} \mathbf{v}_c\right) & \text{if } n\left(w_o, j+1\right) = \text{leftChild}\left[n\left(w_o, j\right)\right] \\ P\left(w_o \mid w_c\right) = \prod_{j=1}^{L(w_o)-1} \sigma\left(-\mathbf{u}_{n(w_o,j)}^{\top} \mathbf{v}_c\right) & \text{otherwise} \end{cases} \tag{1.21}$$

where σ is the sigmoid, and leftChild (n) is the left child node of node n.

For example calculating the conditional probability of generating word w_3 given word w_c requires dot products between the word vector $\mathbf{v}_c$ of w_c and non-leaf node vectors on the path (bold path in figure) from the root to w_3, which is traversed left, right, then left:

$$P\left(w_3 \mid w_c\right) = \sigma\left(\mathbf{u}_{n(w_3,1)}^{\top} \mathbf{v}_c\right) \cdot \sigma\left(-\mathbf{u}_{n(w_3,2)}^{\top} \mathbf{v}_c\right) \cdot \sigma\left(\mathbf{u}_{n(w_3,3)}^{\top} \mathbf{v}_c\right) \tag{1.22}$$

Since $\sigma(x) + \sigma(-x) = 1$, it holds that the conditional probabilities of generating all the words in dictionary $\mathcal{V}$ based on any word w_c sum up to one: $\sum_{w \in \mathcal{V}} P\left(w \mid w_c\right) = 1$. Fortunately, since $L\left(w_o\right) - 1$ is on the order of $\mathcal{O}\left(\log_2 |\mathcal{V}|\right)$ due to the binary tree structure, when the dictionary size $\mathcal{V}$ is huge, the computational cost for each training step using hierarchical softmax is significantly reduced compared with that without approximate training.

1.3.6 Advantages and Disadvantages between CBOW and SG

In [6] these models are also been trained on Google News corpora. As a future reference here are reported the results of the Skip-gram and CBOW trained descrasing linearly the learning rate so that it approaches zero at the end of training. It is worth mentioning that training a model on twice as much data using one epoch gives comparable or better results than iterating over the same data for three epochs, as is shown in Table 1.1, and provides additional small

speedup.

Model	Vector Dim.	Training words	Accuracy [%]			Training [days]
			Semantic	Syntactic	Total	
3 epoch CBOW	300	783M	15.5	53.1	36.1	1
3 epoch Skip-gram	300	783M	50.0	55.9	53.3	3
1 epoch CBOW	300	783M	13.8	49.9	33.6	0.3
1 epoch CBOW	300	1.6 B	16.1	52.6	36.1	0.6
1 epoch CBOW	600	783M	15.4	53.3	36.2	0.7
1 epoch Skip-gram	300	783M	45.6	52.2	49.2	1
1 epoch Skip-gram	300	1.6 B	52.2	55.1	53.8	2
1 epoch Skip-gram	600	783M	56.7	54.5	55.5	2.5

Table 1.1: Comparison of models trained for three epochs on the same data and models trained for one epoch. The CBOW model was trained on subset of the Google News data. Results taken from [3]

As shown in Table 1.1 the Skip-gram produces a much higher accuracy on semantic level if compared to the CBOW. In syntax this does not hold necessarily. Moreover Skipgram works well with small amount of data and is found to represent rare words well. On the other hand, CBOW is faster and has better representations for more frequent words. In every case a higher leads to a bigger time consumption.

1.4 Word Embedding with Global Vectors (GloVe)

Word-word co-occurrences within context windows may carry rich semantic information. For example, in a large corpus word 'solid' is more likely to co-occur with 'ice' than 'steam', but word 'gas' probably co-occurs with 'steam' more frequently than 'ice'. Besides, global corpus statistics of such co-occurrences can be precomputed: this can lead to more efficient training. To leverage statistical information in the entire corpus the SG model can be modified using global corpus statistics such as co-occurrence counts.

1.4.1 Skip-Gram with Global Corpus Statistics

Denoting by q_{ij} the conditional probability $P\left(w_j \mid w_i\right)$ of word w_j given word w_i in the skip-gram model (Eq. 1.4) we have:

$$q_{ij} = \frac{\exp\left(\mathbf{u}_j^\top \mathbf{v}_i\right)}{\sum_{k \in \mathcal{V}} \exp\left(\mathbf{u}_k^\top \mathbf{v}_i\right)} \tag{1.23}$$

where for any index i vectors $\mathbf{v}_i$ and $\mathbf{u}_i$ represent word w_i as the center word and context word, respectively, and $\mathcal{V}$ is the index set of the vocabulary. Consider word w_i that may occur multiple times in the corpus. In the entire corpus, all the context words wherever w_i is taken as their center word form a multiset C_i of word indices that allows for multiple instances of the same element . For any element, its number of instances is called its multiplicity.

Now let's denote the multiplicity of element j in multiset C_i as x_{ij}. This is the global cooccurrence count of word w_j (as the context word) and word w_i (as the center word) in the same context window in the entire corpus. Using such global corpus statistics, the loss function of the SG model is equivalent to:

$$-\sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} x_{ij} \log q_{ij} \tag{1.24}$$

We further denote by x_i the number of all the context words in the context windows where w_i occurs as their center word, which is equivalent to $|C_i|$. Letting p_{ij} be the conditional probability x_{ij}/x_i for generating context word w_j given center word the previous formula can be rewritten as:

$$-\sum_{i \in \mathcal{V}} x_i \sum_{j \in \mathcal{V}} p_{ij} \log q_{ij} \tag{1.25}$$

In (1.25), $-\sum_{j \in \mathcal{V}} p_{ij} \log q_{ij}$ calculates the cross-entropy of the conditional distribution p_{ij} of global corpus statistics and the conditional distribution q_{ij} of model predictions. This loss is also weighted by x_i as explained above. Minimizing this loss function will allow the predicted conditional distribution to get close to the conditional distribution from the global corpus statistics.

1.4.2 The GloVe Model

With this in mind, the GloVe model makes three changes to the SG model based on squared loss [4]:

1. Use variables $p'_{ij} = x_{ij}$ and $q'_{ij} = \exp\left(\mathbf{u}_j^\top \mathbf{v}_i\right)$ that are not probability distributions and take the logarithm of both, so the squared loss term is $\left(\mathbf{u}_j^\top \mathbf{v}_i - \log x_{ij}\right)^2$.

2. Add two scalar model parameters for each word w_i : the center word bias b_i and the context word bias c_i.

3. Replace the weight of each loss term with the weight function $h\left(x_{ij}\right)$, where $h(x)$ is increasing in the interval of $[0, 1]$.

Putting all things together, training GloVe is to minimize the following loss function:

$$\sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} h\left(x_{ij}\right)\left(\mathbf{u}_j^{\top} \mathbf{v}_i + b_i + c_j - \log x_{ij}\right)^2 \tag{1.26}$$

For the weight function, a suggested choice is: $h(x) = (x/c)^{\alpha}$(e.g. $\alpha = 0.75$) if $x < c$ (e.g. $c = 100$); otherwise $h(x) = 1$. When $h(0) = 0$, the squared loss term for any $x_{ij} = 0$ can be omitted for computational efficiency. For example, when using minibatch stochastic gradient descent for training, at each iteration we randomly sample a minibatch of non-zero x_{ij} to calculate gradients and update the model parameters.

It is worth mentioning that $x_{ij} = x_{ji}$ since word appearing in context is a symmetric relation. Unlike word2vec that fits the asymmetric conditional probability p_{ij}, GloVe fits the symmetric $\log x_{ij}$. Therefore, the center word vector and the context word vector of any word are, in the GloVe model, mathematically equivalent. However in practice, owing to different initialization values, the same word may still get different values in these two vectors after training: GloVe sums them up as the output vector.

1.4.3 Interpreting GloVe from the Ratio of Co-occurrence Probabilities

The GloVe model can also be interpreted from another perspective as shown in [4]. Let $p_{ij} \stackrel{\text{def}}{=} P\left(w_j \mid w_i\right)$ be the conditional probability of generating the context word w_j given w_i as the center word in the corpus. Table 1.2 lists several co-occurrence probabilities given words 'ice' and 'steam' and their ratios based on statistics from a large corpus.

$w_k =$	solid	gas	water	fashion
$p_1 = P(w_k \mid \text{ice})$	0.00019	0.000066	0.003	0.000017
$p_2 = P(w_k \mid \text{steam})$	0.000022	0.00078	0.0022	0.000018
p_1/p_2	8.9	0.085	1.36	0.96

Table 1.2: Word-word co-occurrence probabilities and their ratios from a large corpus (adapted from Table 1 in [4])

It can be seen that the ratio of co-occurrence probabilities can intuitively express the relationship between words. Thus, we can design a function of three word vectors to fit this ratio. For the ratio of co-occurrence probabilities p_{ij}/p_{ik} with w_i being the center word and w_j and w_k being the context words, we want to fit this ratio using some function f :

$$f\left(\mathbf{u}_j, \mathbf{u}_k, \mathbf{v}_i\right) \approx \frac{p_{ij}}{p_{ik}} \tag{1.27}$$

Since the ratio of co-occurrence probabilities is a scalar, it is required f to be a scalar function, such as $f\left(\mathbf{u}_j, \mathbf{u}_k, \mathbf{v}_i\right) = f\left(\left(\mathbf{u}_j - \mathbf{u}_k\right)^\top \mathbf{v}_i\right)$. Switching word indices j and k in 1.27, it must hold that $f(x)f(-x) = 1$, so one possibility is $f(x) = \exp(x)$, i.e.:

$$f\left(\mathbf{u}_j, \mathbf{u}_k, \mathbf{v}_i\right) = \frac{\exp\left(\mathbf{u}_j^\top \mathbf{v}_i\right)}{\exp\left(\mathbf{u}_k^\top \mathbf{v}_i\right)} \approx \frac{p_{ij}}{p_{ik}} \tag{1.28}$$

Now we pick $\exp\left(\mathbf{u}_j^\top \mathbf{v}_i\right) \approx \alpha p_{ij}$, where α is a constant. Since $p_{ij} = x_{ij}/x_i$, after taking the logarithm on both sides we get $\mathbf{u}_j^\top \mathbf{v}_i \approx \log \alpha + \log x_{ij} - \log x_i$. We may use additional bias terms to fit $-\log \alpha + \log x_i$, such as the center word bias b_i and the context word bias c_j :

$$\mathbf{u}_j^\top \mathbf{v}_i + b_i + c_j \approx \log x_{ij} \tag{1.29}$$

Measuring the squared error of Eq. (1.29) with weights, the GloVe loss function in Eq. (1.26) is obtained.

1.5 Subword Embedding

Morphology, an essential branch of linguistic analysis, scrutinizes the internal structure and form of words. By exploring the manner in which words are constructed and modified via affixation, such as prefixes and suffixes, morphology uncovers the intricacies of word formation. For instance, if we consider the word 'unhappiness', it comprises the morphemes 'un-' (a prefix indicating negation), 'happy' (the base word), and '-ness' (a suffix indicating a state or quality) . The intricate interplay of these morphemes within the word construction sheds light on the morphological intricacies of language. In this perspective the relationship between 'dog' and 'dogs' is the same as that between 'cat' and 'cats', and the relationship between 'boy' and 'boyfriend' is the same as that between 'girl' and 'girlfriend'. In other languages such as French and Spanish, many verbs have over 40 inflected forms, while in Finnish, a noun may have up

to 15 cases [15]. In linguistics, morphology studies word formation and word relationships. However, the internal structure of words was neither explored in word2vec nor in GloVe.

1.5.1 The FastText Model

Recall how words are represented in word2vec. In both the skip-gram model and the continuous bag-of-words model, different inflected forms of the same word are directly represented by different vectors without shared parameters. To use morphological information, the fastText model proposed a subword embedding approach, where a subword is a character n-gram [15]. The number of n-gram to be extracted is a parameter of the model and usually a number between 2 and 6 works well in many applications. Instead of learning word-level vector representations, fastText can be considered as the subword level skip-gram, where each center word is represented by the sum of its subword vectors.

Now we illustrate how to obtain subwords for each center word in fastText using , for example, the word 'where'. First, add special characters '<' and '>' at the beginning and end of the word to distinguish prefixes and suffixes from other subwords. Then, extract character n-grams from the word. For example, when $n = 3$, we obtain all subwords of length 3: '<wh', 'whe', 'her', 'ere', 're>', and the special void subword ''.

To describe the math behind fastText, for any word w, we denote by $\mathcal{G}_w$ the union of all its subwords of length between 2 and 6 and its special subword. The vocabulary is the union of the subwords of all words. Notice that even a relatively small vocabulary can be blown up when n is small. Letting $\mathbf{z}_g$ be the vector of subword g in the dictionary, the vector $\mathbf{v}_w$ for word w as a center word in the skip-gram model is the sum of its subword vectors:

$$\mathbf{v}_w = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g \tag{1.30}$$

The rest of fastText is the same as the skip-gram model. Compared with the skip-gram model, as mentioned the vocabulary in fastText is larger, resulting in more model parameters. Besides, to calculate the representation of a word, all its subword vectors have to be summed, leading to higher computational complexity. However, thanks to shared parameters from subwords among words with similar structures, rare words and even out-of-vocabulary words may obtain better vector representations in fastText.

Byte Pair Encoding

In fastText, all the extracted subwords have to be of the specified lengths, such as 3 to 6 , thus the vocabulary size cannot be predefined. To allow for variable-length subwords in a fixedsize vocabulary, we can apply a compression algorithm called byte pair encoding (BPE) to extract subwords [16].

Byte pair encoding performs a statistical analysis of the training dataset to discover common symbols within a word, such as consecutive characters of arbitrary length. Starting from symbols of length 1 , byte pair encoding iteratively merges the most frequent pair of consecutive symbols to produce new longer symbols. Note that for efficiency, pairs crossing word boundaries are not considered. In the end, we can use such symbols as subwords to segment words. Byte pair encoding and its variants has been used for input representations in popular natural language processing pretraining models such as GPT-2 [17].

2

Word-to-Word Alignment Techniques and Algorithms

This chapter investigates the various methods and algorithms used for aligning the dense words' representations. These techniques, as the name suggests, align word embeddings in order to establish correspondences between words in different corpora and was first designed for machine translation. The aim of this work is to use this methods to search correspondences between monolingual corpora and to test if this methodology is able to detect 'bias' at word level.

Mikolov et al. in [18] first observe that word embedding models trained on different corpora (in their specific case, corpora of distinct languages) exhibite similar geometric patterns and behaviors as shown in Figure 2.1. This intuition leads to the hypothesis for which word embedding spaces can be transformed from one to another through linear operations. Starting from embeddings e_1 and e_2 independently trained on different corpora, the goal is to construct the linear map A by finding the transformation matrix $\mathbf{W}$:

$$A : \mathcal{V}_1 \xrightarrow{\vec{x} \mapsto \mathbf{W}\vec{x} \approx \vec{y}} \mathcal{V}_2 \tag{2.1}$$

where $\vec{x} = e_1(x)$ and $\vec{y} = e_2(y)$ are representations of corresponding words x and y, and $\mathbf{W} \in \mathbb{R}^{d \times d}$ is the so called translation matrix. If the alignment is used to build cross-lingual embedding models x and y are each the translation of the other, while if the alignment acts on embedding spaces of the same language, x and y should be, semantically, the same word with

different representations.

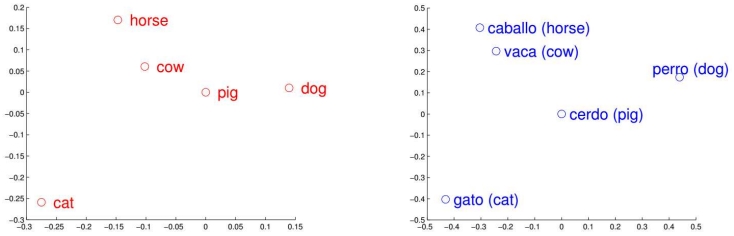


Figure 2.1: Distributed word vector representations of animals in English (left) and Spanish (right). The five vectors in each language were projected down to two dimensions using PCA, and then manually rotated to accentuate their similarity. It can be seen that these concepts have similar geometric arrangements in both spaces, suggesting that it is possible to learn an accurate linear mapping from one space to another. Source: [18].

In the machine translation framework, fixed points are typically established by using parallel corpora, which are collections of texts in multiple languages that have been translated sentence by sentence or word by word. By aligning the embeddings of words appearing in the parallel corpora, a mapping between the source and target language embeddings can be established. However, a parallel corpora is not always required, as shown in [1] where a fully unsupervised method is proposed in the adversarial learning framework. An intuition of the general alignment strategy is given in Figure 2.2.

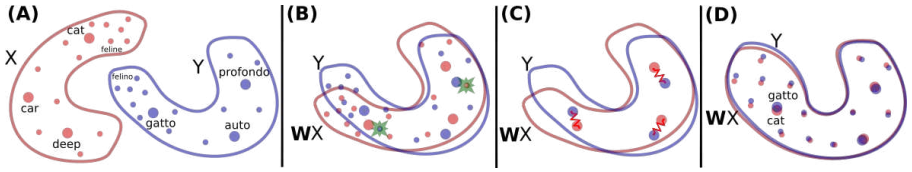


Figure 2.2: Intuition of the alignment method, taken from [1]

In Figure 2.2 (A) there are two distributions of word embeddings, in particular English words in red denoted by X and Italian words in blue denoted by Y, which we want to align/translate. Each dot represents a word in that space. Then a rotation matrix W is learned by adversarial learning and refined by Procrustes method. Finally, the translation is done using W mapping and a distance metric, called CSLS (Cross domain Similarity Local Scaling), which expands the space where there is a high density of points (such as the area around the word 'cat'), so that 'hubs' (like the word 'cat') become less close to other word vectors than they would otherwise (compare to the same region in panel (A)). Clearly, this intuition does not give all the mathematical details of the methodology which can be either supervised or unsupervised.

Once the alignment is established, cross-lingual word translation becomes feasible. By leveraging aligned word embeddings, words from the source language can be mapped to their counterparts in the target language. This enables the translation of words or even entire sentences, facilitating cross-lingual communication, information retrieval, machine translation systems, and, as we expect to discover during this work, also cultural bias.

Taking into account the embeddings e_1 and e_2 defined as:

$$e_1 : \mathcal{D}_1 \to \mathcal{V}_1 \quad e_2 : \mathcal{D}_2 \to \mathcal{V}_2 \tag{2.2}$$

$\mathcal{D}_1$ and $\mathcal{D}_2$ are different vocabularies and $\mathcal{V}_1 \cong \mathbb{R}^{d_1}$ and $\mathcal{V}_2 \cong \mathbb{R}^{d_2}$ are the vector spaces where the words are embedded. Usually, the embedded spaces considered have the same dimensions, hence we denote $d_1 = d_2 = d$, and in our future experiments we will take this as a core assumption. If there exists a correspondence between elements $x \in \mathcal{D}_1$ and $y \in \mathcal{D}_2$, then the alignment between their respective embedding spaces seeks to find a map A as in Eq. (2.1). The scope of these methodology in the machine translation framework is the so called bilingual lexicon induction.

Bilingual lexicon induction is the intrinsic task that is most commonly used to evaluate current cross-lingual word embedding models. Briefly, given a list of N language word forms $w_1^s, \ldots, w_N^s$, the goal is to determine the most appropriate translation w_i^t, for each query form w_i^s. This is commonly accomplished by finding a target language word whose embedding $\mathbf{x}_i^t$ is the nearest neighbour to the source word embedding $\mathbf{x}_i^s$ in the shared semantic space, where similarity is usually computed as the cosine similarity between their embeddings.

There exists seven types of alignment: Word-to-Word, Sentence-to-Sentence, Word-to-Sentence, Sentence-to-Sentence, Knowledge-to-Knowledge, Word-to-Knowledge, Sentence-to-Knowledge, as shown in [5].

For the scope of this work we will focus on Word-to-Word alignment and their paradigms. Three methods that use parallel word-aligned data are distinguished as: mapping-based approaches, approaches based on pseudo-multilingual corpora, and joint methods. To make this classification clearer the definitions are:

a **Mapping-based approaches** that first train monolingual word representations independently on large monolingual corpora and then seek to learn a transformation matrix that maps representations in one language to the representations of the other language.

b **Pseudo-multi-lingual corpora-based approaches** that use monolingual word embedding methods on automatically constructed (or corrupted) corpora that contain words from both the source and the target language.

c **Joint methods** that take parallel text as input and minimize the source and target language monolingual losses jointly with the cross-lingual regularization term.

It was shown that these approaches, modulo optimization strategies and hyper-parameters, are nevertheless often equivalent [19]. For this reason our work will focus on the mapping-based approaches.

Mapping-based approaches are by far the most prominent category of cross-lingual word embedding models and, due to their conceptual simplicity and ease of use are currently the most popular. Mapping-based approaches aim to learn a mapping from the monolingual embedding spaces to a joint cross-lingual space. Approaches in this category differ along multiple dimensions:

1. The mapping method that is used to transform the monolingual embedding spaces into a cross-lingual embedding space.
2. The seed lexicon that is used to learn the mapping.
3. The refinement of the learned mapping.
4. The retrieval of the nearest neighbors.

Then there are four types of mapping methods that have been proposed:

1. **Regression methods** map the embeddings of the source language to the target language space by maximizing their similarity.
2. **Orthogonal methods** map the embeddings in the source language to maximize their similarity with the target language embeddings, but constrain the transformation to be orthogonal.
3. **Canonical methods** map the embeddings of both languages to a new shared space, which maximizes their similarity.
4. **Margin methods** map the embeddings of the source language to maximize the margin between correct translations and other candidates.

However as we will prove in section 2.1.4, the first three methods, under some contraints, can be reduced to the same framework.

Word-to-Word alignment techniques can be supervised and unsupervised depending on the requirement of parallel data. We will further study this two methods in the next sections. Note that also, semi-supervised methods exist; for further reference, see [20].

Moreover, two other cases are worth mentioning: Sentence (or Sentence to-Sentence) Alignment and Document Alignment models. Sentence Alignment models often serve as an entry point to machine translation applications and rely on a parallel corpus of sentences in two languages [5]. Indeed, also sentences can be mapped to vectors of real numbers through sentence embedding. Finally, Document Alignment Models require documents in different languages that are translations of each other, which is very rare.

The main applications of alignment are crosslingual embedding models, which are cross-lingual representations of words in a joint embedding space. They can be used to facilitate cross-lingual transfer, which consists of modelling on data from one language and then applying it to another relying on shared cross-lingual features, for some NLP tasks. An example among these cross-lingual tasks is sentiment analysis, namely the task of determining the sentiment polarity (e.g. positive and negative) of texts in different languages [19].

2.1 Mapping-based Supervised Methods

Supervised learning methods are the most common and data intensive in machine learning applications. In this section, we discuss supervised models that use unsupervised features (embeddings) as inputs with the goal of aligning these resources. Most of the alignment methods are supervised and use a bilingual dictionary of a few thousand entries to learn the mapping. Existing approaches can be classified into four groups. Regression methods are the first adopted [18] and map the embeddings in one language using a least-squares objective that learns the linear transformation minimizing the sum of squared Euclidean distances for the dictionary entries. Other authors incorporated also L_2 regularization into this method [21]. Then there are the orthogonal methods, which map the embeddings in one or both languages under the constraint of the transformation being orthogonal [22]. Other techniques are called canonical methods, which map the embeddings in both languages to a shared space using canonical correlation analysis and extensions of it. This is usually done through Canonical Correlation Analysis (CCA) as first proposed by Faruqui and Dyer [23], who motivate their method as a way to improve the quality of monolingual embeddings using bilingual data.

However, Artetxe et al. in[22] shown the equivalence of different objective functions under orthogonality and different normalization procedures, and clarified that regression, orthogonal and canonical methods essentially differ on the constraints imposed on the mapping.

Margin methods, are the last of the four alignment techniques, which map the embeddings in one language to maximize the margin between the correct translations and the rest of the

candidates. This approach was proposed by Lazaridou et al. [24] as a way to address the hubness problem, with the addition of intruder negative sampling to generate more informative training examples.

2.1.1 Regression Method

Regression methods form the class of solutions first used to address the Word-to-Word embedding alignment problem. Let us begin by defining two corpora: $\mathcal{D}_1$ and $\mathcal{D}_2$, our source and target, respectively, and embedding functions $e_1 : \mathcal{D}_1 \rightarrow \mathbb{R}^d$ and $e_2 : \mathcal{D}_2 \rightarrow \mathbb{R}^d$. As shown in [18], using the n most frequent words $x_1, \ldots, x_n \in \mathcal{D}_1$ and the corresponding $y_1, \ldots, y_n \in \mathcal{D}_2$ as seed words, it is possible to learn a transformation matrix $\mathbf{W}$ using stochastic gradient descent by minimising the squared Euclidean distance (MSE) between the embedding vectors $e_1(x_i)$ and $e_2(y_i)$ [19]. Namely, the function to minimize in this case to find $\mathbf{W} \in \mathbb{R}^{d \times d}$ is defined as:

$$\Omega_{REG}(\mathbf{R}) = \sum_{i=1}^{n} \|\mathbf{R}e_1(x_i) - e_2(y_i)\|^2 \text{ with } \mathbf{R} \in \mathbb{R}^{d \times d} \tag{2.3}$$

Finally, it holds:

$$\mathbf{W} = \underset{\mathbf{R} \in \mathbb{R}^{d \times d}}{\arg\min} \Omega_{REG}(\mathbf{R}) \tag{2.4}$$

Consequently, W will be the so-called leastsquares solution of the linear matrix equation $WX = Y$. This is a well-known problem in linear algebra and can be solved by taking the MoorePenrose pseudoinverse $X^+ = \left(X^T X\right)^{-1} X^T$ as $W = X^+ Y$, which can be computed using SVD. In these case the linear transformation is learned from the source language into the target language, but Shigeto et al. in [21] argue that it is better to map the target language into the source language as a way to address the hubness problem.

Hubness [25] is a phenomenon observed in high-dimensional spaces where some points (known as hubs) are the nearest neighbours of many other points. As translations are assumed to be nearest neighbours in cross-lingual embedding space, hubness has been reported to affect cross-lingual word embedding models. This method was first proposed as a mean of capturing geometric patterns between embeddings across embedding spaces. In the original paper, no additional pre-processing is done on the input word vectors, which were generated using the CBOW word2vec algorithm. The transformation matrix W can then be applied to a new vector $e_1\left(w_N\right)$ to map it into the target space where a cosine similarity search can rank all trans-

lation candidates. Subsequent papers suggested minor tweaks to the regression model having significant impacts on the capacity to learn. These include the addition of L2 regularization and adding pre-processing steps such as embedding vector unit normalization [26].

2.1.2 Orthogonal Method

The original regression model utilized a euclidean distance in learning the transformation matrix, yet relies on cosine similarity to carry out similarity searches in the target space. This 'inconsistency', can be modified adding unit length normalization to the source and target vector spaces and constrain the matrix $\mathbf{W}$ to be orthogonal, that is $\mathbf{W}^\top\mathbf{W} = \mathbf{I}$ where $\mathbf{I}$ is the identity matrix [26]. The pre-processing step and orthogonal constraint then line up with the retrieval method, where we are less concerned with distances between vectors and more concerned with the angles between them. This problem is equivalent to the orthogonal Procrustes Problem, given by:

$$\mathbf{W} = \underset{\mathbf{R}\in\mathbb{R}^{d\times d}}{\arg\min}\|\mathbf{RX} - \mathbf{Y}\|_F \text{ subject to } \mathbf{R}^T\mathbf{R} = I \tag{2.5}$$

where $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{d\times n}$ are the embedding matrices of the seed words, whose columns of the same index $\mathbf{X}^i$ and $\mathbf{Y}^i$ with $i = 1 \ldots n$ are the embedding vectors of two corresponding words x_i and y_i.

Schönemann in [27] find out that the exact solution to this problem can be efficiently computed in linear time with respect to the vocabulary size using SVD:

$$\mathbf{W} = \mathbf{V}\mathbf{U}^T \quad \text{where} \quad \mathbf{Y}^T\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^T \tag{2.6}$$

where Σ is a diagonal matrix in the SVD, and its diagonal entries are the singular values of the matrix being decomposed.

The orthogonal constraints have been motivated in different ways from various authors in [19] is underlined that these constraints lead to the preservation of length normalization, while Artetxe et al. in [22] motivate orthogonality as a means to ensure monolingual invariance. The authors also show that combining normalization, feature whitening helped them achieve superior performance when using CBOW word vectors and 5,000 supervised training examples. The full framework has been packaged and released as open source code under the moniker VecMap*.

*The source code can be found at https://github.com/artetxem/vecmap

2.1.3 Canonical Method

This class of methods is based on the idea of mapping the embeddings in a shared space using Canonical Correlation Analysis (CCA). In [23] there is the first attempt to apply CCA to project words from two languages into a shared embedding space. The innovative aspect of this model is that, instead of finding only one transformation matrix $\mathbf{W}$ which projects the source embedding space $\mathcal{V}_1$ into the target embedding space $\mathcal{V}_2$, it is able to learn two transformation matrices $\mathbf{W}^{s\rightarrow}$ and $\mathbf{W}^{t\rightarrow}$ projecting respectively the embedding spaces $\mathcal{V}_1$ and $\mathcal{V}_2$ into a new joint space which is neither the source nor the target. Relying on the same notation as before, let $e_1(x) = \vec{x}$ and $e_2(y) = \vec{y}$ be the representations of two corresponding words x and y, and let $\mathbf{R}^1, \mathbf{R}^2 \in \mathbb{R}^{d\times d}$ be the possible transformation matrices candidates. The correlation between the projections $\mathbf{R}^1\vec{x}$ and $\mathbf{R}^2\vec{y}$ can be defined as:

$$\rho\left(\mathbf{R}^1\vec{x}, \mathbf{R}^2\vec{y}\right) = \frac{\operatorname{cov}\left(\mathbf{R}^1\vec{x}, \mathbf{R}^2\vec{y}\right)}{\sqrt{\operatorname{var}\left(\mathbf{R}^1\vec{x}\right)\operatorname{var}\left(\mathbf{R}^2\vec{y}\right)}} \tag{2.7}$$

where $\operatorname{cov}(\cdot,\cdot)$ is the covariance and $\operatorname{var}(\cdot)$ is the variance. Then, the aim of CCA methos is to to maximize the function:

$$\Omega_{CCA}\left(\mathbf{R}^1, \mathbf{R}^2\right) = -\sum_{i=1}^{n}\rho\left(\mathbf{R}^1\vec{x}_i, \mathbf{R}^2\vec{y}_i\right) \tag{2.8}$$

which means that

$$\mathbf{W}^{s\rightarrow}, \mathbf{W}^{t\rightarrow} = \underset{\mathbf{W}^{s\rightarrow}, \mathbf{W}^{t\rightarrow}\in\mathbb{R}^{d\times d}}{\arg\max}\ \Omega_{CCA}\left(\mathbf{R}^1, \mathbf{R}^2\right) \tag{2.9}$$

where, as before, $\vec{x}_i$ and $\vec{y}_i$ are the columns of $\mathbf{X}$ and $\mathbf{Y}$.

However, as we will see in section 2.1.4, Artetxe, Labaka, and Agirre, in [22], demonstrate that various objective functions are equivalent when orthogonality and different normalization techniques are applied. They also explain that the distinction between regression, canonical, and orthogonal methods lies in the restrictions imposed on the mapping.

2.1.4 Equivalence of The Methods

Here we prove the equivalence of the regression, orthogonal and canonical methods under specified constraints, as previously shown in [22]. Let X and Y denote the word embedding matrices in two languages for a given bilingual dictionary so that their i th column X_{i*} and

Y_{i*} are the word embeddings of the i th entry in the dictionary. As shown in the regression method our goal is to find a linear transformation matrix W so that WX best approximates Y. We formalized it in Eq. (2.4) and know we can rewrite:

$$\arg\min_{W} \sum_{i} \|WX_{i*} - Y_{i*}\|^2 \tag{2.10}$$

Alternatively, this is equivalent to minimizing the (squared) Frobenius norm of the residual matrix:

$$\arg\min_{W} \|WX - Y\|_F^2 \tag{2.11}$$

Now we introduce the constraints under which the three methods previously mentioned are equivalent.

Orthogonality for monolingual invariance

Monolingual invariance can be obtained requiring W to be an orthogonal matrix $\left(W^T W = I\right)$. Indeed it is needed to preserve the dot products after mapping, avoiding performance degradation in monolingual tasks (e.g. analogy). The exact solution under such orthogonality constraint is given by $W = VU^T$, where $Y^T X = U\Sigma V^T$ is the SVD factorization of $Y^T X$ (cf. Appendix). Thanks to this, the optimal transformation can be efficiently computed in linear time with respect to the vocabulary size. Note that orthogonality enforces an intuitive property, and as such it could be useful to avoid degenerated solutions and learn better bilingual mappings, as we empirically shown in [22].

Length normalization for maximum cosine

Normalizing word embeddings in both languages to be unit vectors guarantees that all training instances contribute equally to the optimization goal. As long as W is orthogonal, this is equivalent to maximizing the sum of cosine similarities for the dictionary entries, which is commonly used for similarity computations:

$$\begin{aligned} &\arg\min_{W} \sum_{i} \left\| W \frac{X_{i*}}{\|X_{i*}\|} - \frac{Z_{i*}}{\|Z_{i*}\|} \right\|^2 \\ &= \arg\max_{W} \sum_{i} \cos\left(WX_{i*}, Z_{i*}\right) \end{aligned} \tag{2.12}$$

This last optimization objective coincides with the one of the ortogonal method proposed by [26], but their work was motivated by an hypothetical inconsistency in [18], where the optimization objective to learn word embeddings uses dot product, the objective to learn mappings uses Euclidean distance and the similarity computations use cosine. However, the fact is that, as long as W is orthogonal, optimizing the squared Euclidean distance of length-normalized embeddings is equivalent to optimizing the cosine, and therefore, the mapping objective proposed by [26] is equivalent to that used by [18] with orthogonality constraint and unit vectors. In fact, the experiments of [22] show that orthogonality is more relevant than length normalization.

Mean centering for maximum covariance

Dimension-wise mean centering captures the intuition that two randomly taken words would not be expected to be semantically similar, ensuring that the expected product of two random embeddings in any dimension and, consequently, their cosine similarity, is zero. As long as W is orthogonal, this is equivalent to maximizing the sum of dimensionwise covariance for the dictionary entries:

$$\begin{aligned} &\arg\min_{W} \|C_m W X - C_m Y\|_F^2 \\ &= \arg\max_{W} \sum_i \operatorname{cov}\left(W_{*i}X, Y_{*i}\right) \end{aligned} \tag{2.13}$$

where C_m denotes the centering matrix.

This equivalence reveals that the canonical method proposed by Faruqui and Dyer [23] is closely related to the regression method. More concretely, Faruqui and Dyer use Canonical Correlation Analysis (CCA) to project the word embeddings in both languages to a shared vector space. CCA maximizes the dimension-wise covariance of both projections (which is equivalent to maximizing the covariance of a single projection if the transformations are constrained to be orthogonal, but adds an implicit restriction to the two mappings, making different dimensions have the same variance and be uncorrelated among themselves:

$$\begin{aligned} &\arg\max_{A,B} \sum_i \operatorname{cov}\left(XA_{i*}, YB_{i*}\right) \\ &\text{s.t.} \quad A^T X^T C_m X A = B^T Y^T C_m Y B = I \end{aligned} \tag{2.14}$$

Therefore, the only fundamental difference between both methods is that, while the orthog-

onal model enforces monolingual invariance, the canonical method do change the monolingual embeddings to meet this restriction. In this regard, as shown in [22] the added restriction has a negative impact on the learning of the bilingual mapping, and it degrades the quality of the monolingual embeddings.

2.1.5 Margin Methods

The methods of the previous two sections rely on variations of mean squared error to compute and learn from the differences between the source and target space. An alternative modeling technique leverages a max-margin based loss function. These objectives seek to reward the weights associated with positive pairs (in this case, words that are direct translations) while reducing the signal from noise pairs generated either randomly or using a heuristic. This model, introduced in [24] is built to reduce hubness. Turning back to the model, in the case of Word-to-Word translation, the association between pairs is defined by their cosine similarity, thus we may define the max-margin loss as

$$\sum_{j \neq i}^{k} \max \left\{ 0, \gamma - \text{dist}\left(\hat{y}_i, \vec{w_i^t}\right) + \text{dist}\left(\hat{y}_i, \vec{w_j^t}\right) \right\} \tag{2.15}$$

where $\hat{y}_i = W\vec{w_i^s}$, dist is a distance measure and γ and k are tunable hyperparameters denoting the margin and the number of noise pairs (negative examples), respectively. The use of this objective was first proposed by [24] to address issues of hubness seen in regression and orthogonal techniques. The presence of hubs is driven by embeddings that dominate the space because of their high cosine similarity with all other vectors in the source or target space. These hubs can be caused by the overall frequencies of words in the underlying corpus [28], a common mean vector present in all word vectors causing issues of anisotropy in the embedding spaces [29], or issues derived from least-squares regression where low variance points are all grouped together in the target space.

Intuitively, the goal of the max-margin objective is to rank the correct translation y_i of $\vec{w}_i$ higher than any other possible translation y_j. In theory, the summation in the equation could range over all possible labels, but in practice this is too expensive (e.g., in our experiments the search space contains almost 200k candidate labels), and it is usually computed over just a portion of the label space.

Margin-based models are also explored in [30], where the authors also aim to combat the issue of hubs by introducing new retrieval criteria. The cross-domain similarity local scaling

(CSLS), which is defined as

$$\text{CSLS}(x, y) = -2\cos(x, y) + \frac{1}{k}\sum_{y' \in N_Y(x)} \cos(x, y') + \frac{1}{k}\sum_{x' \in N_X(y)} \cos(x', y) \tag{2.16}$$

where $N_Y(x)$ is the set of k nearest neighbors of x in the target space. The authors built this retrieval criterion into their margin model by using unpaired words (those with no explicit translation in the training set) as negative samples when computing nearest neighbors. The full objective function, called the relaxed CSLS (RCSLS) is then computed as

$$\frac{1}{n}\sum_{i=1}^{n} -2\cos\left(\hat{y}_i, \vec{w_i^t}\right) + \frac{1}{k}\sum_{w_j \in N_Y(\hat{y}_i)} \cos(\hat{y}_i, \vec{w_j}) + \frac{1}{k}\sum_{\hat{y}_j \in N_X(\vec{w}_i)} \cos(\hat{y}_j, \vec{w}_i) \tag{2.17}$$

As stated in [5]: 'margin-based methods are studied less frequently; we conjecture this is due to the difficulty in selecting informative negative samples and the preference for methods using as little supervision as possible.'

2.2 Unsupervised Methods

Under the goal of restricting the amount of parallel data needed to create alignment between two word spaces, several approaches have been proposed that attempt to leverage the structure of the embedding space itself, completely removing the need for parallel data. A key approach was described in [1] where the authors propose leveraging an adversarial learning paradigm. In this setup, the goal is still to learn a linear map W between the source embedding vectors $e_1(w_i^s)$ and target space embeddings $e_2(w_i^t)$. In order to do that, a discriminator D is trained to recognize and separate the mapped embeddings $We_1(w_i^s)$ from $e_2(w_i^t)$, while an adversarial generator G is trained to fool D. The loss functions given for both models are:

$$\mathcal{L}(\theta_D \mid W) = -\frac{1}{n}\sum_{i=1}^{n} \log P_{\theta_D}(\text{ source } = 1 \mid Wx_i) - \frac{1}{m}\sum_{i=1}^{m} \log P_{\theta_D}(\text{ source } = 0 \mid y_i) \tag{2.18}$$

for the discriminator model, and

$$\mathcal{L}\left(W \mid \theta_D\right) = -\frac{1}{n}\sum_{i=1}^{n} \log P_{\theta_D}\left(\text{ source } = 0 \mid Wx_i\right) - \frac{1}{m}\sum_{i=1}^{m} \log P_{\theta_D}\left(\text{ source } = 1 \mid y_i\right) \tag{2.19}$$

for the generator model. With an initial linear map W learned thanks to this adversarial learning approach, a refinement procedure is then then applied by identifying anchor points as pairs that were frequently identified as translations in the previous step. Anchor points and their corresponding word frequencies are used to solve the orthogonal Procrustes problem to generate a refined mapping matrix W^*. This final matrix is used in conjunction with the CSLS objective described in Eq. (2.16) to mitigate hubness and areas of density in generating a final translation from source to target. The adversarial method has been utilized to generate large benchmark datasets under the name of Multilingual Unsupervised or Supervised Embeddings (MUSE) [†], releasing parallel embedding spaces trained using FastText in 110 languages.

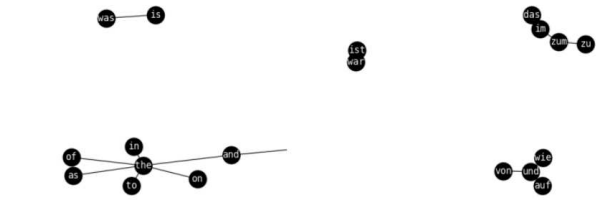
Aside from leveraging similarity distributions of the underlying embedding spaces, methods are also available to treat these embedding spaces as metric spaces, adopting mathematical tools from measure theory and topology to describe their nature. One such metric is the Gromov-Wasserstein distance used to compare two pairs of spaces, rather than the pairwise point-by-point metrics such as similarities. Using this metric, [31] transform the alignment problem to one of finding an optimal transport from source X to target Y. Due to computational costs, the problem is split into two steps where the two spaces are first aligned using the explicit optimization to find an optimal coupling followed by a refinement using an orthogonal Procrustes procedure [5].

Different languages, different graphs

In [2] Søgaard et al. identify the limitations of unsupervised machine translation, finding out that performances are generally worse when monolingual corpora from different domains or different embedding algorithms are used. Regarding this, their critique is based on the idea that: unsupervised approaches to learning crosslingual word embeddings are based on the assumption that monolingual word embedding graphs are approximately isomorphic, that is, after removing a small set of vertices (words). They use as a reference the state-of-the-art unsupervised model of Conneau et al. [1], which relies on an orthogonal alignment, and attempt to improve it. They first investigated the nearest-neighbor graphs of word embedding spaces

[†]The source code can be found at https://github.com/facebookresearch/MUSE

in order to conclude that, in general, monolingual word embeddings are far from isomorphic. This holds even if the two languages are closely related, such as English and German. In Figure 2.3 it is possible to see how different the nearest-neighbor graph of the embeddings of the 10 most frequent English words and the one of their translation in German. This could be due both to the differences between the synctatic structures and the different meaning associations between the two languages.



(a) Top 10 most frequent English words

(b) German translations of top 10 most frequent English words

Figure 2.3: A representation provided by Søgaard et al. [2] of the nearest neighbor graphs of 10 most frequent words in English Wikipedia and of their their automatic translation in German, by using the method of [1].

In our case this is not a problem since we are aligning two monolingual word embeddings of the same language: Italian.

3

Bias Definition and Detection

It is well known that standard machine learning can acquire stereotyped biases from textual data that reflect everyday human culture. Indeed it is now widely recognized that data-driven decision making is not per se safe from producing unfair or biased decisions, either for amplifications of biases already present in the data they learn from, or for algorithmic inaccuracies. As it is often the case with moral and ethical issues, conflicts are present between different and typically equally reasonable positions. Researches, in proposing definitions, have focused on different intuitive notions of 'unfair decisions', often considered as the ones impacting people in different ways on the basis of some personal characteristics, such as gender, ethnicity, age, sexual or political or religious orientations, considered to be protected, or sensitive. The general idea that text corpora capture semantics, including cultural stereotypes and empirical associations, has long been known in corpus linguistics. Usually the techniques adopted were word embeddings which are able to extract associations captured in text corpora; amplifying the signal found in raw statistics [14]. Nowadays most advanced techniques based on transformers are used, as shown in [32]. However, before moving on, it is important to focus on what we mean with bias.

Terminology varies by discipline and the term bias can be used with different meanings. In statistics, there is a narrow and specific definition of bias which can be summarized as lack of internal validity or incorrect assessment of the association between an exposure and an effect in the target population in which the estimated statistic has an expectation that does not equal the true value [33]. In this framework, biases can be classified by the research stage in which they

occur or by the direction of change in an estimate. As another example cognitive sciences has another broader definition, defining bias as: a systematic pattern of deviation from norm or rationality in judgment. In its most general sense, the term bias means simply 'slant'. Given this undifferentiated usage, at times the term is applied with relatively neutral content. A grocery shopper, for example, can be 'biased' by not buying damaged fruit. At other times, the term bias is applied with significant moral meaning. An employer, for example, can be 'biased' by refusing to hire minorities. In this work we focus on instances of the latter. In fact if one wants to develop criteria for judging the quality of systems in use which we do then criteria must be delineated in ways that speak robustly yet precisely to relevant social matters. Focusing on bias of moral import does just that. Accordingly, we use the term bias to refer to computer systems that systematically and unfairly discriminate against certain individuals or groups of individuals in favor of others [34]. In AI and ML systems, which are trained on real world data, bias refers generally to prior information, a necessary prerequisite for intelligent action. As mentioned above this kind of bias can be problematic when such information is derived from aspects of human culture known to lead to harmful behavior. Here, we will call such biases 'stereotyped' and actions taken on their basis 'prejudiced' [35]. Moreover we are concerned with media bias which refers to the bias produced when journalists report about an event in a prejudiced manner or with a slanted viewpoint. It is often related to content favoring a particular view-point or ideology (e.g., political). Media bias can have various negative impacts, e.g., the distribution of false facts, affecting decision-making processes, endangering the readers' trust in news and also affect downsteam applications propagating cultural stereotypes to artificial intelligence [36]. For example in another widely publicized study, Bertrand and Mullainathan [37] sent nearly 5000 identical résumés in response to 1300 job advertisements, varying only the names of the candidates. They found that European-American candidates were 50% more likely to be offered an opportunity to be interviewed. Vectorial representation of words are also used to train the LLM, but it was shown that a vectorial representation encapsulates any biases of the text on which it is trained. A striking example that pinpoint sexism implicit in text is found in [13] where the model complete the analogy 'man is to computer programmer as woman is to x' with x being homemaker. For this reason deploying these word embedding algorithms in practice, for example in automated translation systems or as hiring aids, runs the serious risk of perpetuating problematic biases in important societal contexts [38]. The declinations of this phenomenon, is also indicated with the term cultural semantic conditioning [39].

However 'bias' is not always so evident and there can be the risk to miss the point when searching for 'bias' without a normative motivation. In fact the authors of [35] surveyed 146

papers analyzing 'bias' in NLP systems. They found that in research papers focused on finding textual bias: 'the motivations are often vague, inconsistent, and lacking in normative reasoning'. They further find that these papers' proposed quantitative techniques for measuring or mitigating 'bias' which are poorly matched to their motivations and do not engage with the relevant literature outside of NLP. For this reason it is important to recognize the relationships between language and social hierarchies, what kinds of system behaviors are harmful, in what ways, to whom, and why, as well as the normative reasoning underlying these statements—and to center work around the lived experiences of members of communities affected by NLP systems, while interrogating and re-imagining the power relations between technologists and such communities. Regarding the definition of a measure and assess fairness in AI and ML systems, in the last few years an incredible number of definitions have been proposed, formalizing different perspectives from which to assess and monitor fairness in decision making processes. As reported in [40]: 'popular tutorial presented at the Conference on Fairness, Accountability, and Transparency in 2018 was titled 21 fairness definitions and their politics. The number has grown since then.' The proliferation of fairness definitions is not per se an issue: it reflects the evidence that fairness is a multi-faceted concept, and concentrates on itself different meanings and nuances, in turn depending in complex ways on the specific situation considered.

Fortunately in [32] a comprehensive benchmark that groups different types of media bias (e.g., linguistic, cognitive, political) under a common framework is provided. A unified benchmark encourages the development of more robust systems and ensure a systematic examination of a series of independent subcategories of media bias detection. These are the identified subcategories:

Linguistic bias comprehends all forms of bias induced by lexical features, such as word choice and sentence structure, often subconsciously used. Generally, linguistic bias is expressed through specific word choice that reflects the social-category cognition applied to any described group or individual(s).

Text-level context bias refers to the expression of a text's context, whereby words and statements can shape the context of an article and sway the reader's perspective. These biases can be used to portray a particular opinion in a biased way by criticizing one side more than the other, using inflammatory words, or omitting relevant information.

Reporting-level context bias refers to bias that arises through decisions made by editors and journalists on what events to report and what sources to use. While text-level context bias examines the bias present within an individual article, reporting-level bias focuses on systematic attention given to specific topics. **Cognitive bias** occurs when readers introduce bias by

selecting which articles to read and which sources to trust, which can be amplified on social media. These biases can lead to self-reinforcing cycles, echo chamber effect exposing readers to only one side of an issue.

Hate speech refers to any language that manifests hatred towards a specific group or aims to degrade, humiliate, or offend. Hate speech is usually induced by using linguistic bias. In particular, on social media, the impact of hate speech is significant and exacerbates tensions between the parties involved. However, similar processes can also be observed within, e.g., comments on news websites.

Racial bias is expressed through negative or positive portrayals of racial groups. Research has shown that racial bias in news coverage can severely impact affected minorities, such as strengthening stereotypes and discrimination [41].

Gender bias in media can manifest as discrimination against one gender through underrepresentation or negative portrayal. Gender bias in the media can severely impact perceptions of professions and role models [13].

Political bias refers to a text's political leaning or ideology, potentially influencing the reader's political opinion and, ultimately, their voting behavior [42]. There are several approaches to detect political bias in the media, for example, counting the appearance of certain political parties or ideology-associated words.

we would add to this list also the:

Religious bias which encompasses instances where language, media, or communication exhibit favoritism, prejudice, or discrimination toward particular religious groups or beliefs. Religious bias can have significant societal consequences, including the promotion of religious intolerance, the reinforcement of stereotypes, and the exacerbation of religious conflicts.

The exact overlap between media bias and fake news is yet unclear; therefore, we will not consider it as a bias subcategory. We emphasize that these subcategories are not comprehensive, but are intended to give a light in the immense sea of definitions. In [34] a framework is proposed for analyzing algorithmic biases, splitting them into pre-existing, technical, and emergent. Such subdivision is based on the phases of the learning system in which the bias arises: while the preexisting bias is related exclusively on the input data on which the algorithm is trained, technical bias is caused by technical constraints or technical considerations and emergent bias arises in a context of use with real users. Although word embeddings might face all three types of bias, we focus on the preexisting bias. The expression indicates the phenomenon in which AI systems embody biases that exist independently, exemplifying them. It has roots in social institutions, practices, and attitudes.

3.0.1 Geometry of Bias in Word Embeddings

Defining a measure of bias is not a one-size-fits-all process. In the literature, the measurement of bias is addressed individually for some of the previously illustrated subcategories and it is not straightforward to generalize. In order to review some of the existing techniques adopted in word embeddings bias measuring, we briefly review some notation we are going to use in this chapter. An embedding consists of a unit vector $\vec{w} \in \mathbb{R}^d$, with $\|\vec{w}\| = 1$, for each word (or term) $w \in W$. The normalization of word vectors is two scoped: improves alignment as shown in [43] and simplifies calculations shown later. Since gender bias is one of the most studied type of societal bias affecting NLP algorithms, due to implicit sexism permeating the society, we will start with that. An example of gender bias measure in word embeddings is given in [13] where they consider a set of gender neutral words $N \subset W$, such as fight attendant or shoes, and a set of F-M gender pairs $P \subset W \times W$, such as she-he or mother-father. Note that the size of a set S by $|S|$. In this way similarity between two vectors u and v can be measured by their cosine similarity : $\cos(u, v) = \frac{u \cdot v}{\|u\|\|v\|}$ as shown in section 1.3.1. This normalized similarity between vectors u and v is the cosine of the angle between the two vectors. Since words are normalized $\cos(\vec{w}_1, \vec{w}_2) = \vec{w}_1 \cdot \vec{w}_2$ In this way they input into the analogy generator a seed pair of words (a, b) determining a seed direction $\vec{a} - \vec{b}$ corresponding to the normalized difference between the two seed words. Then all pairs of words x, y are scored by the following metric:

$$S_{a,b}(x, y) = \begin{cases} \cos(\vec{a} - \vec{b}, \vec{x} - \vec{y}) & \text{if } \|\vec{x} - \vec{y}\| \leq \delta \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

where δ is a threshold for semantic similarity. The intuition of the scoring metric is that good analogy pair should be close to parallel to the seed direction while the two words are not too far apart in order to be semantically coherent. They found that $\delta = 1$ often works well in practice. Since all embeddings are normalized, this threshold corresponds to an angle $\leq \pi/3$, indicating that the two words are closer to each other than they are to the origin. In practice, it means that the two words forming the analogy are significantly closer together than two random embedding vectors. Given the embedding and seed words, the top analogous pairs with the largest positive $S_{(a,b)}$ scores can be output. The results obtained in the w2vNEWS dataset are shown in Figure 3.1.

Finally, the authors present a de-biasing algorithm which tries to correct the components of biased words by finding a gender subspace, given by two binary extremes among an axes relying on pairs of gender specific words.

Extreme *she*	Extreme *he*
1. homemaker	1. maestro
2. nurse	2. skipper
3. receptionist	3. protege
4. librarian	4. philosopher
5. socialite	5. captain
6. hairdresser	6. architect
7. nanny	7. financier
8. bookkeeper	8. warrior
9. stylist	9. broadcaster
10. housekeeper	10. magician

Gender stereotype *she-he* analogies

sewing-carpentry	registered nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	lovely-brilliant

Gender appropriate *she-he* analogies

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Figure 3.1: Left The most extreme occupations as projected on to the she-he gender direction on w2vNEWS. Occupations such as businesswoman, where gender is suggested by the orthography, were excluded. **Right** Automatically generated analogies for the pair she-he using the procedure described. Source: [13].

Racial bias is another important type of social discrimination which might emerge from word embeddings. In Semantics derived automatically from language corpora contain human-like biases [14] is shown that vectors representing African American names are closer to negative words than positive words, a trend which does not repeat among European American names. In order to define such associations, the authors use analogy as well. Similarly, on [37] are found occupational stereotypes related to embeddings of proper names depending on the ethnicity. In their experiments, bias detection is achieved by measuring distance among the elements a set of proper names and a set of occupations words [44].

Political bias is the most interesting part to deepen for our next analysis. In [45] Gordon et al. prove that it is possible to apply the same approach of the gender bias, mentioned above [13], in order to investigate the political bias contained in word embeddings. Their work models political bias as a binary choice between the Democratic pole and the Republican pole, since they refer to American politics. In order to apply the methodology of Bolukbasi et al. [13], the authors find binary pairs by analyzing the most frequent words of a collection of tweets of politicians of both parties. However, the pre-processing applied was heavy, as they removed the 10,000 most common words in English such as 'the', 'and', etc. Finally, they conclude that modelling bias along multiple axes or as a range of points along a single axis could be necessary to make a comprehensive analysis, since the politics, as many other fields, is much more complicated than a binary choice. Indeed neither gender nor political bias is as simple in the real world as two points on a single axis. The idea that we are going to apply follows this line; in fact, we are using data from Italian newspapers to model and describe bias without debiasing.

The previous fine-grain bias analysis can be summarized into a single definition: social bias. Social bias refers to systematic and often unconscious preferences, prejudices, or stereotypes that people hold toward individuals or groups based on characteristics such as race, gender,

age, religion, sexual orientation, disability, or other social identifiers. These biases can lead to unfair or discriminatory treatment, both at an individual and societal levels, affecting people's opportunities, experiences, and outcomes. Social bias can manifest in various forms, including implicit bias (unconscious attitudes and stereotypes) and explicit bias (conscious and deliberate prejudices), and it can impact decisions and behaviors in areas such as hiring, education, healthcare, and law enforcement. Addressing social bias is important for promoting fairness, equity, and social justice in society.

In our experiments, we are looking for social bias. Indeed, the presence of one or all of them is unfair since it strengthens discrimination and social polarization. We remind the reader that polarization is a process in which the normal multiplicity of differences in a society increasingly aligns along a single dimension and people increasingly perceive and describe politics and society in terms of 'Us' versus 'Them'. The politics and discourse of opposition and the social–psychological intergroup conflict dynamics produced by this alignment are a main source of the risks polarization generates for democracy, although we recognize that it can also produce opportunities for democracy [46].

For what concern our experiments, we report two lists of Italian words where testing our methodology. The lists are divided in words that are likely to be unbiased (Table 3.2) which can be used as parallel corpora during a supervised alignment, and biased words (Table 3.1) which are usually carriers of social bias, being it political, racial or gender; we retrieved them from [47], [39] [45] and our experience.

$\mathcal{B}$
indiano, africano, straniero, alieno , zingara, nigeriano, immigrato, negro, nero, schiavo, tribù, omosessuale, gay, lesbica, transessuale, diverso, uomo, donna padrone, vecchio, nuovo, operaio, comunista, fascista, nord, sud

Table 3.1: Test set of the words subject to societal bias

The idea behind Table 3.2, is that words belonging to the categories of inanimate objects and common animals and natural elements are less involved in cultural change over time.

$\mathcal{S}$
casa, albero, tavolo, lampada, libro, specchio, scatola, forchetta
sedia, telefono, bottiglia, fornello, motore, barca, matita, tazza
piatto, carta, stereo, foglia, bastone, nuvola, shampoo, cappello, finestra
cuscino, acqua, fuoco, libro, porta, strada, sentiero
gatto, cane, scuola, inchiostro, penna, osso, provviste, dizionario, ombrello, forbice

Table 3.2: Test set of the words likely to be stable

4

Proposed Alignment Methods

4.1 Introduction to Linear and Nonlinear Optimization Problems

In this part, we will review some fundamental principles of linear and nonlinear programming to provide a foundation for the subsequent sections, which will present a series of results related to nonlinear programming that will be beneficial in validating our alignment approach.

A linear program is a constrained optimization problem in which the function to be maximized (or minimized) is linear and the constraints are described by linear equations and/or inequalities:

$$\begin{aligned} \max \text{ (or min)} \quad & c_1 x_1 + \cdots + c_n x_n \\ \text{s.t.} \quad & a_{11} x_1 + \cdots + a_{1n} x_n \sim b_1 \\ & \vdots \\ & a_{m1} x_1 + \cdots + a_{mn} x_n \sim b_m \end{aligned} \tag{4.1}$$

where the symbol $\sim$ can stand for one of the operators $\leq, \geq, =$, and $a_{ij}, b_i, c_j \in \mathbb{R} \quad i \in [1, m], j \in [1, n]$. The set of all the vectors $x \in \mathbb{R}^n$ satisfying all the constrains, namely the feasible solutions, is called the feasible region. A vector $\bar{x} \in \mathbb{R}^n$ is an optimal solution of the linear program if $\bar{x}$ is feasible and $c^T \bar{x} \geq c^T x \quad \forall x$ in the feasible region, and the corresponding value $c^T \bar{x}$ is called optimal value.

Given a linear program, one and only one of the following alternatives holds:

a) The problem has at least one optimal solution;

b) The problem is infeasible, i.e., it has no feasible solutions;

c) The problem is unbounded, i.e., for every $K \in \mathbb{R}$, there exists a feasible solution x such that $c^T x > K$ (for minimization problems).

Every linear program of the form (4.1) is equivalent to a linear program in standard form, that is, of the type:

$$\begin{array}{ll} \max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array} \tag{4.2}$$

where $A \in \mathbb{R}^{m \times n}$, and $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$ and x is a vector of variable in $\mathbb{R}^n$

A problem which is not in one of the previous form is a nonlinear problem.

An optimization problem, in general, is defined as the minimization or maximization of a real valued function on a specified set. Letting $\mathcal{F}$ be the feasible set and $f : \mathcal{F} \to \mathbb{R}$ be the objective function, the problem can be represented as following:

$$\min f(x) \quad \text{s.t. } x \in \mathcal{F} \tag{4.3}$$

or

$$\max f(x) \quad \text{s.t. } x \in \mathcal{F} \tag{4.4}$$

However, it is important to notice that it is always possible to turn a minimization problem into a maximization problem by changing the sign of f, and viceversa. In particular, it holds:

$$\max_{x \in \mathcal{F}} f(x) = -\min_{x \in \mathcal{F}} (-f(x)) \tag{4.5}$$

For what concern nonlinear optimization problems, are now provided some results useful to understand the tools we use in section 4.4.

4.2 Linearization of nonlinear problems

Sometimes, it is possible to linearize a nonlinear problem, that is, to transform it into a linear one (of the form 4.1). Among all the examples of this useful operation, here we report just the linearization of absolute value problems, which is necessary to understand the computation we do in section 4.4.

Let us consider a problem of the following form:

$$\min_{x,y} \sum_j c_j |x_j| + \sum_k d_k y_k \quad \text{s.t. } (x, y) \in C \tag{4.6}$$

where C, the feasible region, is a polyhedron, and that $c_j \geq 0$. Now, we transform the variables in the following way:

$$x = x_j^+ - x_j^-, x_j^+ \geq 0, x_j^- \geq 0 \tag{4.7}$$

Such transformation is not unique, indeed there are two possible cases:

- $x_j \geq 0 \Rightarrow x_j^+ = x_j + \delta = |x_j| + \delta$ and $x_j^- = \delta$
- $x_j < 0 \Rightarrow x_j^+ = \delta$ and $x_j^- = -x_j + \delta = |x_j| + \delta$

with $\delta \geq 0$. When $\delta = 0$, one component must be 0 and the other one is consequently $|x_j|$. Moreover, we notice that:

$$x_j^+ + x_j^- = |x_j| + 2\delta$$

Now, by making a replacement in the objective function of term $|x_j|$ with the sum of $x_j^+$and x_j^-, we obtain that for the optimal solution $x_j^+ = 0$ or $x_j^- = 0$ (or, equivalently, $\delta = 0$). As a result, it is possible to rewrite 4.6 as:

$$\begin{aligned} \min_{x,y} \textstyle\sum_j c_j \left(x_j^+ + x_j^-\right) + \sum_k d_k y_k \quad \text{s.t.} \quad & \left(x_j^+ - x_j^-, y\right) \in C \\ & x_j^+ \geq 0, x_j^- \geq 0 \quad \forall j = 1, \ldots, n \end{aligned} \tag{4.8}$$

where C is the feasible region.

It is possible to obtain a different formulation, by simply rewriting the absolut value as:

$$|x_j| = \max\{x_j, -x_j\} \tag{4.9}$$

Then by replacing it in Eq. (4.6), we have:

$$\min_{x,y} \sum_j c_j \max\{x_j, -x_j\} + \sum_k d_k y_k \quad \text{s.t. } (x,y) \in C \tag{4.10}$$

which, by introducing a new variable z_j, can be rewrite as following:

$$\begin{aligned} \min_{x,y,z} \textstyle\sum_j c_j z_j + \sum_k d_k y_k \quad \text{s.t.} \quad & z_j = \max\{x_j, -x_j\} \\ & (x,y) \in C \end{aligned} \tag{4.11}$$

Finally, we obtain:

$$\min_{x,y,z} \sum_j c_j z_j + \sum_k d_k y_k \quad \text{s.t.} \quad -z_j \leq x_j \leq z_j \quad j = 1, \ldots, n \tag{4.12}$$

Example: Linearization of the MSE in Linear Regression

The linear model can be written as:

$$y = a^T x + b \tag{4.13}$$

where $x \in \mathbb{R}^n$ is the input vector for the model, $y \in \mathbb{R}$ is the output, $a \in \mathbb{R}^n$ an $b \in \mathbb{R}$ are the parameters related to the model. Moreover, we assume to have a finite set T of input-output samples, named training set:

$$T := \left\{ \left(x^1, y^1\right), \ldots, \left(x^m, y^l\right) \right\} \tag{4.14}$$

Then, let us define the error between real and model output:

$$E_i = y^i - \left(a^T x^i + b\right) \tag{4.15}$$

Since our goal is to minimize the errors over the training set $E_i, i = 1, \ldots, l$, a classic approach for calculating model parameters is considering the square loss function and then minimize the MSE:

$$\min_{a,b} \sum_{i=1}^{l} \left(y^i - a^T x^i - b\right)^2 \tag{4.16}$$

The training of the model usually relies on the minimization of the MSE, but also the minimization of another loss is allowed [48]. Indeed we can rewrite the loss function by relying on the absolute value formulation as:

$$\min_{a,b} \sum_{i=1}^{l} \left|y^i - a^T x^i - b\right| \tag{4.17}$$

and then, by using the transformation shown in the section 4.2, we obtain:

$$\min_{a,b,z} \sum_{i=1}^{l} z_i \quad \text{s.t. } \left|y^i - a^T x^i - b\right| \leq z_i \quad \forall i = 1, \ldots, l \tag{4.18}$$

that is:

$$\min_{a,b,z} \sum_{i=1}^{l} z_i \quad \text{s.t.} \quad -z_i \leq y^i - a^T x^i - b \leq z_i \quad \forall i = 1, \ldots, l \tag{4.19}$$

This is the well-known Least Absolute Deviation (LAD) model (also known as least absolute residual, least absolute error or least absolute value model. ℓ_1-norm is a good choice for two reasons: first, the model we get is very easy to solve (since it is equivalent to a linear programming problem); second, ℓ_1-norm is less sensitive to outliers. As a cons there is the fact that the model obtained after the minimization is not always unique, but can differ in a range depending on the distribution of data.

4.3 Frank-Wolfe method

The Frank-Wolfe method also called conditional gradient method or reduced gradient method is an optimization algorithm originally proposed by Frank and Wolfe in 1956 to solve quadratic programming problems with linear constraints. In this section we describe the classical method and its properties. This theoretical part will find application in section 4.4.4 where an alignment method proposed in [39] will be discussed.

4.3.1 Description of the algorithm

Let us consider a problem of the form

$$\min f(x) \quad \text{s.t.} \quad x \in C \tag{4.20}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is a continuously differentiable function and $C \subseteq \mathbb{R}^n$ is a convex compact set.

We start with a feasible solution and, at each iteration, we define a descent direction in the current iterate x_k by solving the problem:

$$\min_{x \in C} \nabla f\left(x_k\right)^T \left(x - x_k\right) \tag{4.21}$$

We notice that this is equivalent to minimize the linear approximation of f in x_k :

$$\min_{x \in C} f\left(x_k\right) + \nabla f\left(x_k\right)^T \left(x - x_k\right) \tag{4.22}$$

By the Weierstrass theorem amd compactness of C, we can deduce that there exists a solution $\hat{x}_k \in C$ for the linearized problem. Now, let us see a useful proposition:

Now, under the previous hypothesis, $C \subseteq \mathbb{R}^n$ being a convex set, and adding the condition that $f \in C^1\left(\mathbb{R}^n\right)$ is a convex function, it can be proved that $x^* \in C$ is a global solution of the following problem:

$$\min f(x) \quad \text{s.t.} \quad x \in C \tag{4.23}$$

if and only if

$$\nabla f\left(x^*\right)^T \left(x - x^*\right) \geq 0 \qquad \forall x \in C \tag{4.24}$$

Therefore in Eq. (4.22), we have two cases. If $\nabla f\left(x_k\right)^T \left(\hat{x}_k - x_k\right) = 0$ then we have

$$0 = \nabla f\left(x_k\right)^T \left(\hat{x}_k - x_k\right) \leq \nabla f\left(x_k\right)^T \left(x - x_k\right) \quad \forall x \in C$$

and x_k satisfies first order optimality conditions.

On the contrary, if $\nabla f\left(x_k\right)^T \left(\hat{x}_k - x_k\right) < 0$ we have a new descent direction in x_k given by $d_k = \hat{x}_k - x_k$. Thus we can have a new iterate given by $x_{k+1} = x_k + \alpha_k d_k$ with $\alpha_k \in (0, 1]$ calculated by means of a line search.

Finally, we report the pseudocode:

Algorithm 4.1 General FW Algorithm

1: Choose a point $x_1 \in C$
2: **for** $k = 1, 2, \ldots$
3: Compute $\hat{x}_k = \arg\min_{x \in C} \nabla f\left(x_k\right)^T \left(x - x_k\right)$
4: **if** $\hat{x}_k$ satisfies a specific condition
5: STOP
6: **end if**
7: Choose $\alpha_k \in (0, 1]$ as a suitable step size
8: Update $x_{k+1} = x_k + \alpha_k \left(\hat{x}_k - x_k\right)$
9: **end for**

4.4 Contributions

Let us recap the notation adopted. It is important to precise that the present work is based on the comparison of two different monolingual word embeddings in the same language. We will refer to them as e_1 and e_2 :

$$e_1 : \mathcal{D}_1 \to \mathcal{V}_2 \quad e_2 : \mathcal{D}_2 \to \mathcal{V}_2 \tag{4.25}$$

where $\mathcal{D}_1$ and $\mathcal{D}_2$ are the vocabularies and $\mathcal{V}_1$ and $\mathcal{V}_2$ are the corresponding embedding spaces of dimensions d. We consider $\mathcal{D}_1 = \mathcal{D}_2 = \mathcal{D}$ and we denote $|\mathcal{D}| = N$. All the word vectors $\overrightarrow{x_i}$ and $\overrightarrow{y_i}$ with $i = 1, \ldots, N$ are put one after the other as the columns of the matrices $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{d \times N}$ respectively.

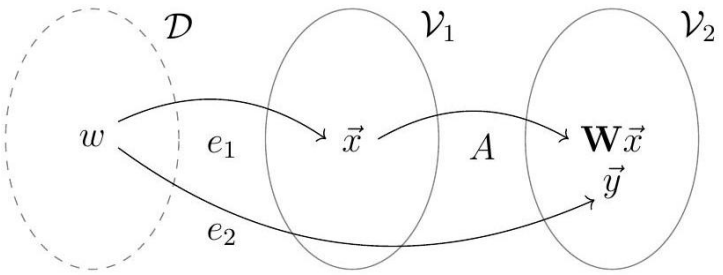


Figure 4.1: Representation of the embedding maps $e_1 : \mathcal{D} \to \mathcal{V}_1, e_2 : \mathcal{D} \to \mathcal{V}_2$ and the alignment map $A : \mathcal{V}_1 \to \mathcal{V}_2$ acting in turn on a word w of a vocabulary $\mathcal{D}$.

4.4.1 How Alignment Should Highlight Biases

The underlying concept of this study posits that the utilization of an alignment algorithm, conventionally employed in the domain of machine language translation, may serve as a valuable tool when applied to two distinct embedding spaces housing text characterized by inherent bias. The central premise of this research is that the employment of such an alignment algorithm, followed by the mapping of these two embeddings onto each other, facilitates the elucidation of bias within the text. This elucidation is achieved through the examination of word proximity using techniques such as k-nearest neighbors or CSLS. As explained in chapter 3, every word embedding contains cultural peculiarities inherited by data, which can be then propagated through the alignment.

Let us suppose to have two word embeddings e_1 and e_2 trained on two different corpora of data of the same language reflecting different cultural traits. The goal of a classic alignment $A : \mathcal{V}_1 \to \mathcal{V}_2$ should be minimizing the error among all the words $w \in \mathcal{D}$, namely between the image $A(\vec{x})$ and $\vec{y}$, such that $e_1(w) = \vec{x}$ and $e_2(w) = \vec{y}$, as represented in figure 4.1.

Now, we make an hypothesis: we suppose that some words would be changed more by the alignment A due to the semantic cultural conditioning, while other more stable, less likely to change semantic meaning depending on societal biases or historic characteristics of the data, would change less. Consequently the approach proposed in [39] is to change the classic Procrustes alignment, by starting from a simple idea: instead of looking for a transformation which tries to minimize the error on all $w \in \mathcal{D}$, they want to obtain a map A and a corresponding matrix $\mathbf{W}$ which minimizes the error only on words which would not drastically change after applying A. In other words, the ideal alignment would leave the images $A(\vec{x}) = \mathbf{W}\vec{x}$ of representations of words more likely to be subject to cultural semantic conditioning farther from the corresponding $\vec{y}$ than the words more likely to be stable.

4.4.2 Embedding Alignment: A Nonlinear Programming Approach

In this section alignments A_1 and A_2 are described. These alignment methods first proposed in [39] are here described and the first one of them is also been written in python language and tested in our dataset.

For all the alignments described, the desired A_i is a map of the type:

$$A_i : \mathcal{V}_1 \underset{\vec{x} \mapsto \mathbf{W}_i \vec{y}}{\longrightarrow} \mathcal{V}_2 \tag{4.26}$$

As explained in chapter 2, in order to find the map A it is necessary to find a transformation matrix $\mathbf{W}_i \in \mathbb{R}^{d \times d}$.

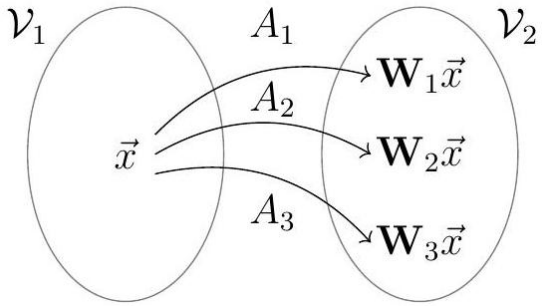


Figure 4.2: Representation of the alignment maps $A_i : \mathcal{V}_1 \to \mathcal{V}_2, i = 1, 2, 3$ acting on a word vector $\vec{x}$ of the first embedding space $\mathcal{V}_1$.

4.4.3 A_1 : An Alignment Optimized by a Linear Decomposition

In order to obtain the first alignment matrix $\mathbf{W}_1$, thanks to this new method, we consider the following optimization problem:

$$\mathbf{W}_1 = \underset{W \in \mathbb{R}^{d \times d}}{\arg\min} \|\mathbf{WX} - \mathbf{Y}\|_1 \tag{4.27}$$

which differs from the ortogonal method of section 2.1.2 and equation 2.5 only because now, instead of a Frobenius norm, a norm-1 is taken. Under this formulation, the problem coincides with:

$$\mathbf{W}_1 = \underset{W \in \mathbb{R}^{d \times d}}{\arg\min} \sum_{i=1}^{N} \|\mathbf{W}\overrightarrow{x_i} - \overrightarrow{y_i}\|_1 \tag{4.28}$$

where $\overrightarrow{x_i}$ and $\overrightarrow{y_i}$ are the i-th columns of the matrices $\mathbf{X}$ and $\mathbf{Y}$.

Let $\overrightarrow{x_i} := x_i$, $\overrightarrow{y_i} := y_i$, the j-th element of $\overrightarrow{y_i}$ be y_i^j, and the j-th column of $\mathbf{W}$ be w^j. Then the formulation above becomes:

$$\mathbf{W}_1 = \underset{W \in \mathbb{R}^{d \times d}}{\arg\min} \sum_{i=1}^{N} \sum_{j=1}^{300} \left| \left(w^j\right)^T x_i - y_i^j \right| = \underset{W \in \mathbb{R}^{d \times d}}{\arg\min} \sum_{j=1}^{300} \sum_{i=1}^{N} \left| \left(w^j\right)^T x_i - y_i^j \right| \tag{4.29}$$

Consequently, by the nature of the problem, it is possible to solve the latter formulation by combining the solutions of 300 subproblems of the type:

$$\min_{w^j \in \mathbb{R}^d} \sum_{i=1}^{N} \left| \left(w^j\right)^T x_i - y_i^j \right| \quad \forall j = 1 \dots 300 \tag{4.30}$$

Each of these non-linear problems behaves like the linear regression model explained in section 4.2, and consequently it can be replaced by the following equivalent linear problem:

$$\min_{w^j \in \mathbb{R}^d, z \in \mathbb{R}^N} \sum_{i=1}^{N} z_i \quad \text{s.t.} \quad -z_i \leq \left(w^j\right)^T x_i - y_i^j \leq z_i \quad \forall j = 1 \dots 300 \tag{4.31}$$

where $z \in \mathbb{R}^N$ is a new variable.

The subdivision into smaller problems is the key to the solution of the problem.

We than produced an amplpy script for linear optimization and we worked to parallelize it into multiple GPU's provided by the HPC framework of the department. However as we will see in the last chapeter we did not achieve this result.

4.4.4 A_2 : IMPROVING THE ALIGNMENT THROUGH FRANK-WOLFE METHOD

For the computation of the second alignment matrix $\mathbf{W}_2$ using the FW method, we start from the following formulation, based on the o-norm [39], which, by the way, is not a proper norm.

Taking into account $\mathbf{W} \in \mathbb{R}^{d \times d}$, $\mathbf{A} \in \mathbb{R}^{d \times N}$, $z \in \mathbb{R}^N$ we can write:

$$\mathbf{W}_2 = \arg\min_{\mathbf{W}, \mathbf{A}, z} \|z\|_0 \quad \text{s.t.} \quad -a_i \leq \mathbf{W} x_i - y_i \leq a_i, \quad e^T a_i = z_i \tag{4.32}$$

where a_i is the i-th column of $\mathbf{A}$ as usual, and e is a vector of 1 s. In our case the 0-norm should lead to a better solution for our purposes, since it minimize the error function z by distributing less its components between 0 and the maximal value.

It is possible then to rely on the following approximation of the objective function:

$$\|z\|_0 \approx \sum_{i=1}^{N} \left(1 - e^{-\alpha z_i}\right) = f(z) \tag{4.33}$$

where α is a fixed positive integer. Therefore, we obtain a concave function, given by a finite sum of concave functions, depending only on the variable z.

$$\nabla f\left(x_k\right) \cdot x = \nabla f\left(z_k\right) \cdot z \tag{4.34}$$

since $\frac{\partial f}{\partial w_j^l} = \frac{\partial f}{\partial a_i^j} = 0 \quad \forall \quad j = 1, .., d, l = 1, \ldots, d, i = 1, \ldots, N$. As a result, the minimization problem on which the computation of $\mathbf{W}_2$ relies becomes:

$$\min_{\mathbf{A},\mathbf{W},z} \nabla f\left(z_k\right) \cdot z \quad \text{s.t.} \quad -a_i \leq \mathbf{W}x_i - y_i \leq a_i, \quad e^T a_i = z_i \tag{4.35}$$

Let us call Ω the feasible set containing all the possible $\mathbf{W} \in \mathbb{R}^{d \times d}$, $\mathbf{A} \in \mathbb{R}^{d \times N}, z \in \mathbb{R}^N$ satisfying the constraints above. Now, let us see the pseudocode of the version of the FW algorithm we use. We recall that $z_k \in \mathbb{R}^N$, $g_k := \nabla f\left(z_k\right) \in \mathbb{R}^N$ and it are parameters which change value throughout the iterations of the algorithm, while ϵ, used for the cut-off condition, is fixed. At the beginning of the algorithm, $\left(g_k\right)_i$ is initialized as $\left(g_0\right)_i$

Algorithm 4.2 Norm-0 FW Algorithm

1: Initialize iteration counter it to 0.
2: Initialize vector $\left(g_k\right)_i$ for each $i = 1$ to N using $\left(g_0\right) \cdot i$.
3: Perform optimization to find $\mathbf{A}$, $\mathbf{W}$, and z that minimize $\nabla f(z_k) \cdot z$ subject to $Z \in \Omega$.
4: Increment iteration counter it to 1.
5: **repeat**
6: Copy values of vector z into $\left(z_k\right)_i$ for each $i = 1$ to N.
7: Calculate $\left(g_k\right)_i$ for each $i = 1$ to N using $\alpha e^{-\alpha(z_k)_i}$.
8: Perform optimization to find $\mathbf{A}$, $\mathbf{W}$, and z that minimize $\nabla f(z_k) \cdot z$ subject to $z \in \Omega$.
9: Increment iteration counter it by 1.
10: **until** $\sum_{i=1}^{N} (g_k)_i (z_i - (z_k)_i) \geq -\epsilon$

However, the processing of such an algorithm would again require the resolution of a problem containing almost 5 millions of constraints at each iteration (in line 3 and then in line 8). It is possible to realize a subdivision of the problem into smaller problems in order to make the computation lighter, by the following substitutions:

$$e^T a_i = \sum_{j=1}^{300} a_i^j = z_i \Rightarrow$$

$$\Rightarrow \nabla f(z_k) \cdot z = \sum_{i=1}^{N} (g_k)_i z_i = \sum_{i=1}^{N} (g_k)_i \sum_{j=1}^{300} a_i^j = \sum_{i=1}^{N} \sum_{j=1}^{300} (g_k)_i a_i^j = \sum_{j=1}^{300} \sum_{i=1}^{N} (g_k)_i a_i^j \tag{4.36}$$

Consequently, the optimization problem becomes:

$$\min_{\mathbf{A},\mathbf{W}} \sum_{j=1}^{300} \sum_{i=1}^{N} (g_k)_i a_i^j \quad \text{s.t.} \quad -a_i \leq \mathbf{W}x_i - y_i \leq a_i \tag{4.37}$$

Since $f(z)$ is a concave function, $(g_k)_i > 0 \quad \forall i = 1, \ldots, N$, this means that the previous problem can be solved by combining the solutions of 300 subproblems of the type:

$$\min_{a^j, w^j} \sum_{i=1}^{N} (g_k)_i a_i^j \quad \text{s.t.} \quad -a_i^j \leq \left(w^j\right)^T x_i - y_i \leq a_i^j \quad \forall j = 1, \ldots, 300 \tag{4.38}$$

where $w^j \in \mathbb{R}^d$ and $a^j \in \mathbb{R}^N$ are the j-th rows of $\mathbf{W}$ and $\mathbf{A}$. Similarly as before, let us call Ω_j the feasible set containing all the possible $w^j \in \mathbb{R}^d$ and $a^j \in \mathbb{R}^N$ respecting the constraints above, for each j. In this case, the pseudocode becomes:

Algorithm 4.3 FW Optimization Algorithm Rewritten

1: Initialize iteration counter it to 0.
2: Initialize vector $(g_k)_i$ for each $i = 1$ to N using $(g_0)_i$.
3: Find a^j and w^j that minimize $\sum_{i=1}^{N} (g_k)_i a_i^j$ subject to $a^j, w^j \in \Omega_j$ for $j = 1$ to 300.
4: Increment iteration counter it to 1.
5: **repeat**
6: Compute $(z_k)_i$ as $\sum_{j=1}^{300} a_i^j$ for each $i = 1$ to N.
7: Update $(g_k)_i$ as $\alpha e^{-\alpha(z_k)_i}$ for each $i = 1$ to N.
8: Find a^j and w^j that minimize $\sum_{i=1}^{N} (g_k)_i a_i^j$ subject to $a^j, w^j \in \Omega_j$ for $j = 1$ to 300.
9: Increment iteration counter it by 1.
10: **until** $\sum_{i=1}^{N} (g_k)_i \left(\sum_{j=1}^{300} a_i^j - (z_k)_i \right) \geq -\epsilon$

We are certain that the algorithm converges in a finite number of steps because, it can be

proved, that the problem:

$$\min f(x) \quad \text{s.t.} \quad x \in C \tag{4.39}$$

with $f \in C^1(\mathbb{R}^n)$ concave function lower bounded on C, and $C \subseteq \mathbb{R}^n$ polyhedron, can be optimized via a Frank-Wolfe algorithm with unit stepsize (as in our case). The algortihm, indeed, converges to a stationary point in a finite number of steps.

In our case, the function f to minimize is concave, and the feasible sets Ω_j are polyhedrons, since they are intersections of linear constraints, thus we can apply this technique. Moreover, it is possible to obtain a further reduction of the dimension of the problem by applying the idea proposed by Rinaldi et al. $((2008))$, which is based on the following consideration:

$$\begin{aligned} \text{if } (z_k)_i = 0 \Rightarrow \sum_{i=1}^{300} a_i^j = 0 \Rightarrow a_i^j = 0 \quad \forall j = 1, \ldots, 300 \Rightarrow \\ \Rightarrow 0 \leq \left(w^j\right)^T x_i - y_i \leq 0 \Rightarrow \left(w^j\right)^T x_i - y_i = 0 \end{aligned}$$

Consequently, let us define the set of indices $I_0 = \{i = 1, \ldots, N \text{ s.t. } (z_k)_i \neq 0\}$, and the j-th minimization subproblem becomes:

$$\begin{array}{ll} \min_{a^j, w^j} & \sum_{i \in I_0} (g_k)_i \, a_i^j \\ \text{s.t.} & \left(w^j\right)^T x_i - y_i = 0 \quad \forall i \notin I_0 \\ & -a_i^j \leq \left(w^j\right)^T x_i - y_i \leq a_i^j \quad \forall i \in I_0 \end{array}$$

Let us call $\bar{\Omega}_j$ the space containing all the possible $w^j \in \mathbb{R}^d$ and $a^j \in \mathbb{R}^N$ respecting the new constraints above, for each j. Finally, the pseudocode can be modified by replacing the lines 3 and 8 with:

$$a^j, w^j \leftarrow \arg\min_{a^j, w^j} \sum_{i \in I_0} (g_k)_i \, a_i^j \text{ such that } a^j, w^j \in \bar{\Omega}_j \forall j = 1 \ldots 300$$

Again, through the modeling language AMPL and CPLEX algorithms it is possible to find the optimal solution for $\mathbf{W}_3{}^{25}$

4.4.5 Notions of 'Distance' in Embedding Spaces Preserving Words' Similarity

Cosine similarity

One of the fundamental measures for our experiments is the cosine similarity which is often used in NLP as we showed in section 1.3.1. In particular we define it as:

$$s_i(w) := \cos\left(\mathbf{W}_i\vec{x}, \vec{y}\right) \tag{4.40}$$

This formula is used in our experiments as a measure of distance both for our nearest neighbours computation between the target space and the the mapped space, but also to find the nearest neighbours of the target space itself.

where $\vec{x} \in \mathcal{V}_1$ and $\vec{y} \in \mathcal{V}_2$ are the representations of w in the two embedding spaces.

CSLS (Cross-Domain Similarity Local Scaling)

The CSLS (Cross-Domain Similarity Local Scaling) is another way to measure distances among words in embedding spaces and it has been proved to work better on machine translation tasks [1]. This score is not necessarily a number between 0 and 1 if the vectros are not normalized.

The CSLS score is calculated using the formula proposed in [30]:

$$\mathrm{CSLS}(x, y) = -2\cos(x, y) + \frac{1}{k}\sum_{y' \in N_Y(x)} \cos\left(x, y'\right) + \frac{1}{k}\sum_{x' \in N_X(y)} \cos\left(x', y\right) \tag{4.41}$$

where $N_Y(x)$ is the set of k nearest neighbors of x in the target space. The authors built this retrieval criterion into their margin model by using unpaired words (those with no explicit translation in the training set) as negative samples when computing nearest neighbors

The CSLS score depends on the difference between the cosine similarity and the norms of the target and source words. This means that CSLS scores can be positive, negative, or close to zero, depending on the specific vectors and their relationships.

Typically, when using CSLS for nearest neighbor search or other tasks words of the target space with an high CSLS score are considered as more similar to the source word. However, there is no specific range restriction for CSLS scores, and their interpretation should depend on the specific context and application.

5 Experiments

5.1 Our Newspapers Dataset

The data we are provided with are from the JRC (Joint Research Center Data Catalog) under the EMM (European Media Monitor). The Europe Media Monitor (EMM)* family of applications has been analysing up to 220.000 news reports per day since 2004. EMM recognizes names mentioned in the news in more than twenty languages and decides automatically for each newly found name whether it belongs to a new entity or whether it is a spelling variant of a previously known entity. This resource allows EMM users to display news about people or organisations even if their names are spelt differently or if the news articles are written in different languages and scripts.

We requested two datasets of all their Italian news articles concerning politics from December 2017 to April 2018 and from June 2022 to October 2022. We received 331k articles for the period 2017 – 2018 and 591k for 2022. As it can be seen in Figure 5.3 the sources are heterogeneous. For the first period we counted 510 sources of newspapers articles. In the second period this number doubled, rising to 1018 sources. For this reason, we restricted ourselves to the journals active in both periods. From this intersection, we obtained a list of 431 newspapers that we classified in left and right to test the methodology and search for social bias.

This choice was made because of the Italian elections, which have always been a period of

*For more informations see http://emm.newsbrief.eu/overview.html

great polarization of opinions in newspapers among the different parties. In particular, for what concern the first period, the elections took place Sunday 4 March 2018, the results saw the centre-right establish itself as the most voted coalition, with around 37% of the votes, while the single most voted list, the Five Star Movement (M5S), garnered over 32% of the votes [49]. It is important to remember that the Five Star Movement and the League are considered anti-establishment parties thus their view or belief is one which stands in opposition to the conventional social, political, and economic principles of a society. For what concern the second period data, in the elections of Sunday 25 September 2022 the results saw the center-right led by Giorgia Meloni with the Fratelli d'Italia party establish itself as the coalition with the most votes, with around 44% of the preferences, winning an absolute majority in both chambers. These data represent a good testing ground for our alignment algorithm since we expect a great deal of polarization and political bias due to the political situation.

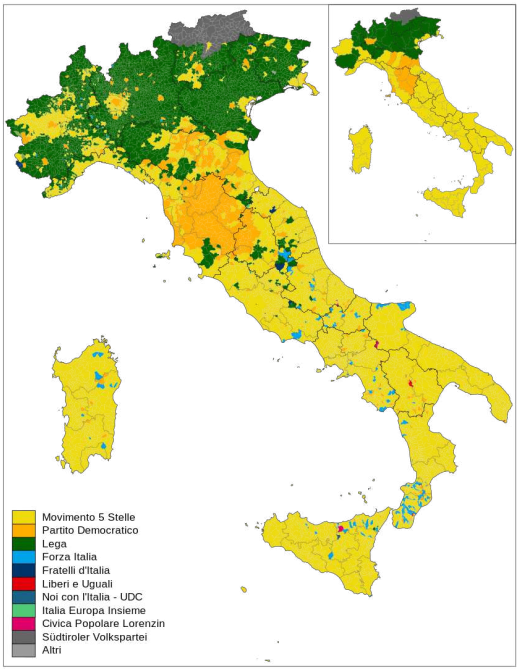


Figure 5.1: Distribution of votes in the Chamber of Deputies by party list for 2018 elections, source Wikipedia

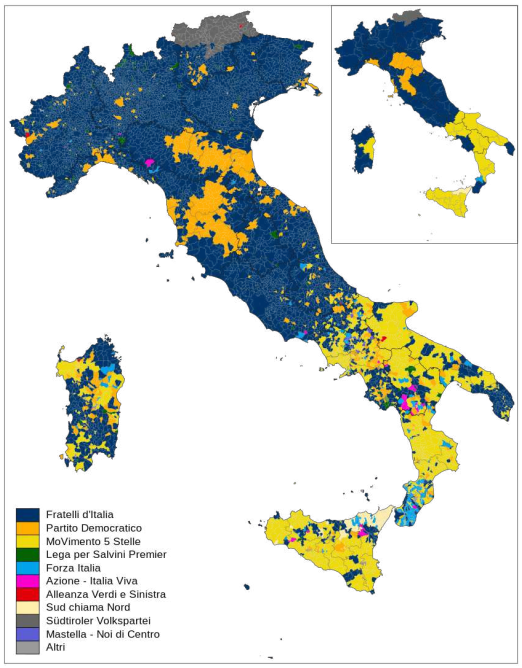


Figure 5.2: Distribution of votes in the Chamber of Deputies by party list for 2022 elections, source Wikipedia

The classification of these 431 newspapers as left and right, in order to create the two embedding spaces, was done referring to [50]. The categorical classification is:

- Right party: 2
- Center party: 1
- Left party: 0
- Unclear: un

For the sake of space, we do not report the full list with all the journals common to the two periods, but just the names of the sources classified as right or left party.

LEFT PARTY	RIGHT PARTY
avanti, liberta, mattinopadova, ilmattino repubblica, corriere, la7, ilrestodelcarlino, ilmanifesto lastampa, ilmessaggero, redattoresociale, espressorepubblica nextquotidiano, flcgil, radiopopolare	247libero, ilsussidiario, libero-news LaNazione, ilmiogiornale, ilgiornale iltempo, panorama, avvenire secoloditalia, loccidentale

Table 5.1: Set of classified newspapers in left and right party

As demonstrated in Figure 5.3, there is a disproportionate number of articles published by 247libero, which is classified as right wing, compared to left wing newspapers.

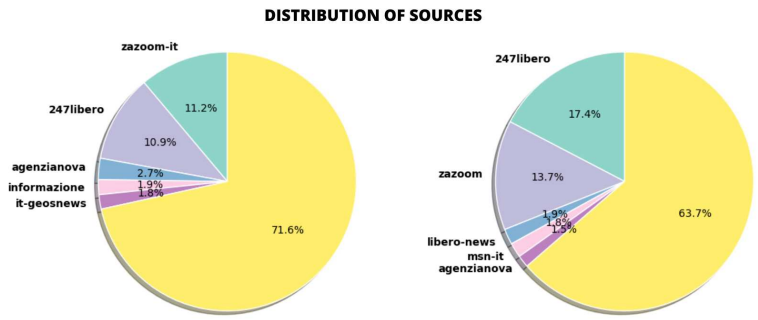


Figure 5.3: Distribution of Italian newspapers sources composing the dataset. Percentage number of articles for every source.

As we can see there are many newpapers inside our list. Classifying newspapers based on their political affiliation can be a challenging task, and it often involves analyzing their content, editorial stance, and historical context. Since we have to classify each one of them as left or right party the classification could be personal and dictated to our current knowledge and could lead to unexpected results during the experiments.

Italian elections and the Chameleon-Like Characteristics of the Italian Political Scene

As we mentioned in the previous chapter we are searching for social bias in Italian newspapers published after the elections. To better understand the context we provide a brief description of the Italian electoral setting in order to understand how subtle and complex is the whole topic.

National-level elections in Italy are periodically called to form a parliament consisting of two houses: the Chamber of Deputies (Camera dei Deputati) with 400 members; and the Senate of the Republic (Senato della Repubblica) with 200 elected members, plus a few appointed senators for life. Italy is a parliamentary republic, and the President of the Republic is elected for a seven-year term. Italy's electoral system has been constantly reformed since the 1980s in an attempt to adapt to the country's parliamentary system, but not without complexity[†]. It is currently governed by the Rosatellum bis law of 2017, which enshrines direct universal suffrage through a so-called mixed electoral system, that is, one that combines the use of majority and proportional representation. Elections for the lower house of parliament and the Senate are held in one round. While general elections are normally held every five years in Italy, early elections can be called by the president after dissolution of parliament, as happened in 2022. The Italian electoral system has an additional specificity due to its mixed voting system. It allows Italian MPs and senators to have multiple candidacies. Each candidate can be the leading candidate in a constituency (called a single-member constituency) and, at the same time, be present on up to five other regional lists. In addition to this complexity, it should be noted that government policies are often not well-defined when it comes to major issues such as combating tax evasion, managing immigration flows, and handling welfare. This is due to the so-called 'Chameleon-Like' or 'Transformism' effect adopted by political parties to secure as many votes as possible during elections [51].

5.2 Preprocessing

In the context of our research, we conducted a series of preprocessing steps on the textual data. For the design of the preprocessing we took inspiration from [52]. These steps were aimed at standardizing and cleaning the text, ensuring that it is ready for subsequent analysis. It is important to note that these preprocessing steps were specifically designed for Italian text data, and they may vary for other languages or specific tasks.

[†]For further informations visit https://temi.camera.it/leg18/temi/tl18_riforma_elettorale.html

1. **Exclusion of Title:** To streamline our analysis, we excluded the titles of articles from our text data. This choice is due to the fact that in the 80% of cases the title is a simple repetition of the first line of the article.

2. **Removal of Non-Alphanumeric Tokens:** All non-alphanumeric tokens, such as special characters and symbols, were removed from the text. This step helped eliminate noise and ensured that our analysis focused on meaningful content.

3. **Lowercasing:** To maintain consistency and reduce the dimensionality of the text data, all characters were converted to lowercase.

4. **Removal of Direct Quotes:** Any text enclosed within quotation marks was removed from the corpus. This step aimed to exclude quoted content that might not be relevant to our analysis.

5. **Handling Non-ASCII Characters:** Non-ASCII characters were replaced with their closest ASCII equivalents. This transformation ensured that the text was consistently encoded.

6. **Period Handling:** Periods (full stops) were removed, except in cases where they did not indicate the end of a sentence. Additionally, end-of-sentence periods were replaced with the token 'PERIOD'.

7. **Numerals Handling:** Numerals were removed from the text, except in cases where they represented ordinal numbers (e.g., 'first', 'second').

Another preprocessing step could be remove all proper names (words starting with a capital letter).

In some applications there is also a subsampling or minimum count parameter in order to mitigated the influence of rare and infrequent words, but we did not apply this technique to preprocessing. The previous preprocessing steps were crucial to prepare the text data for the subsequent analysis. By standardizing the text and removing noise, we aimed to enhance the quality and consistency of our data, allowing for more robust and meaningful results in our research. It is often considered good practice to evaluate embeddings using analogy or syntax tests. However, in our case, since we are working with Italian word embeddings rather than English ones, we encountered challenges in finding suitable tests tailored to our context. The lack of appropriate evaluation benchmarks for Italian embeddings posed a significant obstacle; thus, we do not have an evaluation of our embeddings.

As a disclaimer for the mathematical notation used refer to 2.

5.3 MUSE Framework

5.3.1 Embedding Space Creation With FastText

MUSE library provide an alignment technique based on FastText embeddings. These embeddings guarantees more flexibility because they take into account subword informations as mentioned in section 1.5.1. In this way we can test bias of words less worried about making an out of vocabulary (OOV) mistake because of plurals or word variations.

The corpus of text considered for the experiments are all 331k articles of the first period (p1) divided into left and right respectively according to the criterion of Table 5.1

The software is optimised to work using CUDA and was adapted and run on the cluster of GPUs provided by the university.

In all experiments the source space corresponds to the right-party articles while the target space corresponds to the left-party articles.

5.3.2 Supervised Results

We should remember that the MUSE method, in its supervised form, employs Procrustes alignment, as described in the orthogonal methodology of section 2.1.2. This methodology is supervised because it requires the presence of parallel data, which means having a dictionary of source words and their respective translations in the target space. In our case we used as parallel data the words itemized in 3.2. This is essential for aligning the two spaces through SVD decomposition. This alignment algorithm enables the calculation of the translation matrix for a substantial dictionary. In the case at hand, the dictionary for both spaces comprises approximately 200k words.

Since the words more likely to be affected by societal bias (Table 3.1) are too many to be plotted in the same space, we divided the analysis into substeps. In particular we further divided our list of words into sub-lists containing the gender/sexual related, political and racial connotation words.

Words With Racial Discriminative Connotation

Words with racial connotations are terms associated with specific racial or ethnic groups and can be offensive, derogatory, or perpetuate stereotypes. It is important to note that using such

words can be hurtful, disrespectful, and perpetuate discrimination and prejudice. The words we selected for our experiments correspond to the first row of Table 3.1.

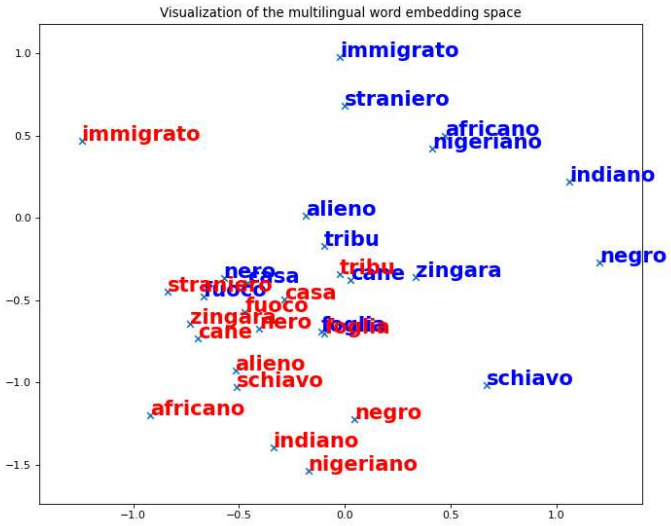


Figure 5.4: Distribution of racial related words in Italian using supervised MUSE. Source in blue (right party), target in red. Reduction to 2D space via PCA.

As we can see in Figure 5.4, where we plot the distribution of words in a two dimensional space, the words belonging to the parallel dictionary are almost perfectly superimposed and thus aligned, see words: 'foglia', 'fuoco' and 'casa'. The Variance explained by the PCA is 0.07. The other words corresponding to racial related words are however still not aligned between the two right party (blue) and left party (red) embedding spaces, showing that either stable words are not such good pinpoints or that these words are used differently by the two parties.

After this preliminary analysis, we moved on to the next step of computing the 5 nearest neighbours using as metric distance the cosine similarity as explained in section 4.4.5. The complete results can be seen in the BONs repository. Here, for the sake of space we report just the more interesting results. On the left side we have right party words and on the left side there are right party words.

Nearest neighbors of 'africano':

1. 1.0000 - africano
2. 0.8861 - nordafricano

Nearest neighbors of 'negro':

1. 1.0000 - negro
2. 0.7185 - bianco

Nearest neighbors of 'schiavo':

1. 1.0000 - schiavo
2. 0.8541 - schiavotto

Nearest neighbors of 'africano':

1. 0.3901 - ritrosiadiffidenza
2. 0.3834 - sombrero

Nearest neighbors of 'negro':

1. 0.4047 - marigold
2. 0.3920 - marlboro

Nearest neighbors of 'schiavo':

1. 0.4228 - malocchio

As expected right newspapers use highly discriminative words such as 'negro' and usually also other terms as 'africano' and 'schiavo'. On the contrary left side journals never used that words directly since we never have a cosine similarity of one. All cosine similarity in this case are very low meaning that in the left party dictionary is difficult to find words analogically related to racial biased ones.

Words With Gender Discriminative Connotation

The same analysis is applied to words with a gender discriminative connotation. An example of the most interesting results is given by the following list. On the left silde we have right party words and on the left side there are right party words.

Nearest neighbors of 'omosessuale':

1. 1.0000 - omosessuale
2. 0.9218 - omosessualita

Nearest neighbors of 'gay':

1. 1.0000 - gay
2. 0.7185 - gaypride
3. 0.6803 - lesbichegaybisessualitrans

Nearest neighbors of 'lesbica':

1. 1.0000 - lesbica
2. 0.8451 - arcilesbica

Nearest neighbors of 'omosessuale':

1. 1.0000 - omosessuale
2. 0.8654 - omosessualita

Nearest neighbors of 'gay':

1. 1.0000 - gay
2. 0.6786 - gaypride

Nearest neighbors of 'lesbica':

1. 1.0000 - lesbica
2. 0.7696 - arcilesbica

In Figure 5.5 we can see the result after applying PCA reduction.

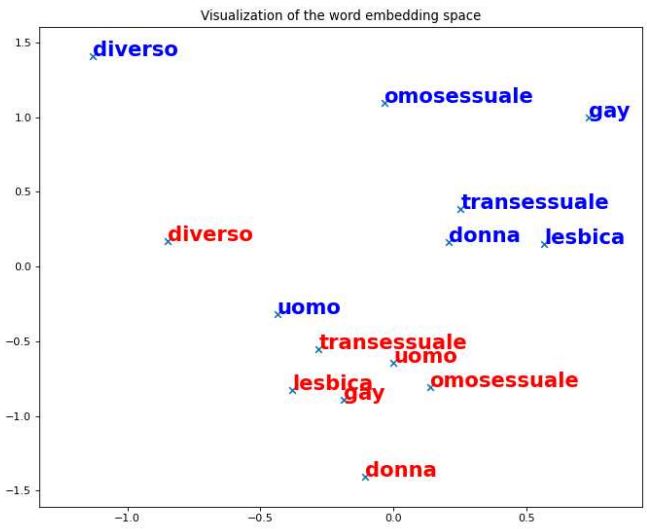


Figure 5.5: Distribution of gender related words in Italian using supervised MUSE. Source in blue (right party), target in red (left party). Reduction to 2D space via PCA.

Words Carrying Political Bias

As in the previous experiments we report a list of the most impressive results obtained applying cosine similarity distance to find the nearest neighbor of a word. On the left side we have right party words and on the left side there are right party words.

Nearest neighbors of 'padrone':

1. 1.0000 - padrone
2. 0.8331 - padrepadrone

Nearest neighbors of 'nuovo':

1. 1.0000 - nuovo
2. 0.7158 - nuovoinizio

Nearest neighbors of 'operaio':

1. 1.0000 - operaio
2. 0.8482 - preteoperaio

Nearest neighbors of 'comunista':

1. 1.0000 - comunista
2. 0.9330 - excomunista

Nearest neighbors of 'fascista':

1. 1.0000 - fascista
2. 0.9246 - afascista
3. 0.8922 - parafascista

Nearest neighbors of 'padrone':

1. 1.0000 - padrone
2. 0.8465 - canepadrone

Nearest neighbors of 'nuovo':

1. 1.0000 - nuovo
2. 0.6551 - nuova

Nearest neighbors of 'operaio':

1. 1.0000 - operaio
2. 0.8979 - operaiodi

Nearest neighbors of 'comunista':

1. 1.0000 - comunista
2. 0.9633 - excomunista
3. 0.9394 - neocomunista

Nearest neighbors of 'fascista':

1. 1.0000 - fascista
2. 0.9399 - sfascista
3. 0.9375 - postfascista

As we can see in Figure 5.6 we have that the two sub spaces of words of the right and left side appear divided into two sub spaces when represented via PCA reduction. This implies that the

words we choose are used differently by the two parties. We can notice that the supervised alignment is done respecting the parallel data we have given since the word 'foglia' is superimposed at itself.

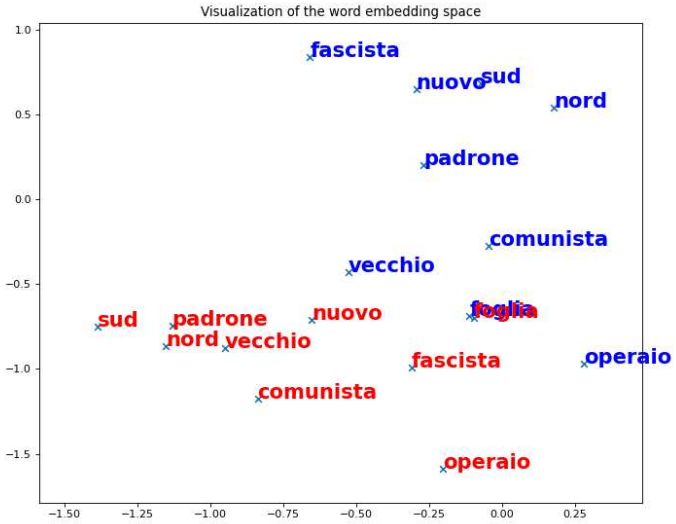


Figure 5.6: Distribution of political related words in Italian using supervised MUSE. Source in blue (right party), target in red. Reduction to 2D space via PCA.

The first consideration to do is that the two parties used the word 'padrone' in two different ways. For the left wing the second nearest neighbor was 'canepadrone' which is an Italian offense. In Italian, indeed, the word 'cane' means dog and it is used as an offense in some cases such this one. On the contrary the right party second nearest neighbor is 'padrepadrone' which is used when one imposes its dominion over others.

5.3.3 Unsupervised Results

In this section, we apply the same methodology to the case where the alignment is carried out using unsupervised methods and adversarial training, as proposed in [1]. In this scenario, the model is not provided with any parallel data dictionary that defines fixed points around which to perform the alignment.

Words With Racial Discriminative Connotation

In the following list we can compare words of right party on the left side of the list and on the left party on the other side. As usual cosine similarity is computed to find the nearest words in both the mapped space WX and compared with the ones of the target space Y. We focus on words with racial discriminative connotation.

Nearest neighbors of 'africano':

1. 1.0000 - africano
2. 0.8861 - nordafricano

Nearest neighbors of 'africano':

1. 0.6686 - africano
2. 0.6666 - nordafricano

Nearest neighbors of 'indiano':

1. 1.0000 - indiano
2. 0.7681 - francoindiano

Nearest neighbors of 'indiano':

1. 0.5835 - indiavolato
2. 0.5460 - kawasaki

Nearest neighbors of 'schiavo':

1. 1.0000 - schiavo
2. 0.8541 - schiavotto

Nearest neighbors of 'schiavo':

1. 0.5957 - schiavo
2. 0.5861 - schiavoni

In Figure 5.7 we can see a plot of the result after applying PCA and superimposing mapped source and target space. Variance explained by PCA in this case is 0.09.

Surprisingly the results are similar to the ones obtained with the supervised method.

Words With Gender Discriminative Connotation

In this case the results obtained on the words that should be subject to social bias are similar to the results obtained with the unsupervised method. For this reason we just report the PCA reduction plot in Figure 5.8.

Words Carrying Political Bias

Also in this case the results of the method are similar and in some cases equal to the results obtained using the supervised approach. For this reason er report only the PCA plot of the alignment in Figure 5.9. It is important to notice that, since the method is unsupervised it does

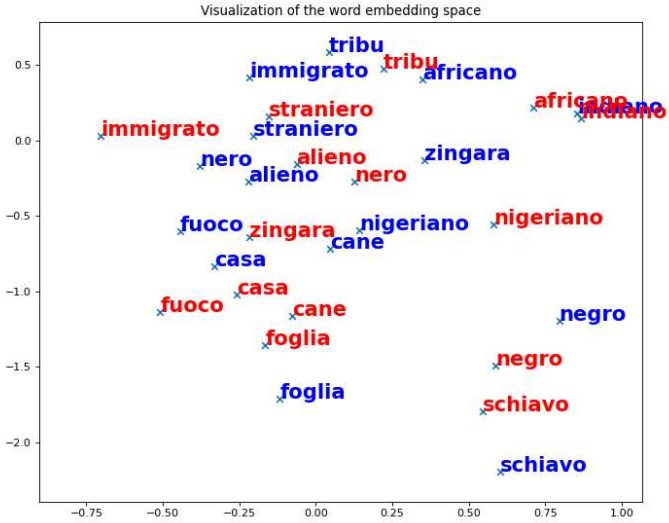


Figure 5.7: Distribution of racial related words in Italian using unsupervised MUSE. Source in blue (right party), target in red. Reduction to 2D space via PCA.

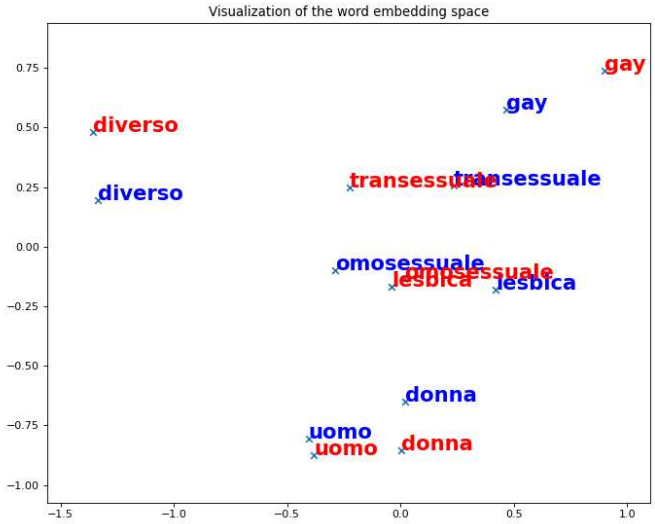


Figure 5.8: Distribution of gender related words in Italian using unsupervised MUSE. Source in blue (right party), target in red. Reduction to 2D space via PCA.

not has in input parallel data. This fact can be noticed looking, for example, at the word 'foglia' which in this case is not superimposed in itself.

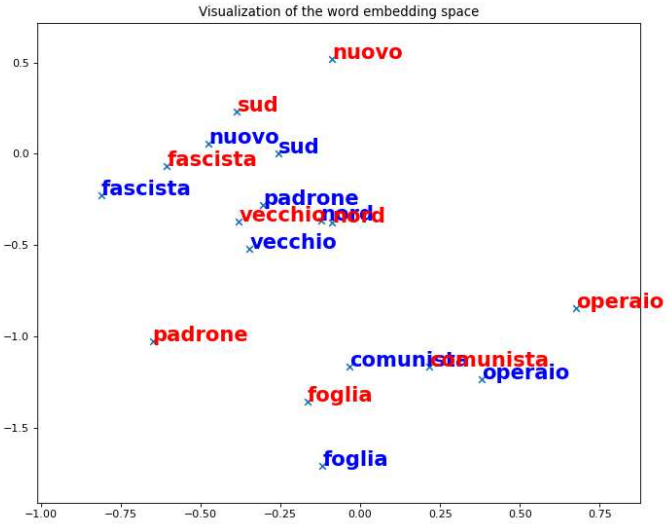


Figure 5.9: Distribution of political related words in Italian using unsupervised MUSE. Source in blue (right party), target in red. Reduction to 2D space via PCA.

From the previous analysis we can highlight the fact that MUSE unsupervised and supervised method produce the comparable results in terms of bias highlighting via nearest neighbours. The same result in the framework of machine translation is also mentioned in [1].

5.4 Norm One approach

Here we present the results of the application of the norm-1 algorithm to our dataset. This approach is different form the once adopted in MUSE since, all embeddings used were trained using the Gensim library. The training model used is the Skip-Gram, the words' window considered is 5 words long and the vector size is $d = 300$ and trained for five epochs. The algorithm is tested on the dataset using the amplpy module and the linear optimization program CPLEX, for more information on the scripts used please refer to the BONs repository[‡]. The algorithm was tested on a dataset of the first period (p1). In particular, we just analysed approximately the first 10k articles belonging to the left and right newspapers respectively. This choice was

[‡]For additional informations see: https://github.com/Pietrosanguin/BONs

made because of the compuational bottleneck imposed by the amplpy algorithm implementation. After applying preprocessing (refer to 5.2), the words common to both text corpora were taken. Clearly, the constraint of having to take the intersection of the two vocabularies presents a not insignificant restriction in itself. This intersection resulted in 4367 words common to both corpora, i.e. common to the left and right articles. The present algorithm was thus launched to optimise, by means of linear programming, a relatively small problem. We are in fact talking about two source and target embedding matrices $X, Y \in \mathbb{R}^{4367\times 300}$, thus relatively small when compared to the matrices optimised by MUSE which can have as many as 200k rows.

However, the use of amplpy and CMPLEX takes quite a long time to solve this kind of problem. We are talking about 150 min. However there is a sequence of mini-goals that is to nearly every programming problem: make it run, make it right and make it fast. Note that this list does not start with make it fast. For the moment, we have therefore limited ourselves to make it run and make it right. It is important to remember that this type of algorithm does not require parallel data and should be able to reveal any bias within the text. The algorithm implemented is still not parallelized to work on multiple GPUs since it makes use of the library pulse which is not still optimized to support CUDA. Furthermore the algorithm requires the two dictionaries to have the same number of words $\mathcal{D}_1 = \mathcal{D}_2$ since it aims to extract the words which change more. For these reasons we took the intersection of the dictionaries. The map A and the corresponding translation matrix W minimizes the error only on words which would not drastically change after applying A. Because of this limitations the vocabulary considered in the experiments is given by the words that belong to both the source and the target space $\mathcal{D}_1 \cap \mathcal{D}_2$. This operation is a further reduction in vocabulary dimension. In the experiment source is in blue, which in this case represents left wing words and target is in red representing right party words, the result is shown in Figure 5.10.

Because of the limitation on the number of words imposed by the algorithm (we are considering just 4367 words some of the biased words reported in Table 3.1 are OOV. For this reason we tested just the following words that were part of the vocabulary and subject to social bias. The words are: 'vecchio', 'ragazza', 'ragazzo', 'uomo', 'donna', 'comunista', 'fascista', 'fuoco'.

From Figure 5.10 we can see that the result is not properly what expected. The aligned words are indeed the once that we expected to be more far when represented in this joint space. Indeed words like 'comunista' and 'fascista' are strongly politically biased words. What results from this analysis is that words 'vecchio', 'ragazzo' and 'uomo' are strongly biased in our text corpora. The tests done with the FW norm-0 methodology gave really similar results, thus we decided

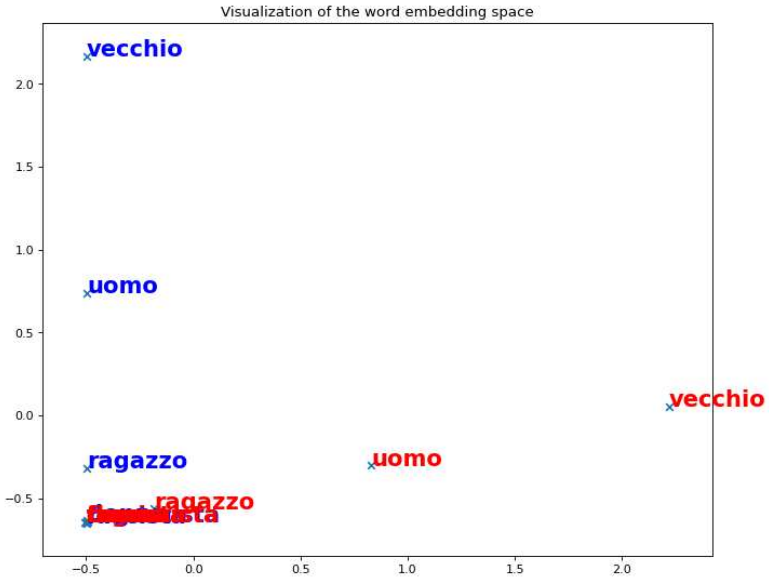


Figure 5.10: Distribution of political related words in Italian. Source in blue (left party), target in red (right party). Reduction to 2D space via PCA.

to not report them.

6

Conclusion and Future Works

It is clear from the research conducted in this thesis that word embeddings, while powerful tools for natural language processing, are not immune to the biases present in the corpora on which they are trained. The main objective of this project was to examine how embedding spaces change following an alignment. Specifically, we aimed to determine whether the same identical words in the same language are mapped to the same vector after an alignment. The outcomes obtained when applying this method to real-world data, such as newspaper articles, do not necessarily validate the effectiveness of this technique as they may not align with our initial expectations as experimenters and this result can be caused by many different causes. The results obtained in this study demonstrate that certain words in the Italian language are more prone to bias than others, as evidenced by the differences in their embeddings across different corpora. Notably, words like 'negro,' 'africano,' and 'schiavo' were found to be highly discriminative in right-leaning newspapers, while left-leaning newspapers showed significantly lower cosine similarities for these words, suggesting a conscious avoidance of such biased terminology. To facilitate further research and the replication of our experiments, we have created a shared and open-source GitHub repository named BONs. This repository contains the necessary resources for analyzing bias in monolingual texts, aligning embeddings using the MUSE library, and conducting similar experiments in other linguistic contexts. However, it is important to acknowledge that our proposed alignment algorithm, adapted from unsupervised methods and linear programming, did not yield the expected alignment results for words that are commonly associated with bias. This discrepancy may be attributed to factors such as potential misclassi-

fication of texts as left or right-leaning, limited text data used (approximately 4300 words), and other underlying complexities in bias detection.

In terms of future work, there are several avenues to explore. One potential direction is to conduct additional tests using a broader range of words to gain a more comprehensive understanding of bias in different contexts. Additionally, considering the limitations of cosine similarity, using alternative similarity measures such as CSLS (Cross-Domain Similarity Local Scaling) could provide more nuanced insights into bias detection.

Furthermore, evaluating Italian embeddings using established benchmarks and testing their performance against other linguistic tasks would be beneficial. This evaluation could help validate the effectiveness of the alignment algorithm and its applicability to different languages.

In conclusion, this thesis has shed light on the intricacies of bias within word embeddings and introduced an alignment algorithm as a tool for bias detection. While challenges remain in refining and improving the alignment process, this research aims at addressing bias in natural language processing and contributes to the ongoing efforts to create fairer and more equitable AI systems.

A Appendix

A.1 Proof of solution under orthogonality

Constraining W to be orthogonal $\left(W^T W = I\right)$, the original minimization problem can be reformulated as follows (see 2.1.4):

$$
\begin{aligned}
&\arg\min_W \sum_i \|X_{i*}W - Y_{i*}\|^2 \\
&= \arg\min_W \sum_i \left(\|X_{i*}W\|^2 + \|Y_{i*}\|^2 - 2X_{i*}WY_{i*}^T\right) \\
&= \arg\max_W \sum_i X_{i*}WY_{i*}^T \\
&= \arg\max_W \operatorname{Tr}\left(XWY^T\right) \\
&= \arg\max_W \operatorname{Tr}\left(Y^T XW\right)
\end{aligned}
$$

In the above expression, $\operatorname{Tr}(\cdot)$ denotes the trace operator (the sum of all the elements in the main diagonal), and the last equality is given by its cyclic property. At this point, we can take the SVD of $Y^T X$ as $Y^T X = U\Sigma V^T$, so $\operatorname{Tr}\left(Y^T XW\right) = \operatorname{Tr}\left(U\Sigma V^T W\right) = \operatorname{Tr}\left(\Sigma V^T WU\right)$. Since V^T, W and U are orthogonal matrices, their product $V^T WU$ will also be an orthogonal matrix. In addition to that, given that Σ is a diagonal matrix, its trace after an orthogonal transformation will be maximal when the values in its main diagonal are preserved after the

mapping, that is, when the orthogonal transformation matrix is the identity matrix. This will happen when $V^T W U = I$ in our case, so the optimal solution will be $W = VU^T$.

References

[1] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou, "Word translation without parallel data," *CoRR*, vol. abs/1710.04087, 2017. [Online]. Available: http://arxiv.org/abs/1710.04087

[2] A. Søgaard, S. Ruder, and I. Vulić, "On the limitations of unsupervised bilingual dictionary induction," 2018.

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

[4] 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.

[5] A. Kalinowski and Y. An, "A survey of embedding space alignment methods for language and knowledge graphs," 2020.

[6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[7] D. E. Rumelhart and A. A. Abrahamson, "A model for analogical reasoning." *Cognitive Psychology*, vol. 5, pp. 1–28, 1973.

[8] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," 2023.

[9] Wikipedia, "Homonym," https://en.wikipedia.org/wiki/Homonym, [Online; accessed 6 July 2023].

[10] Z. S. Harris, "Distributional structure," vol. 10, no. 2-3, pp. 146–162, 1954. [Online]. Available: https://doi.org/10.1080/00437956.1954.11659520

[11] J. Firth, "A synopsis of linguistic theory, 1930-1955," *Studies in Linguistic Analysis*, pp. 10–32, 1957. [Online]. Available: https://cir.nii.ac.jp/crid/1574231874045325568

[12] D. Jurafsky and J. H. Martin, *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, N.J.: Pearson Prentice Hall, 2009. [Online]. Available: http://www.amazon.com/Speech-Language-Processing-2nd-Edition/dp/0131873210/ref=pd_bxgy_b_img_y

[13] T. Bolukbasi, K.-W. Chang, J. Zou, V. Saligrama, and A. Kalai, "Man is to computer programmer as woman is to homemaker? debiasing word embeddings," 2016.

[14] A. Caliskan, J. J. Bryson, and A. Narayanan, "Semantics derived automatically from language corpora contain human-like biases," *Science*, vol. 356, no. 6334, pp. 183–186, apr 2017. [Online]. Available: https://doi.org/10.1126%2Fscience.aal4230

[15] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *CoRR*, vol. abs/1607.04606, 2016. [Online]. Available: http://arxiv.org/abs/1607.04606

[16] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *CoRR*, vol. abs/1508.07909, 2015. [Online]. Available: http://arxiv.org/abs/1508.07909

[17] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:160025533

[18] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting similarities among languages for machine translation," *CoRR*, vol. abs/1309.4168, 2013. [Online]. Available: http://arxiv.org/abs/1309.4168

[19] S. Ruder, I. Vulić , and A. Søgaard, "A survey of cross-lingual word embedding models," *Journal of Artificial Intelligence Research*, vol. 65, pp. 569–631, aug 2019. [Online]. Available: https://doi.org/10.1613%2Fjair.1.11640

[20] M. Artetxe, G. Labaka, and E. Agirre, "Learning bilingual word embeddings with (almost) no bilingual data," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 451–462.

[21] Y. Shigeto, I. Suzuki, K. Hara, M. Shimbo, and Y. Matsumoto, "Ridge regression, hubness, and zero-shot learning," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part I 15*. Springer, 2015, pp. 135–151.

[22] M. Artetxe, G. Labaka, and E. Agirre, "Learning principled bilingual mappings of word embeddings while preserving monolingual invariance," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2289–2294. [Online]. Available: https://aclanthology.org/D16-1250

[23] M. Faruqui and C. Dyer, "Improving vector space word representations using multilingual correlation," in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Gothenburg, Sweden: Association for Computational Linguistics, Apr. 2014, pp. 462–471. [Online]. Available: https://aclanthology.org/E14-1049

[24] A. Lazaridou, G. Dinu, and M. Baroni, "Hubness and pollution: Delving into cross-space mapping for zero-shot learning," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 270–280. [Online]. Available: https://aclanthology.org/P15-1027

[25] M. Radovanovi263;, A. Nanopoulos, and M. Ivanovi263;, "Hubs in space: Popular nearest neighbors in high-dimensional data," *Journal of Machine Learning Research*, vol. 11, no. 86, pp. 2487–2531, 2010. [Online]. Available: http://jmlr.org/papers/v11/radovanovic10a.html

[26] C. Xing, D. Wang, C. Liu, and Y. Lin, "Normalized word embedding and orthogonal transform for bilingual word translation," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, May–Jun. 2015, pp. 1006–1011. [Online]. Available: https://aclanthology.org/N15-1104

[27] P. H. Schönemann, "A generalized solution of the orthogonal procrustes problem," *Psychometrika*, vol. 31, no. 1, pp. 1–10, March 1966. [Online]. Available: https://doi.org/10.1007/BF02289451

[28] C. Gong, D. He, X. Tan, T. Qin, L. Wang, and T. Liu, "FRAGE: frequency-agnostic word representation," *CoRR*, vol. abs/1809.06858, 2018. [Online]. Available: http://arxiv.org/abs/1809.06858

[29] J. Mu, S. Bhat, and P. Viswanath, "All-but-the-top: Simple and effective postprocessing for word representations," *CoRR*, vol. abs/1702.01417, 2017. [Online]. Available: http://arxiv.org/abs/1702.01417

[30] A. Joulin, P. Bojanowski, T. Mikolov, H. Jégou, and E. Grave, "Loss in translation: Learning bilingual word mapping with a retrieval criterion," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 2979–2984. [Online]. Available: https://aclanthology.org/D18-1330

[31] D. Alvarez-Melis and T. S. Jaakkola, "Gromov-wasserstein alignment of word embedding spaces," *CoRR*, vol. abs/1809.00013, 2018. [Online]. Available: http://arxiv.org/abs/1809.00013

[32] M. Wessel, T. Horych, T. Ruas, A. Aizawa, B. Gipp, and T. Spinde, "Introducing mbib-the first media bias identification benchmark task and dataset collection," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023, pp. 2765–2774.

[33] M. Delgado-Rodriguez and J. Llorca, "Bias," *Journal of Epidemiology & Community Health*, vol. 58, no. 8, pp. 635–641, 2004.

[34] B. Friedman and H. Nissenbaum, "Bias in computer systems," *ACM Trans. Inf. Syst.*, vol. 14, pp. 330–347, 1996. [Online]. Available: https://api.semanticscholar.org/CorpusID:207195759

[35] S. L. Blodgett, S. Barocas, H. D. I. au2, and H. Wallach, "Language (technology) is power: A critical survey of "bias" in nlp," 2020.

[36] A. S. Gerber, D. Karlan, and D. Bergan, "Does the media matter? a field experiment measuring the effect of newspapers on voting behavior and political opinions," *American Economic Journal: Applied Economics*, vol. 1, no. 2, pp. 35–52, 2009. [Online]. Available: http://www.jstor.org/stable/25760159

[37] M. Bertrand and S. Mullainathan, "Are emily and greg more employable than lakisha and jamal? a field experiment on labor market discrimination," *American Economic Review*, vol. 94, no. 4, pp. 991–1013, September 2004. [Online]. Available: https://www.aeaweb.org/articles?id=10.1257/0002828042002561

[38] M.-E. Brunet, C. Alkalay-Houlihan, A. Anderson, and R. Zemel, "Understanding the origins of bias in word embeddings," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 803–811. [Online]. Available: https://proceedings.mlr.press/v97/brunet19a.html

[39] E. Della Casa, "Bias analysis in word embeddings with alignment techniques," Master's thesis, Università degli studi di Padova, 2022. [Online]. Available: https://hdl.handle.net/20.500.12608/43380

[40] A. Castelnovo, R. Crupi, G. Greco, D. Regoli, I. G. Penco, and A. C. Cosentini, "A clarification of the nuances in the fairness metrics landscape," *Scientific Reports*, vol. 12, no. 1, mar 2022. [Online]. Available: https://doi.org/10.1038%2Fs41598-022-07939-1

[41] J. E. Fair, "War, famine, and poverty: Race in the construction of africa's media image," *Journal of Communication Inquiry*, vol. 17, no. 2, pp. 5–22, 1993. [Online]. Available: https://doi.org/10.1177/019685999301700202

[42] S. Feldman, "Political ideology," in *The Oxford Handbook of Political Psychology, 2nd ed.* New York, NY, US: Oxford University Press, 2013, pp. 591–626.

[43] M. Artetxe, G. Labaka, and E. Agirre, "Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 5012–5019.

[44] N. Garg, L. Schiebinger, D. Jurafsky, and J. Zou, "Word embeddings quantify 100 years of gender and ethnic stereotypes," *CoRR*, vol. abs/1711.08412, 2017. [Online]. Available: http://arxiv.org/abs/1711.08412

[45] J. Gordon, M. Babaeianjelodar, and J. N. Matthews, "Studying political bias via word embeddings," *Companion Proceedings of the Web Conference 2020*, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:212410179

[46] A. E. Wilson, V. A. Parker, and M. Feinberg, "Polarization in the contemporary political and media landscape," *Current Opinion in Behavioral Sciences*, vol. 34, pp. 223–228, 2020, political Ideologies. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352154620301078

[47] Ongig. (2020) 25+ examples of biased language. [Online]. Available: https://blog.ongig.com/diversity-and-inclusion/biased-language-examples/

[48] D. Birkes and Y. Dodge, *Alternative methods of regression*. John Wiley & Sons, 2011.

[49] "Elezioni 2018, il calcolo dei seggi nel proporzionale: per il m5s altri 133 deputati," *La Repubblica*, 2018. [Online]. Available: https://www.repubblica.it/speciali/politica/elezioni2018/2018/03/06/news/elezioni_2018_il_calcolo_dei_seggi_nel_proporzionale_per_il_m5s_altri_133_deputati-190563291/

[50] P. R. Center. (2023) News media and political attitudes in italy. [Online]. Available: https://www.pewresearch.org/global/fact-sheet/news-media-and-political-attitudes-in-italy/

[51] M. Valbruzzi, "Is trasformismo a useful category for analysing modern italian politics?" *Journal of Modern Italian Studies*, vol. 19, no. 2, pp. 169–185, 2014.

[52] S. D'Alonzo and M. Tegmark, "Machine-learning media bias," *CoRR*, vol. abs/2109.00024, 2021. [Online]. Available: https://arxiv.org/abs/2109.00024