quote

Building a global dictionary for semantic technologies

Eszter Iklódi

Universitat Politècnica de València
Departamento de Sistemas Informáticos y Computación

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

IARFID

Valencia 2018

Dirigido por:

María José Castro Bleda

Acknowledgement

First of all, I would like to express my special gratitude to my supervisor, Gábor Recski, who, together with András Kornai, had the initial idea of this project. Besides suggesting an interesting and prosperous research topic, he supervised my work with the most care and devotion, not just over the thesis semester itself, but over the course of my whole master studies as well. Although I spent the actual thesis year abroad, he was always flexible and cooperative with my rather Spanish working habits.

Second of all, I am very grateful to Gábor Borbély for all the personal and online sessions we had. He helped a lot understand the underlying maths of the model, and he always suggested great ideas fur further improvements.

I would also like to express my gratitude to Maria Jose Castro-Bleda who was my supervisor at the Universitat Politècnica de València (UPV) during the second semester of my Erasmus stay, and who gave me access to a laboratory at the university to facilitate my work.

In addition to my academic help, I would also like to express my deepest gratitude to Zsolt Iklódi, Dávid Kelemen, and Stephan Laschet who troubled to read my thesis work, giving feedback, indicating typos, and suggesting so many great ideas that made my text more understandable even for non-expert readers.

A special thanks to all the friends I made here in Valencia, and to all of those who visited me all over this year, with whom I could always get away from work and clear my mind, so that later I could return to work with more commitment and pleasure.

Finally, I would like to thank my parents for all the support they gave me all over my life, for helping me if I needed help, but letting me do it my own way if I wanted so.

Resumen

El procesamiento del lenguaje natural juega un papel cada vez más importante en nuestra vida cotidiana. En nuestro mundo digital, el uso del lenguaje natural para la comunicación entre máquinas y humanos se ha convertido en un requisito básico. Para cumplir con este requisito, es inevitable analizar los lenguajes humanos semánticamente.

En el estado del arte actual, los sistemas representan el significado de las palabras con vectores de alta dimensión, es decir, con *word embeddings*. Dentro del campo de la semántica computacional, una nueva línea de investigación se centra en encontrar mapeos entre embeddings de diferentes idiomas (Mikolov et al., 2013b), (Smith et al., 2017), (Conneau et al., 2017).

Este trabajo de tesis propone un método inovador para encontrar mapeos lineales entre vectores de palabras para varios idiomas. En comparación con otros enfoques encontrados en la literatura, este método no aprende matrices de traducción entre dos idiomas específicos, sino entre un idioma determinado y un espacio universal compartido. Como datos de entrada, el sistema requiere vectores de palabras entrenados previamente y un diccionario de traducción de palabras para los idiomas dados.

Para los experimentos se usaron *fastText* embeddings (Conneau et al., 2017). Primero, el sistema fue entrenado entre dos idiomas aplicando dos conjuntos diferentes de entrenamiento: los datos de referencia de Dinu en inglés-italiano (Dinu et al., 2014), y los pares de traducción en inglés-italiano extraídos de la base de datos PanLex (Kamholz et al., 2014). Posteriormente, el sistema se entrenó en tres idiomas: inglés, italiano y español, al mismo tiempo utilizando pares de traducción multilingüe extraídos de la base de datos PanLex.

El sistema funciona en los idiomas inglés-italiano significativamente mejor que el sistema básico de Mikolov et al. (2013b), y proporciona un rendimiento comparable con los sistemas más sofisticados de Faruqui and Dyer (2014) y Dinu et al. (2014). Sin embargo, los sistemas del estado del arte actual son mucho mejores que el propuesto. Entrenar el sistema en tres idiomas al mismo tiempo arroja peores resultados que entrenarlo en los idiomas por parejas. Aprovechando la riqueza de la base de datos PanLex, el método propuesto permite aprender mapeos lineales entre pares de idiomas arbitrarios.

Abstract

Computer-driven natural language processing plays an increasingly important role in our everyday life. In the current digital world, using natural language for human-machine communication has become a basic requirement. In order to meet this requirement, it is inevitable to analyze human languages semantically.

Nowadays, state-of-the-art systems represent word meaning with high dimensional vectors, i.e. word embeddings. Within the field of computational semantics a new research direction focuses on finding mappings between embeddings of different languages (Mikolov et al., 2013b), (Smith et al., 2017), (Conneau et al., 2017).

This thesis work proposes a novel method for finding linear mappings between word vectors for various languages. Compared to previous approaches, this method does not learn translation matrices between two specific languages, but between a given language and a shared, universal space. As input data the system requires pre-trained word embeddings and a word translation dictionary for the given languages.

The *fastText* embeddings (Conneau et al., 2017) were used for the experiments. First, the system was trained between two languages applying two different training data; Dinu's English-Italian benchmark data (Dinu et al., 2014), and English-Italian translation pairs extracted from the PanLex database (Kamholz et al., 2014). Thereafter, the system was trained on three languages - English, Italian, and Spanish - at the same time using multilingual translation pairs extracted from the PanLex database.

The system performs on English-Italian languages with the best setting significantly better than the baseline system of Mikolov et al. (2013b), and it provides a comparable performance with the more sophisticated systems of Faruqui and Dyer (2014) and Dinu et al. (2014). Current state-of-the-art systems, however, are still much better than the proposed one. Training the system on three languages at the same time gives worse results than training it on the languages pairwise. Exploiting the richness of the PanLex database, the proposed method makes it possible to learn linear mappings between arbitrary language pairs.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The aim of this chapter is to summarize the motivation of this work and to review the tasks of the field of Natural Language Processing (NLP).

1.1 Natural Language Processing

NLP is a vibrant interdisciplinary field with many different names, all reflecting a different aspect of it. It is often referred to as speech and language processing, human language technology, computational linguistics, or speech recognition and synthesis. The main goal of this field is to make computers capable of using human languages as a communication protocol between machines and human users.

NLP is a complex field of study since it deals with what is considered to be one of the most delicate characteristics of human beings: human languages. This field is strongly connected with artificial intelligence since humans conceive the world mainly in terms of human languages.

Although they are nowhere near as fast as digital channels, human languages are still a very effective way of communication. When one says only the minimum message the listeners can fill in the rest with their world and common knowledge, and can easily figure out the missing or misunderstood parts from the context of the situation. They are also able to resolve ambiguities, homonyms etc. without even noticing it. Nonetheless, for a computer these tasks are not trivial at all.

Computer integrated human language communication has gone as far as assigning truly intelligent machines the ability of being capable of processing language as skillfully as humans do. This idea was first introduced in the 1950s by Alan Turing who proposed what came to be known as the Turing test.

To get a more detailed overview of what NLP is about, interested readers are encouraged to consult Dan Jurafsky's *Speech and language processing* book (Jurafsky and Martin, 2017). For those who prefer video lectures, the course *Natural Language Processing with Deep Learning* held by Christopher Manning and Richard Socher, professors of the Stanford University School of Engineering, can

give a deeper insight into this topic. This course is available on YouTube[1].

1.1.1 Common tasks of NLP

NLP comprises a wide variety of tasks. Some of them like spam detection, part-of-speech (POS) tagging, or named entity recognition are considered to be mostly solved problems. Applications for these tasks are now out in the market and they are usually integrated into smart devices even by default.

Great progress has recently been made with other tasks, which implies the existence of already fair enough applications but means that research work is yet to be done. Among them there are tasks such as sentiment analysis, words sense disambiguation, syntactic parsing, and machine translation, just to mention a few of them.

What is still considered to be quite challenging is to understand the meaning of a text. There are numerous tasks where dealing with semantics is inevitable in order to make relevant progress. Such tasks include question answering, dialogues, summarization, paraphrases, or text inference, just to mention a few.

1.1.2 Motivation for NLP research

Nowadays NLP technologies are becoming more and more integrated into our everyday life. With the advent of smart phones the importance of language has gone even further. These devices have small and rather inconvenient keyboards, thus speech-driven communication seems very appealing. Big companies such as Amazon, Apple, Facebook, or Google are all releasing products that use natural languages, a.k.a. human languages, to communicate with users. Since this thesis aims to contribute to the research field of word meaning and universal semantic representations, only those applications are listed below that can directly take advantage of these contributions.

Speech-driven assistance applications can make our everyday life more enjoyable, more comfortable and more convenient. They already help children develop delicate skills and they provide an immense amount of help for elderly people or people living with disabilities. These systems are using speech input for which first automatic speech recognition technologies have to be applied. But after that, in order to understand the goal of the user, a semantic analysis must be run as well.

An early version of conversational agents and certain strongly domain-based chatbots are already out on the market, providing 24 hour, immediate assistance for customers. By letting computers do the monotone and non-creative tasks employees could have more interesting jobs, tasks that only humans are able to do, or their working hours could be decreased, either of which would greatly benefit society (Economist, 2017).

[1] `https://www.youtube.com/playlist?list=PL3FW7Lu3i5Jsnh1rnUwq_TcylNr7EkRe6`

Advances in machine translation have already created a world where non-English speakers can also enjoy the benefits of the English-based web services. Generally, it can be said that for widespread languages machine translation has already reached a fairly usable state, for rare languages, however, it is still facing difficulties.

There are also numerous Web related tasks that are strongly reliant on the semantic analysis of the text. One promising application would be Web-based question answering which can be considered as an extended version of the classical Web search. Instead of searching just for key words complete questions could also be used when communicating with the search engine, just like in the case of human-to-human communication (Radev et al., 2005). For all these applications, however, it is inevitable to look beyond the syntactic surface and dig deeper into the underlying semantics.

1.2 Thesis objectives

The main focus of this study is word meaning. Given the need for robust representations for many languages, the question of whether human conceptual structure is universal has recently gained interest not only among cognitive scientists, (Rosch and Mervis, 1975), (Lakoff, 2008), (Gärdenfors, 2004), but among computational linguists as well. Youn et al. (2016) showed that human conceptual structure is independent of certain non-linguistic factors such as geography, climate, topology or literary traditions. Based on such findings this work proposes a procedure to construct a universal semantic representation in the form of translation matrices that serve to map each language to a universal space. As for pre-trained word vectors the *fastText* word embedding, discussed in Section 3.1.1, is used, which contains word vectors for 294 languages. During the training process a set of word translation pairs extracted from various gold dictionaries are aligned. These dictionaries involve Dinu's data (Dinu et al., 2014), discussed in Section 3.1.2, on the one hand, and the PanLex database (Kamholz et al., 2014), discussed in Section 3.1.3, on the other hand.

1.3 Thesis results

First, the system is trained and tested on the train and test sets proposed by Dinu et al. (2014). This data, which has recently become a benchmark data on word translation tasks, contains English-Italian word translation pairs. The proposed method reaches significantly better results on this benchmark data than Mikolov's baseline system (Mikolov et al., 2013b), both on the English-Italian and on the Italian-English tasks. Furthermore, these results are also comparable with the performance of more elaborated systems, like the system of Faruqui and Dyer (2014) or Dinu et al. (2014). Next, the model is trained on English-Italian word translation pairs extracted from the PanLex database (Kamholz et al., 2014). Results show that the system performs better when it is trained

on Dinu's data than when training it on PanLex translation pairs. Finally, further dictionaries containing three languages - English, Italian, and Spanish - are extracted from the PanLex database, and the system is trained on all of them at the same time. Experiments show that results are worse when the system is trained using all the three languages at the same time than when it is trained on the languages pairwise.

1.4 References

The code of our system is available on Github[2]. The whole code base was implemented by the author of this thesis except for an earlier version of the script which extracts translation pairs from the PanLex database[3], which was implemented by the supervisor of this thesis, Gábor Recski.

1.5 Document structure

The thesis is structured as follows:

- **Chapter 1** briefly explains the goals and the motivation of this work and introduces the research field of NLP. It also summarizes the main contributions and the results of this thesis work.
- **Chapter 2** discusses the state-of-the-art semantic word representations, the word embeddings. It briefly presents the standard *word2vec* learning procedure for monolingual word vectors and it introduces the concept of multilingual word embeddings.
- **Chapter 3** describes the available resources for multilingual embedding learning that were utilized during this work. It also introduces the proposed model in detail. It explains the learning procedure and the basic infrastructural and architectural features of the implemented system.
- **Chapter 4** presents all the experiments. It summarizes the results and compares them with the performance of other systems.
- **Chapter 5** summarizes the contributions of this thesis and describes future work. It suggests follow-ups which could not be included here due to time limitations, or which are beyond the scope of this thesis work.

[2] `https://github.com/Eszti/dipterv`
[3] `https://github.com/Eszti/dipterv/blob/master/panlex/scripts/panlex/extract_tsv.py`

Chapter 2

Word embeddings

2.1 Semantic encoding of words

Within the field of natural language processing a more specific area concentrates on semantic representations which are being leveraged both by classical semantic tasks such as question answering or chatbots and by other NLP tasks which in the strict sense of the word are not considered semantic tasks such as machine translation or syntactic parsing. A crucial part of all semantic tasks is to have a proper word representation which is capable of encoding the meaning as well.

One way to build a semantic representation is to use a distributional model. The idea is based on the observation that synonyms or words with similar meanings tend to occur in similar contexts, or as it was phrased by Firth in 1957: "You shall know a word by the company it keeps" (Firth, 1957). For example, in the following two sentences *"The cat is walking in the bedroom"* and *"A dog was running in a room"* words like *"dog"* and *"cat"* have exactly the same semantic and grammatical roles therefore we could easily imagine the two sentences in the following variations: *"The dog is walking in the bedroom"* and *"A cat was running in a room"* (Bengio et al., 2003). Based on this intuition, what distributional models are aiming to do is to compute the meaning of a word from the distribution of words around it (Jurafsky and Martin, 2017). The obtained meaning representations are usually high dimensional vectors, called word embeddings, which refer to their characteristic feature that they model a world by embedding it into a vector space.

One such model was first introduced by Bengio et al. (2003), whose primary purpose, though, was to construct a novel language model. Language modelling is the task of learning the joint probability function of word sequences in a given language. It is usually done by n-grams which are predicting the probability of a word in a sequence given the N previous ones. By increasing the number of words in the language, i.e. increasing the vocabulary size, the number of probabilities to learn grows exponentially. This problem is often called the "curse of dimensionality". Bengio et al. was the first to suggest applying a multilayer neural network for learning language models. The network consisted of input, projection, hidden, and output layers shown in Figure 2.1. The network

was fed by the N previous words in the sequence. At the input layer every word was represented by a vector using 1-of-V encoding, a.k.a one-hot encoding, where V denotes the size of the vocabulary. 1-of-V encoding vectors have a length of V, with all values being 0, except for one that corresponds to the given word of the vocabulary. They obtained a distributed representation for each word along with a probability function for word sequences. This probability function could predict never seen sentences as well if they were made of words with similar representations. The obtained word representations were feature vectors, having much smaller number of features than the size of the vocabulary. For the vocabulary they used 17K words, which means that the neural network was fed with 17K dimensional vectors, and for the number of features they ran experiments with 30, 60, and 100 features. These feature vectors can be regarded as an early version of a word embedding. These days word vectors usually have a dimension of 300 to 1000. With their proposed model Bengio et al. not only managed to reduce the dimension of the vectors encoding words, but they obtained a more meaningful word representation as well. This approach improved the state-of-the-art n-gram models with differences between 10 and 20 % in perplexity, both on a smaller, containing 1 million words, and on a larger, containing 15 million words, corpus.

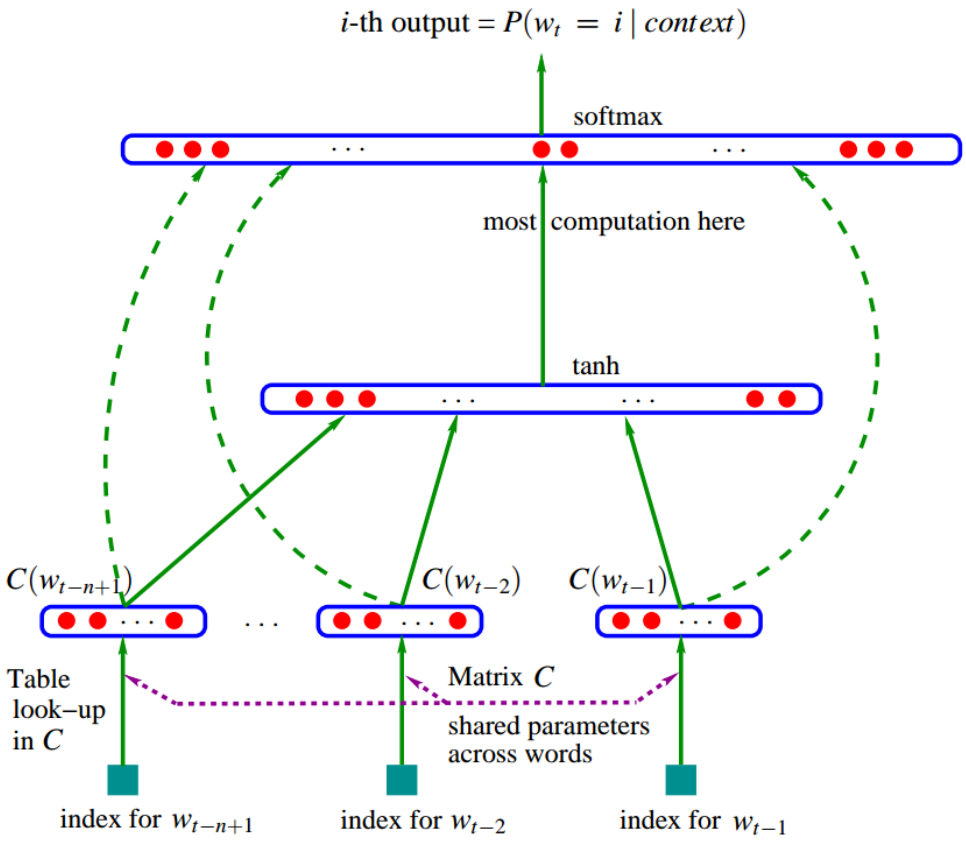

Figure 2.1: Network architecture proposed by Bengio et al.

Mikolov et al. (2013c) showed that the characteristics of word embeddings go well beyond syntac-

tic regularities. They showed that applying simple vector operations, for example vector addition and subtraction, can often produce meaningful results. For example, it was shown that if *vector("King") - vector("Man") + vector("Woman")* is calculated the result vector is the one closest to the vector representation of the word *Queen*. Moreover, state-of-the-art results on word similarity tasks are all held by word embeddings, where the similarity of two words is measured by the normalized dot product of the two corresponding word vectors. This measure is called the cosine similarity of words.

Another way to build semantic representations is to utilize lexical databases. In some previous works of the research team a hybrid system was created, which leveraged both the *4lang* othologycal model described in (Kornai, 2010), (Kornai, 2012), and (Acs et al., 2013) and various distributional models, i.e. various word embeddings. This system reached a state-of-the-art score (Recski et al., 2016) on the *SimLex-999* benchmark data (Hill et al., 2015).

The following sections describes the basic procedure of training word embeddings and, following that, it focuses on multilingual word embeddings, a more specific field of computational semantics.

2.2 Models for learning word embeddings

Mikolov et al. (2013a) suggested a Bag-of-words Neural Network, more specifically the following two architectures. The first one, denoted as the Continuous Bag-of-Words Model (CBOW) tried to predict the current word based on the context, whereas the second one, denoted as the continuous skip-gram model tried to maximize the classification of a word based on another word in the same sentence. Both models worked better than the model suggested by Bengio et al. (2003) both on semantic and syntactic tasks, while between the two models of Mikolov the CBOW turned out to be slightly better on syntactic tasks and the skip-gram on semantic tasks. Mikolov's procedure has become known as the *word2vec*[1] procedure. The architecture of the CBOW and the skip-gram models are shown in Figure 2.2.

Embeddings are usually evaluated on word similarity and word analogy tasks. Besides providing quite promising results on them, they have also been applied to many downstream tasks, from Named Entity Recognition (NER) and chunking (Turian et al., 2010) to dependency parsing (Bansal et al., 2014). It has furthermore been shown that weakly supervised embedding algorithms can also lead to huge improvements for tasks like sentiment analysis (Tang et al., 2014).

2.3 Multilingual word embeddings

The aim of this section is to describe the importance of multilingual word embeddings. It also explains how it is possible to incorporate word embeddings trained on monolingual text corpora into

[1] `http://deeplearning4j.org/word2vec`

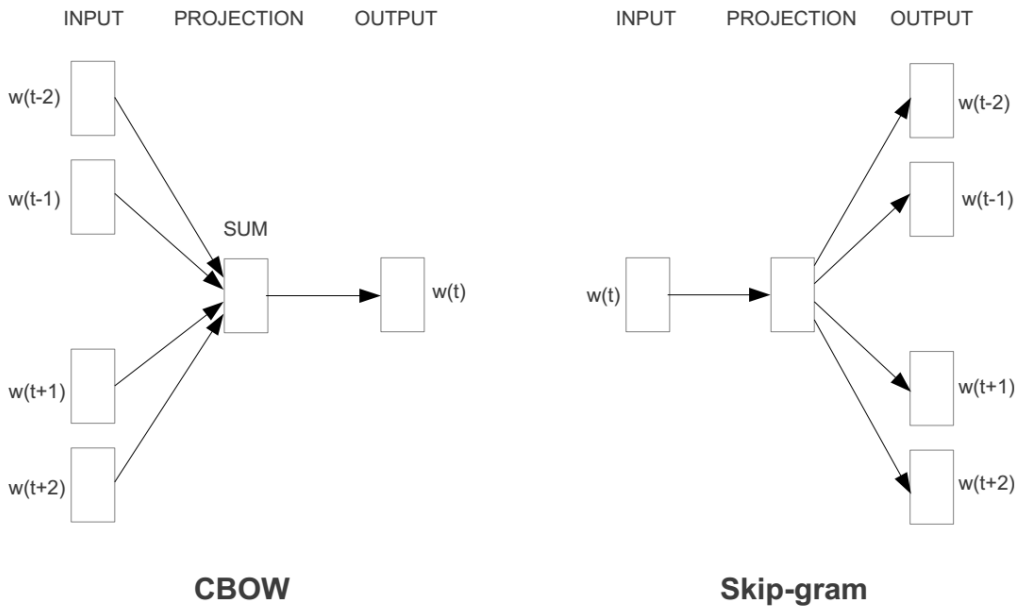

Figure 2.2: Bag-of-words neural networks suggested by Mikolov et al.

a multilingual context. After that, a brief summary is presented about the previous attempts on constructing cross-lingual word vector representations.

2.3.1 Motivation

The question how to model representations is a highly interdisciplinary issue to discuss. Within cognitive science, traditionally there are two dominating approaches to this problem. The first one is a *symbolic* one which states that cognitive systems can be described as Turing machines. The second one, denoted as *associationism*, says that representations are associations among different kinds of information elements. In his book, *Conceptual Spaces: The Geometry of Thought* Gärdenfors (2004) advocates a third approach, which he calls *conceptual* from. This representation is based on using geometrical structures rather than symbols or connections among neurons.

To go a step further one could ask whether these structures are universal among all human beings. Approaching this question with the eyes of a computer scientist this problem might be formulated as whether it is possible to model meaning universally, i.e. independently of language. Current meaning representations are leaned from monolingual corpora, and therefore infer language dependency. But is there a way to find one single representation instead of a different one for each and every human language?

Youn et al. (2016) suggested that the human brain may reflect distinct features of cultural, historical, and environmental background in addition to properties universal to human cognition. They provided an empirical measure of semantic proximity between concepts using entries of the Swadesh list (Swadesh, 1952). The Swadesh list is a cross-linguistic dictionary which includes a

110- and a 207-item list of basic concepts in approximately 2000 languages. Youn et al. took 22 concepts of this list that refer to material entities (e.g. STONE, EARTH, SAND, ASHES), celestial objects (e.g., SUN, MOON, STAR), natural settings (e.g., DAY, NIGHT), and geographic features (e.g., LAKE. MOUNTAIN). Then, they applied translation and back-translation through various languages. As a result of numbers of polysemies in the resulting graph originally distinct concepts become connected. For example the Spanish word CIELO in English both means HEAVEN and SKY. Thus by applying English-Spanish-English translation and back-translation the two English words HEAVEN and SKY become connected. The more such polysemous words are found, the stronger such connections become. For example, if besides Spanish, we also apply the translation and back-translation through German, the same polysemy appears: the German word HIMMEL in English both means HEAVEN and SKY, just like the Spanish word CIELO. The procedure is shown on Figures 2.3 and 2.4.

Statistical analysis of the obtained graphs constructed over the polysemies observed in the above-mentioned 22-word-long subset of basic vocabulary showed that the structural properties of these graphs are consistent across different language groups, and largely independent of geography, environment, and the presence or absence of literary traditions. Based on these findings it seems reasonable to assume that the structure of meaning, at least to a certain extent, is universal. Therefore representing semantics at universal level seems to be a valid approach.

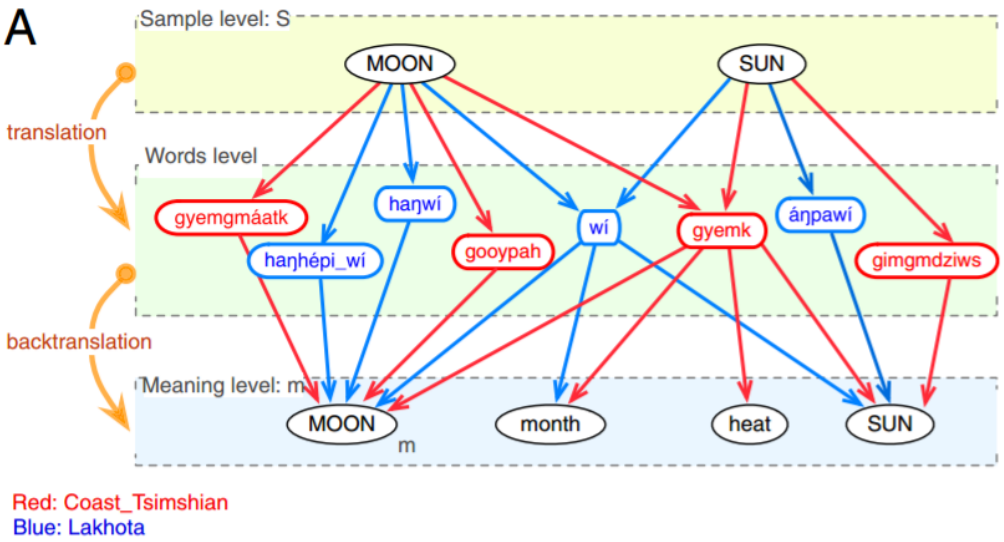

Figure 2.3: Translating MOON and SUN through polysemous words.

2.3.2 Tasks

Beyond the theoretical level of whether meaning is universal there are numerous practical problems for which cross-lingual embeddings might be useful. In this section different tasks are proposed, where solutions can be facilitated by utilizing multi-lingual embeddings.

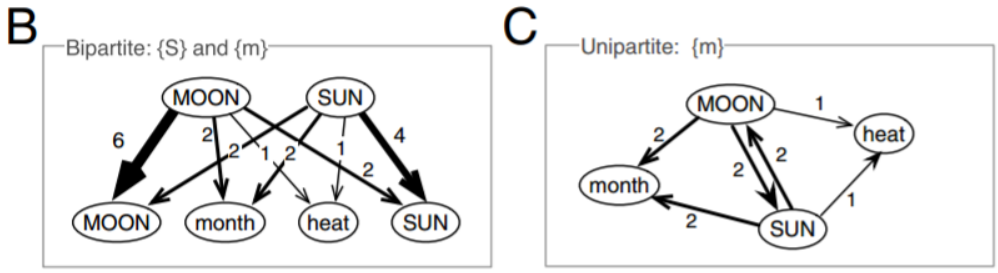

Figure 2.4: Making links between English concepts through eliminating the internal nodes.

2.3.2.1 Cross-language part-of-speech tagging

POS tagging is the task for annotating a text with part-of-speech tags. The fundamental idea behind the multilingual learning of part-of-speech tagging is that when assigning part-of-speech tags the patterns of ambiguity differ across languages. A word with part-of-speech tag ambiguity in one language may correspond to an unambiguous word in the other language. For example, the word “can” in English may function as an auxiliary verb, a noun, or a regular verb; however, after translating the sentence into other languages, the different meanings of ”can” are likely to be expressed with different lexemes. By combining natural cues from multiple languages, the structure of each POS tagger becomes more apparent (Naseem et al., 2009).

2.3.2.2 Cross-language super sense tagging

SuperSense Tagging is the problem of assigning ”supersense” categories, for example person or act, to the senses of words according to their context in large scale texts. Opposite to Named Entity Recognition (NER) systems a Super Sense Tagger does not make a difference between proper and common names. These ”supersense” categories include general concepts defined by WordNet (Beckwith et al., 1991), which originally introduces 45 lexicographer’s categories (Fellbaum, 1998).

Attempts for creating such systems have already been made. For example Picca et al. (2009) trained a multilingual super sense tagger on the Italian and English languages. Despite the fact that they did not use any word embeddings, the introduction of multilingual word embeddings to this task could significantly facilitate the development of multilingual knowledge induction, ontology engineering, and knowledge retrieval.

2.3.2.3 Machine translation

Machine translation is the task of translating a text automatically with a computer from a source language to a target language. Current translation models often fail to generate good translations for infrequent words or phrases. Previous works tried to improve this by inducing new translation rules from monolingual data with a semi-supervised algorithm. Nevertheless, this approach does not scale very well since it is computationally quite expensive. Zhao et al. (2015) proposed a much

faster and simpler method that creates translation rules for infrequent phrases based on phrases with similar continuous representations, i.e. with similar word vectors, for which a translation is known. Their method improved a phrase-based baseline by up to 1.6 BLEU on Arabic-English translation, and it was three-orders of magnitudes faster than existing semi-supervised methods and 0.5 BLEU more accurate.

By introducing a universal vector space, in order to cover all possible translation pairs for n languages, instead of having to train $\binom{n}{2}$ translators it would be enough to train only $2n$ translators, for each language from the source space to the universal space and vica versa. This would significantly simplify the Machine Translation task.

2.3.2.4 Under-resourced languages

Dictionaries and phrase tables are the basis of modern statistical machine translation systems. Mikolov et al. (2013b) showed a method that can automate the process of generating and extending dictionaries and phrase tables. They could translate missing word and phrase entries by learning language structures based on large monolingual data and mapping between languages from small bilingual data. This is a powerful opportunity for rare languages to join the mostly English-based world of the Web and for non-English speakers to enjoy its benefits without having to speak English.

2.3.3 Applications

Facebook has already made use of multilingual embeddings (Stoyanov and Ayan, 2018). To better serve their community they offer features like Recommendations and M Suggestions in many languages. These services are based on text classification, which refers to the process of assigning a predefined category from a set to a document of text. With language-specific NLP techniques, supporting a new language implies solving the problem once again from scratch. One way is to train a separate classifier for each language, which means collecting a separate, large set of training data every time. Collecting data is an expensive and time-consuming process, which becomes increasingly difficult when scaling it up to support more than 100 languages. Another way is to train only one classifier, for example an English one, and then, before applying this classifier for languages different from English, as a pre-processing step, the text will first be translated to English. This solution is prone to error propagation and, in addition, it involves an additional call to the translation service which leads to a significant degradation in performance.

Using multilingual embeddings in order to help applications scale to more languages is a great advantage. Since the words in a new language will appear close to the words in the trained languages in the embedding space, the classifier will be able to perform well on new languages as well. It is not necessary to call translation services, therefore it does not affect the performance either.

2.4 State-of-the-art multilingual embedding models

This section presents a brief history on cross-lingual word vector representations. First the baseline approach of Mikolov et al. (2013b) is described, and then various attempts are studied, which intended to improve this baseline system and to alleviate its errors. Finally, some recent attempts are summarized, which aimed to obtain multilingual word embeddings without using any parallel data.

2.4.1 First attempt: Mikolov et al. (2013b)

Right after publishing their *word2vec* procedure, Mikolov et al. went even further by noticing that continuous word embedding spaces exhibit similar structures across languages. They applied a simple two-step procedure:

- Firstly, monolingual models of languages using huge corpora were built, e.g. by using the *word2vec* method.
- Secondly, a small bilingual dictionary was used to learn linear projection between the languages. These words are often referred to as anchor points. The optimization problem was the following:

$$\min_{W} \sum_{i=1}^{n} ||Wx_i - z_i||^2 \tag{2.1}$$

  where W denotes the transformation matrix, and $\{x_i, z_i\}_{i=1}^{n}$ are the continuous vector representations of word translation pairs, with x_i being in the source language space and z_i in the target language space.
- Finally, at test time, any word can be translated from the source language by projecting its source language vector representation to the target language space. Once the vector in the target language space is obtained, the most similar word vector can serve as the output of the translation. The percentage of how many times the right translations are among the N closest words is called precision@N.

Applying only the translation matrices, they achieved 51% precision@5 score for translation of words between English and Spanish. To obtain dictionaries first they created monolingual corpora from the WMT11[2] text data. Then they took the most frequent words from these monolingual source datasets and translated them using on-line Google Translate. Beside simple words, they also used short phrases as dictionary entries. In addition to the promising result on the English-Spanish

[2] `http://www.statmt.org/wmt11/training-monolingual.tgz`

word translation task, this method seemed to be working even for distant language pairs like English and Vietnamese as well.

The simple procedure of Mikolov et al. also serves as a guideline to follow for constructing new multilingual word vector models. Most of the various improvements described below proposed different procedures for the second step. This thesis also proposes a novel way of finding the linear projections for Mikolov's second step using different datasets.

2.4.2 Improvements of Mikolov's model

Since Mikolov's experiments various attempts have been made to improve the cross-lingual embeddings. Below, the basic ideas of these methods and their results are summarized.

2.4.2.1 Faruqui and Dyer (2014)

Faruqui and Dyer proposed a procedure to obtain multilingual word embeddings by concatenating the two word vectors coming from the two languages. This procedure, however, has significant drawbacks, such as increases in dimension, the introduction of irrelevant data, the incapacity of generalization across languages, and the handling of out of vocabulary words. To counter these problems, they used canonical correlation analysis (CCA), which is a way of measuring the linear relationship between two multidimensional variables. For each of the two variables it finds a projection vector that is optimal with respect to correlations. The great advantage of this procedure is that these new projection vectors preserve or even reduce the dimansionality. The obtained multi-lingual embeddings were tested on the following four different standard word similarity tasks:

- On the WS-353 dataset (Finkelstein et al., 2001), which contains 353 pairs of English words that have been assigned similarity ratings by humans. This dataset was later further divided into two different fragments *similarity*, WS-SIM, and *relatedness*, WS-REL by Agirre et al. (2009) who claimed that these two are different kinds of relations and should be dealt with separately

- On the RG-65 dataset which contains 65 pairs of nouns ranked by humans (Rubenstein and Goodenough, 1965).

- On the MC-30 dataset which contains 30 pairs of nouns ranked by humans (Miller and Charles, 1991).

- On the MTurk-287 dataset (Radinsky et al., 2011) consisting of 287 pairs of words, which has been constructed by crowdsourcing the human similarity ratings using Amazon Mechanical Turk.

These word representations obtained after using multilingual evidence performed significantly better on the above-mentioned evaluation tasks compared to the monolingual vectors. The method was more suitable for semantic encoding than for syntactic encoding. As a conclusion, it was shown that multilingual evidence is an important resource even for purely monolingual applications.

2.4.2.2 Xing et al. (2015)

Xing et al. showed that bilingual translation can be largely improved by normalizing the embeddings and by restricting the transformation matrices into orthogonal ones.

In order to compare their results with Mikolov et al. (2013b), they largely followed their settings to create an English-Spanish dictionary. After extracting the monolingual datasets from the WMT11 corpus they selected the 6000 most frequent words in English and employed Google's online translation service to translate them into Spanish. The resulting 6000 English-Spanish word pairs were used to train and test the obtained bilingual transformation matrices using cross validation. First they reproduced Mikolov's results and then they showed that their method outperformed those results with approximately 10 % on this English-Spanish setting. The exact numbers are shown in Table 2.1.

	eng - ita	
Precision	@1	@5
Mikolov et al. (2013b)	33%	51%
Mikolov on Xing's data	30.43%	49.43%
Xing et al. (2015)	**38.99%**	**59.16%**

Table 2.1: Comparing Mikolov's results with Xing's. The first row shows results reported in (Mikolov et al., 2013b), the second row contains the numbers obtained by Xing using Mikolov's method, and the last row presents the results of Xing's procedure. Experiments of the last two rows were carried out on the exact same dataset. The original dataset that Mikolov experimented with was not published.

2.4.2.3 Dinu et al. (2014)

Dinu et al. studied the phenomenon of hubs. He showed that the neighbourhoods of the mapped vectors are strongly polluted by hubs, which are vectors that tend to be near a high proportion of items. Thus their correct labels will be pushed down in the neighbour lists when looking up for word translations. They proposed a method that computes hubness scores for target space vectors and penalizes those vectors that are close to many words, i.e. hubs are down-ranked in the neighbouring lists. The experiments were carried out on an English-Italian dataset created by themselves and discussed in detail in Section 3.1.2.

2.4.2.4 Lazaridou et al. (2015)

Lazaridou et al. studied some theoretical and empirical properties of a general cross-space mapping function, and tested them on cross-linguistic (word translation) and cross-modal (image labelling)

tasks. By introducing negative samples during the learning process they could reach state-of-the-art results on Dinu's English-Italian word translation task. Settings for the negative examples were studied both by choosing them randomly and by choosing "intruders" which are near the mapped vector, but far from the actual gold target space vector. The "intruder" approach achieved better results, and was able to do so even after a few training epochs.

2.4.2.5 Ammar et al. (2016)

Ammar et al. proposed methods for estimating and evaluating embeddings of words in more than fifty languages in a single shared embedding space. Since English usually offers the largest corpora and biligual dictionaries, they used the English embeddings to serve as the shared embedding space. First they introduced a multilingual clustering approach called *MultiCluster*. Then, they extended various bilingual methods for multilingual usages, such as Faruqui's CCA procedure, which they called *MultiCCA*, or the method of Luong et al. (2015), which they called *MultiSkip*. Finally they experimented with another procedure called *translation-invariance*, which was proposed by Huang et al. (2015).

The *MultiCluster* and *MultiCCA* methods were tested on 59 languages, while the *MultiSkip* and *translation-invariance* methods on only 12 languages for which high-quality parallel data was available. For the 12 languages the bilingual dictionaries were extracted from the Europarl parallel corpora, while for the remaining 47 languages, dictionaries were formed by translating the 20k most common words in the English monolingual corpus with Google Translate.

This thesis also proposes a method which is capable of projecting multiple number of languages into a single, shared embedding space. This procedure, however, instead of taking the English embedding as the shared space, it projects all the different embeddings into an independent, universal space.

2.4.2.6 Artetxe et al. (2016)

Artetxe et al. built a generic framework that generalizes previous works made on cross-linguistic embeddings. Procedures of Mikolov et al. (2013b), Faruqui and Dyer (2014), and Xing et al. (2015) were implemented as part of their framework. For evaluating the methods they used the same English-Italian dataset by Dinu, discussed in Section 3.1.2. As a conclusion they published that of the proposed methods with the best overall results were the ones with orthogonality constraint and a global pre-processing with length normalization and dimension-wise mean centering. Table 2.2 shows their result summary.

2.4.2.7 Smith et al. (2017)

Smith et al. also proved that translation matrices should be orthogonal. They applied singular value decomposition (SVD) to achieve this. Besides, they introduced a novel "inverted softmax" method

	eng - ita
Precision	@1
Mikolov et al. (2013b)	34.93%
Xing et al. (2015)	36.87%
Faruqui and Dyer (2014)	37.80%
Artetxe et al. (2016)	**39.27%**

Table 2.2: Artetxe's summary on Dinu's data (Artetxe et al., 2016)

for identifying translation pairs, with which they improved the precision of Mikolov. Orthogonal transformations also turned out to be more robust to noise, which made it possible to learn the transformations without expert bilingual resource by constructing a "pseudo-dictionary" from the identical character strings.

For evalutaion they also used Dinu's English-Italian setting. In order to compare their method with the previous ones they reproduced the previous experiments both in English-Italian and Italian-English directions and published a summary in the form of tables that are presented here as Table 2.3 and Table 2.4. All the methods turned out to be more accurate when translating from English to Italian. This is not surprising at all, given the fact that many English words can be translated to either the male or female form of the Italian word.

Precision	**@1**	**@5**	**@10**
Mikolov et al. (2013b)	0.338	0.483	0.539
Faruqui and Dyer (2014)	0.361	0.527	0.581
Dinu et al. (2014)	0.385	0.564	0.639
Smith et al. (2017)	**0.431**	**0.607**	**0.664**

Table 2.3: English to Italian results on Dinu's data published by Smith

Precision	**@1**	**@5**	**@10**
Mikolov et al. (2013b)	0.249	0.410	0.474
Faruqui and Dyer (2014)	0.310	0.499	0.570
Dinu et al. (2014)	0.246	0.454	0.541
Smith et al. (2017)	**0.380**	**0.585**	**0.636**

Table 2.4: Italian to English results on Dinu's data published by Smith

2.4.3 Models without parallel data

While all the above-mentioned methods rely on biligual word lexicons, most recent studies are aiming to eliminate the need for any parallel data at all. Smith et al. (2017) already made attempts for the alleviation of parallel data supervision by introducing character-level information, but the results were not on par with their supervised counterparts. In addition, these methods are strictly limited to language pairs sharing a common alphabet.

Conneau et al. (2017) introduced an unsupervised way for aligning monolingual word embedding spaces between two languages without using any parallel corpora. Their experiments showed that this method can be applied even for distant language pairs like English-Russian or English-Chinese.

On Dinu's benchmark setting they reported results with two different embeddings. First, they used word vectors trained on the WaCky datasets (Baroni et al., 2009), just like all previous systems did so far. Then, they experimented with embeddings trained on Wikipedia using their novel *fastText* method discussed in Section 3.1.1. Conneau's unsupervised method reached comparable results with Smith's supervised model when training it with the WaCky embeddings, but it performed significantly better when training it with their *fastText* embeddings. Their system reached state-of-the-art scores on Dinu's benchmark data, both in English-Italian and in Italian-English directions. Results are summarized in Table 2.5 and 2.6.

Precision	**@1**	**@5**	**@10**
Smith et al. (2017)	0.431	0.607	0.651
Conneau et al. (2017) - WaCky	0.451	0.607	0.651
Conneau et al. (2017) - fastText	**0.662**	**0.804**	**0.834**

Table 2.5: English to Italian results on Dinu's data published by Conneau

Precision	**@1**	**@5**	**@10**
Smith et al. (2017)	0.380	0.585	0.636
Conneau et al. (2017) - WaCky	0.383	0.578	0.628
Conneau et al. (2017) - fastText	**0.587**	**0.765**	**0.809**

Table 2.6: Italian to English results on Dinu's data published by Conneau

Chapter 3

Proposed model

This thesis work proposes an approach to learn translation matrices between distributional word vector spaces. The method requires multilingual pre-trained word embeddings and a multilingual gold dictionary containing word translation pairs. This chapter first describes the utilized multilingual resources and then discusses the approach in detail.

3.1 Multilingual data

This section briefly describes the data resources that were used during the experiments carried out within the scope of this work. These involve the pre-trained *fastText* embedding published by the Facebook AI research group and two gold bilingual dictionaries. One of them was constructed by Dinu et al. (2014) and the other was extracted from the PanLex database (Kamholz et al., 2014) as part of this thesis work.

3.1.1 The *fastText* embedding

The usual technique for obtaining continuous word representation, i.e. word embeddings, is to represent each word of the vocabulary by a distinct vector, without parameter sharing. Such vectors completely ignore the morphology of words which is a significant limitation especially for agglutinating languages, e.g. Hungarian. In these languages new words are formed by stringing together morphemes which leads to large vocabularies and many rare words.

The Facebook AI Research (Conneau et al., 2017) group proposed a new approach based on the skipgram model (Mikolov et al., 2013a), but this time, contrary to the previously mentioned methods, parameter sharing was applied and words were represented as a bag of character n-grams (Bojanowski et al., 2016). First, a vector representation was associated to each character n-gram. Next, the word vectors were constructed as the sum of these character n-gram representations. With this method they were capable of computing the vector representations of words previously unseen in the training data. Moreover, the procedure turned out to be faster than the previous ones as well. The model was evaluated both on word similarity and word analogy tasks. The results showed that

this model outperformed Mikolov's `CBOW` and `skipgram` baseline systems that did not take sub-word information into account. It also did better than methods relying on morphological analysis. Their pre-trained word vectors trained on Wikipedia are available for 294 languages[1].

3.1.2 English-Italian setup of Dinu

Dinu et al. (2014) constructed an English-Italian gold dictionary split into a train and a test set that is now being used as benchmark data for evaluating English-Italian word translation tasks. Both train and test translation pairs were extracted from a dictionary built from Europarl en-it[2] (Tiedemann, 2012).

For the test set they used 1,500 English words split into 5 frequency bins, 300 randomly chosen in each bin. The bins are defined in terms of rank in the frequency-sorted lexicon: [1-5K], [5K-20K], [20K-50K], [50K-100K], and [100K-200K]. Some of these 1500 English words have multiple Italian translations in the Europarl dictionary, so the resulting test set contains 1869 word pairs all together, with 1500 different English, and with 1849 different Italian words. See Table 3.1.

For the training set, the above-mentioned Europarl dictionary was first sorted by the English frequency, then the top 5k entries were extracted and care was taken to avoid any overlap with test elements on the English side. On the Italian side, however, an overlap of 113 words is still present. In the end the train set contains 5k word pairs with 3442 different English, and 4549 different Italian words. See Table 3.1.

Set	Language	No. words
train (5000 word pairs)	eng	3442
	ita	4549
test (1869 word pairs)	eng	1500
	ita	1849

Table 3.1: Statistics of word counts.

Below there is a list of the different categories of Italian overlaps:

- **Singular-plural correspondence:** In Italian when the last vowel of a substantive is accented, the plural form is the same as the singular. For example *comunità* and *attività*. See Table 3.2.

- **Italian word mistaken for English word:** The English translation is the same as the original Italian word. For example in the test set the Italian word *segni* is not translated and the same happens with *vecchi*. See Table 3.3.

- **Different verb forms:** The same Italian word can be translated into different English verb tenses. For example *sostenere*. See Table 3.4.

[1] `http://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md`
[2] `http://opus.lingfil.uu.se/`

- **Synonyms and homonyms:** One Italian word can be translated into several English words which are synonymous except in case of homonymy. This phenomenon is actually fairly understandable and acceptable under all circumstances. See Table 3.5.

- **Errors in the translation:** For example the plural form of Italian words *gatti* and *passeggeri* are translated both as the correct plural form and the incorrect singular form. See examples in Table 3.6.

Italian	English - train	English - test
comunità	communities	community
attività	activities	activity

Table 3.2: Singular-plural correspondence

Italian	English - train	English - test
segni	signs	segni
vecchi	old	vecchi

Table 3.3: Italian word is mistaken for English word

Italian	English - train	English - test
sostenere	support	supporting

Table 3.4: Different verb forms

Italian	English - train	English - test
risposte	answers	responses
sufficiente	sufficient	enough

Table 3.5: Synonyms and homonyms

Italian	English - train	English - test	Explanation
gatti	cat	cats	it only means cats
passeggeri	passengers	passenger	it only means passengers

Table 3.6: Errors in the translation

3.1.3 Panlex

PanLex (Kamholz et al., 2014) is a nonprofit organization that aims to build a multilingual lexical database from available dictionaries in all languages. As part of this thesis work gold data is extracted from this database, which is then used for the training of the proposed multilingual word embedding model.

English	Italian	Confidence values
Sarajevo	Sarajevo	9
euro	euro	9
simple	semplice	8
difficult	difficile	8
college	università	7
plausible	verisimile	7
sea	mare	6
sky	cielo	6
better	meglio	5
inform	informare	5
combustible	combustibile	4
office	ufficio	4
sorcerer	conscitore	3
it	ella	3
Great Wall of China	Grande muraglia cinese	2
factory workers	lavoratori dell'industria	2
stay	restare	1
sometimes	qualche volta	1

Table 3.7: Sample of PanLex entries of the extracted tsv file

3.1.3.1 Brief description of PanLex

The name PanLex is coming from the words *panlingual* and *lexical*, which reflect the main objective of this project: to collect word translations in possibly all languages. They are basically digitizing and centering the content of different, already existing dictionaries made by domain experts. Own translations are not accepted. To each translation pair a confidence value is assigned, which can be used for filtering the extracted data. These confidence values are in the range of [1, 9], with 9 meaning high and 1 meaning low confidentiality. The main purpose is to preserve the diversity of languages, so the collection of "threatened" or "endangered" languages, and dictionaries of rare language combinations are top priority,

PanLex also exhibits different *language varieties* that include, among others, regional variations and different writing systems. A *language variety* is denoted with a three-letter *language code*, e.g. `eng` for English, and with a three-digit *variety code*, e.g. `000`. To the most widely spoken variety of a language usually the `000` *variety code* is assigned. When extracting data from the PanLex database, in all cases, the *language variety* with the smallest *variety code* was taken. A script for extracting the translation pairs and creating a tab separated file (tsv) from them was implemented as part of this work[3]. A sample of the obtained tsv file is shown in Table **??**.

[3] `https://github.com/Eszti/dipterv/blob/master/panlex/scripts/panlex/extract_tsv.py`

3.2 Description of the proposed method

This section describes the proposed model in detail. First the metrics used during training and evaluation processes are defined. Then the equation used for optimizing is presented. Finally, some implementation issues are discussed.

In summary, this work proposes a novel method for learning linear mappings between word translation pairs in the form of translation matrices. These translation matrices learn to map pre-trained word embeddings into a universal vector space. During training the cosine similarity of word translation pairs is maximized, which is calculated in the universal space. After mapping the embeddings of two different languages into this universal space, the cosine similarity of the actual translation pairs should be high. At test time the system is evaluated with the precision metric, principally used for word translation tasks.

3.2.1 Cosine similarity and precision

This thesis combines two kinds of tasks, namely the word similarity and the word translation tasks. In word similarity tasks the extent to which the meanings of two words are similar is what is to be sought, while the objective of word translation tasks is to retrieve the right target language translations of words given in the source language. In this section the cosine similarity and the precision metrics are explained. The former, cosine similarity, is a measure for the performance of word similarity tasks, while the latter, precision, is used for the evaluation of word translation tasks.

3.2.1.1 Cosine similarity

Cosine similarity[4] is a measure of similarity between two non-zero vectors. It is calculated as the normalized dot product of two vectors, as shown in Equation 3.1. In fact, cosine similarity is a space that measures the cosine of the angle of two vectors. It is important to note that cosine similarity is not a proper distance metric, since the triangle inequality property does not apply. In word similarity tasks, however, this metric is used for measuring the similarity of two words represented as word vectors. Although cosine similarity values by definition are in range of [-1, 1], in word similarity tasks it is particularly used in positive space, [0, 1], where parallel vectors are similar and orthogonal vectors are dissimilar.

$$cos\_sim = \cos\theta = \frac{\vec{a} \cdot \vec{b}}{||\vec{a}|| \cdot ||\vec{b}||} \tag{3.1}$$

[4] https://en.wikipedia.org/wiki/Cosine_similarity

3.2.1.2 Precision

Precision is a metric used for measuring the performance of translator systems, which intend to learn to translate from a source language into a target language. On the target side a look-up space is defined, which could, for example, correspond to the most frequent 200k words of the target language, as in our experiments. After translating a word, the N word vectors of the look-up space that are closest to the translated one are regarded. The Precision @N metric denotes the percentage of how many times the real translation of a word is found among the N closest word vectors in the look-up space. Usual N values are 1, 5, and 10.

3.2.2 Equation to optimize

The objective of the proposed method is to learn linear mappings in the form of translation matrices that are obtained by maximizing the cosine similarity of gold word translation pairs in a universal space. Therefore, for each language one single translation matrix is searched that maps the language from its original vector space to the universal one.

The method tries to bring the translation pairs close together in a shared, universal space. Therefore, it is not only applicable for language pairs but for any number of languages as well. The main advantage is that by introducing new languages the number of the learned parameters remains linear to the number of languages since instead of learning pair-wise translation matrices, for each language only one matrix is learned, the one that maps directly to this shared, universal space.

Let L be a set of languages, and TP a set of translation pairs where each entry is a tuple of two in the form of (w_1, w_2) where w_1 is a word in language L_1 and w_2 is a word in language L_2, and both L_1 and L_2 are in L. Then, let's consider the following equation:

$$\frac{1}{|TP|} \cdot \sum_{\substack{L_1, L_2 \\ \in L}} \sum_{\substack{(w_1, w_2) \\ \in TP}} cos\_sim(w_1 \cdot T_1, w_2 \cdot T_2) \tag{3.2}$$

where T_1 and T_2 are translation matrices mapping L_1 and L_2 to the universal space. Since the equation is normalized with the number of translation pairs in the TP set, the optimal value of this function is 1. Off-the-shelf optimizers are programmed to find local minimum values, and therefore the loss function is multiplied by -1 so that it will be a minimization task.

It should be noted that if w_1 and w_2 values are normalized, as Xing et al. suggested, the *cos_sim* reduces to the simple dot product of the translated vectors. During the experiments the word vectors are always normalized. At test time the system is evaluated with the precision metric, more specifically with Precision @1, @5, and @10. The distance assigned to the word vectors in the look-up space is the *cos_sim*.

3.3 Properties of the training process

This section discusses the issues of the training process. First, it briefly describes a general machine learning process and then the various parameters of the implemented system are discussed.

3.3.1 Machine learning in summary

Machine learning is a generic procedure which enables computers to learn a specific task, without being explicitly programmed how to do it. The task is formulated as an objective function of various parameters. The aim of the learning process is to find the optimal values of these parameters.

Machine learning can be applied as an iterative process, which is how it was applied in this work. Over the training process in each iteration some data is fed to the system. At the beginning, the parameters of the objective function are initialized with random values. Then, in each iteration, based on the given data, these parameters are updated, so that the value of the objective function would get closer to its optimum. At the update step the parameters are modified by a value proportional to the derivative of the objective function.

In statistical learning it is essential for a system to be capable of generalization, i.e. to be able to perform well on new data as well. Therefore, after the training process the performance of these systems is measured on an independent dataset. The dataset used for training is called the *training* set, and the one used for testing is called the *test* set. The learning process itself has several hyper parameters, such as the learning rate which adjusts the speed of the learning process. These hyper parameters also need to be tuned through various experiments. During these experiments the system is trained on the *training* set and tested on the so-called *development* or *validation* set. Then, usually, with the best hyper parameter setting the system is trained once more on the union of the previous *training* and *development* sets, and then, it is tested one last time on the independent *test* set. The obtained results are regarded as the real performance of the system.

Tensorflow[5] is an open source software library for machine learning developed by Google, which was used for the implementation of the proposed method as well.

3.3.2 Adjustable parameters

There are several configuration parameters that need to be adjusted during the training process using the development set. This section first describes the generic parameters that are used in all training processes. Next, more specific parameters are discussed that are special properties of the implemented system. In this section there is only a list of the parameters, their actual adjustment process is discussed in Section 4.

[5] `https://www.tensorflow.org/`

3.3.2.1 Generic parameters

A machine learning process has several hyper parameters that can be adjusted. Only those are listed below that were tuned during the experimentation phase of this thesis work:

- **optimizer:** This is the method for finding the optimum value of the objective function. The most common optimizers are Stochastic Gradient Descent (SGD), Adagrad, Adadelta, Adam, and Adamax.
- **epochs:** One epoch is the number of iterations after which every example of the training set was seen exactly once. The more epochs are done, the more the system has learned.
- **batch size:** This is the number of training examples that are given to the system in one iteration. This number varies from 1 to the number of all training examples. Since in one iteration the parameters are updated only once, the following general rule applies: by using a smaller batch size the system needs to be trained for less epochs compared to choosing a bigger size. When applying the SGD optimizer, by convention, Batch Gradient Descent (BGD) refers to the setting when one batch includes all the training data, and Mini-Batch Gradient Descent (MBGD) is used for a batch size of one training example (Ng, 2016).
- **learning rate:** This parameter controls the speed of the learning process. A higher learning rate means a faster learning process, since steps taken towards the local optimum are bigger. The drawback, though, is that the actual optimum value can easily be missed. A lower learning rate can overcome this problem, but then the learning process takes longer. It is important to find the balance between accuracy and speed in the parameter adjustment experimental phase.
- **Batch size - learning rate relation:** Goyal et al. (2017) studied the behaviour of different batch size and learning rate combinations, running their experiments on the ImageNet database (Russakovsky et al., 2015). As a rule of thumb they determined the following relation between these two parameters: if an experiment with a base batch size b and a base learning rate η terminates in time t, then if the batch size is increased by a factor of k, i.e. $new\_batch\_size = b \cdot k$, then, in order to keep the execution time at t, $new\_learning\_rate = \eta \cdot k$ should be applied. In this case, in addition to the same execution times, the learning curves of the two learning processes are very similar as well.

3.3.2.2 Specific parameters

Besides the generic configuration parameters, the system has some specific configuration parameters as well. These are the following:

- **SVD:** From an arbitrary transformation matrix T an orthogonal T' can be obtained by applying the singular value decomposition (SVD) procedure. Smith et al. (2017) suggested applying SVD to the transformation matrices, in order to keep them orthogonal. Therefore, a parameter whether to apply the SVD procedure was added to the configuration parameters of the implemented system as well.

- **SVD mode:** For applying the SVD procedure three different modes are proposed, mode 0, 1, and 2. 0 means no SVD at all, 1 means doing an SVD regularly, i.e. on every n-th batch, and 2 means doing SVD only once at the very beginning, right after the first batch.

- **SVD frequency:** When applying SVD with mode 1, this option corresponds to the value n, i.e. the frequency of how often an SVD will be applied on the translation matrices.

- **Embedding limit:** The number of words occurring in an embedding varies from language to language. In order to be able to evaluate the system in the same way for different languages, only the first n lines of the given word embeddings are taken into account. This way look-up spaces will have the same size for every language.

3.3.2.3 Parameters for evaluation

At test time we used different metrics for evaluation:

- **Loss:** At training time the system optimizes for the cosine similarity using the entries of the training set. Testing the system in the canonical machine learning way means calculating this value for the entries in the test.

- **Precision:** More important metrics for the evaluation of the system are the precision scores. The system is capable of calculating any number of precision values that are set in the configuration file.

- **Number of small singular values of the translation matrices:** Many small singular values of a transformation matrix are indicators of mapping the data to a lower-dimensional space, which might lead to problems since it usually implies information loss. The meaning of "small" is rather relative, so a limit can be set for "small" values to monitor the singular values of the learned translation matrices.

3.4 Implementation issues

In this section the relevant features of the software architecture are discussed. The implemented code is available as an open source project[6]. The proposed method is implemented in Python 3[7]

[6] `https://github.com/Eszti/dipterv`
[7] `https://www.python.org/download/releases/3.0/`

using the following python packages: numpy[8], matplotlib[9], sklearn[10], gensim[11], and tensorflow[12].

3.4.1 Configuration files

During development it was important to implement the system in a flexible, and widely configurable way. The main idea behind the software architecture is that once the code base of the system is ready, it is expected to leave the code itself intact in the experimenting phase. Modifying only human-readable configuration files makes the whole experimental process much more transparent and traceable.

3.4.2 Embedding representation

Working with multilingual embeddings always leads to encoding issues. Therefore, Pyhton 3 improvements, which feature a `str` type that contains Unicode characters and uses UTF-8 for default encoding, are especially useful. In the implemented framework embeddings are represented as a floating point matrix with a shape of N*D, where N is the number of the words and D is the dimension of the embedding, along with an index2word list which assigns a word to each row of the matrix. For the common embedding properties a base class was created, and various sub-classes were derived for handling the different embedding formats.

[8] `http://www.numpy.org/`
[9] `https://matplotlib.org/`
[10] `http://scikit-learn.org/stable/`
[11] `https://pypi.python.org/pypi/gensim`
[12] `https://www.tensorflow.org/`

Chapter 4

Experiments

This chapter presents the experiments. First, parameter adjustment is discussed in detail, and then two different datasets, Dinu's English-Italian setting (Dinu et al., 2014) and an English-Italian PanLex subset, are used for testing the best setting. Finally, further experiments are presented, which combine these two datasets.

4.1 Baseline experimental setting

This section describes the baseline experimental setting that was used for parameter adjustment. For the baseline system the *fastText* embedding, described in Section 3.1.1, was used as a pre-trained embedding and the system was trained on Dinu's English-Italian data, described in Section 3.1.2. For parameter adjustment Dinu's training data was split into train and validation sets such that no overlap was present on the English side, i.e. no word appeared in both sets; this follows Dinu's procedure of constructing their original training and test sets. It should be noted that this does not apply for Italian words. For the word count and overlap statistics of Dinu's original training and test sets see Table 4.1 and for the same statistics of the newly produced training and validation sets see Table 4.2.

Number of English words	train	3442
Number of Italian words		4549
Number of English words	test	1500
Number of Italian words		1849
overlap English		0
overlap Italian		113

Table 4.1: Statistics of the original train and test split of Dinu's data

The system was adjusted on the previously described training and validation split for which the proposed procedure is discussed in Section 3.2. For the optimizer the tensorflow implementation[1] of the Adagrad algorithm (Duchi et al., 2011) was used.

[1] `https://www.tensorflow.org/api_docs/python/tf/train/AdagradOptimizer`

Number of English words	train	3098
Number of Italian words		4129
Number of English words	valid	344
Number of Italian words		499
overlap English		0
overlap Italian		80

Table 4.2: Statistics of the new train and validation split of Dinu's data

For evaluation the most frequent 200k words of the target space embedding were used as look-up space for calculating Precision @1, @5, and @10. In all cases both English-Italian and Italian-English precision scores were observed. In addition, the average cosine similarity value of the validation set was also checked. During training and validation as well the precision and similarity values were all calculated in the universal space. Gold dictionaries were constructed from the input data files themselves. Following Dinu, any word appearing in the dictionary was considered a valid translation. Various translations may come from synonyms or different male-female forms on the Italian side.

4.1.1 Adjusting basic parameters

The best learning rate and batch size setting was searched with the experimental setting described above. In all cases the system was trained for 10k epochs applying an initial SVD. Over the 10k training epochs evaluation was run on the validation set at every 1000th epoch. In the tables below the maximum precision values are shown which, in most of the cases, are not from the last epoch. It is essential to train the system long enough in order to see the learning curve reach its maximum value and break down.

4.1.1.1 Learning rate

For learning rate experiments the value of the batch size was fixed at 64 and various experiments were run with the following learning rates: 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, suggested by Andrew Ng in the Stanford Machine Learning Coursera course (Ng, 2016). Table 4.3 summarizes the experiments. Best results occur when the learning rate is 0.1.

LR	cos_sim	English - Italian Precision			Italian - English Precision		
		@1	@5	@10	@1	@5	@10
0.001	0.988743	0.1831	0.1831	0.3721	0.1667	0.2851	0.3494
0.003	0.995905	0.3401	0.5058	0.5669	0.3032	0.4799	0.5462
0.01	0.998957	0.4651	0.6366	0.6802	0.4036	0.6185	0.6586
0.03	0.999824	0.5262	0.7006	0.7645	0.4438	0.6506	0.6988
0.1	0.999994	**0.5407**	**0.7297**	**0.7645**	**0.4618**	**0.6546**	0.6948
0.3	1.000000	0.5407	0.7151	0.7645	0.4478	0.6526	**0.7028**
1	1.000000	0.4535	0.6483	0.6977	0.3554	0.5542	0.6265
3	1.000000	0.0698	0.1599	0.1890	0.0462	0.0462	0.1586

Table 4.3: Learning rate experiments. "LR" stands for "learning rate", and "cos_sim" denotes the average cosine similarity of the training set.

4.1.1.2 Batch size

Next, various experiments were run with 0.1 learning rate and the following batch sizes: 16, 32, 64, 128, 256. Table 4.4 summarizes the results. Best results occur when the batch size is 64.

BS	cos_sim	English - Italian Precision			Italian - English Precision		
		@1	@5	@10	@1	@5	@10
16	1.000000	0.5320	0.7209	0.7616	0.4418	0.6446	0.7008
32	1.000000	0.5203	0.7064	0.7558	0.4398	0.6446	0.6948
64	0.999994	**0.5465**	0.7209	**0.7878**	**0.4578**	**0.6627**	0.7068
128	0.999946	0.5407	**0.7267**	0.7645	0.4458	0.6586	**0.7129**
256	0.999949	0.5320	0.7093	0.7645	0.4398	0.6627	0.7088

Table 4.4: Batch size experiments. "BS" stands for "batch size", and "cos_sim" denotes the average cosine similarity of the training set.

4.1.1.3 Conclusions

Figure 4.1 shows the learning curve of the experiment with learning rate = 0.1 and batch size = 64. The red line shows the average cosine similarity on the training set and the green line on the validation set. Validation was done only 10 times over the 10k epochs, so compared to the training curve the validation curve is obviously very steep in the beginning.

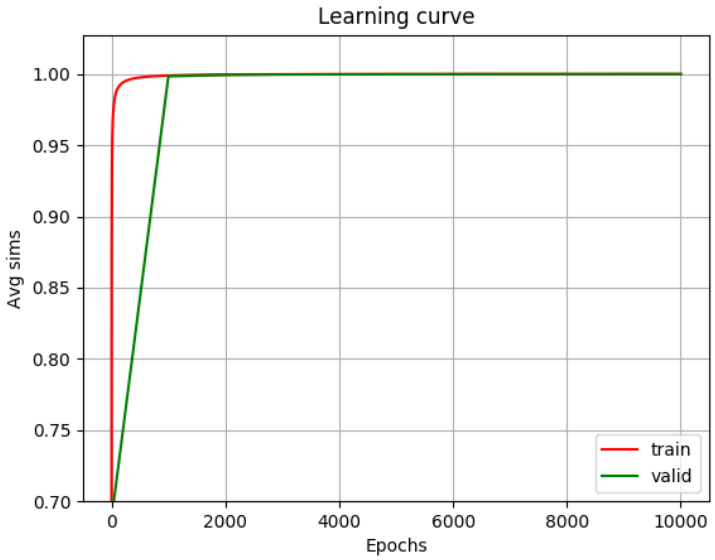

Figure 4.1: Learning curve of experimenting with learning rate = 0.1, batch size = 64.

Figure 4.2 shows the precision curves of English-Italian and Figure 4.3 the precision curves of Italian-English direction of the same experiment. As the average cosine similarity is getting higher, the precision is growing as well. After a certain point, however, the system reaches over-fitting and the precision curves start to decrease.

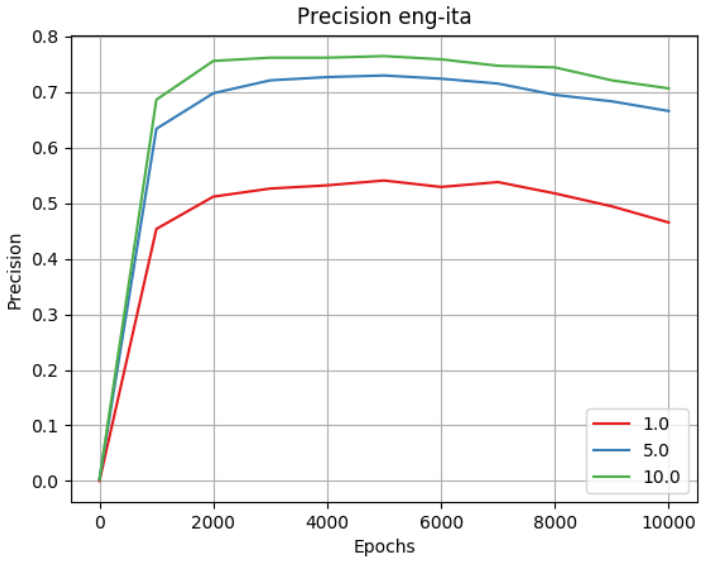

Figure 4.2: eng-ita precision curves, learning rate = 0.1, batch size = 64.

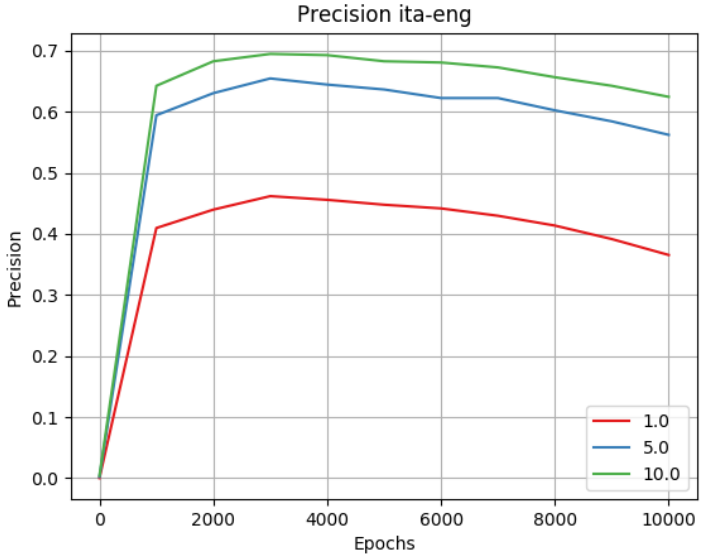

Figure 4.3: ita-eng precision curves, learning rate = 0.1, batch size = 64.

Once the translation matrices are learned through optimization on the cosine similarity, the system becomes apt for various multilingual applications as well such as word translation tasks. The results of the experiments above show that there is a clear correlation between the cosine similarity and the precision values.

4.1.2 Experimenting with SVD

Previous works, such as (Smith et al., 2017) or (Conneau et al., 2017), suggested restricting the transformation matrix to an orthogonal one. Based on these findings this system also features a configuration option of applying an SVD, explained in Section 3.3.2.2. Three different SVD modes were studied:

- **0**: Not using SVD at all.
- **1**: Using SVD after every n-th epoch.
- **2**: Using SVD only once, at the beginning.

In the following experiments 200 epochs were done, and evaluation was performed on every 10th epoch.

4.1.2.1 SVD mode = 0

This experiment was carried out without applying any SVD. Translation matrices were initialized with random numbers. Figure 4.4 shows that similarity values are monotone increasing, meaning that the system is learning. But the learning process is relatively slow since even after 200 epochs the similarity score is still quite low, bearing in mind that the optimal value is 1.0.

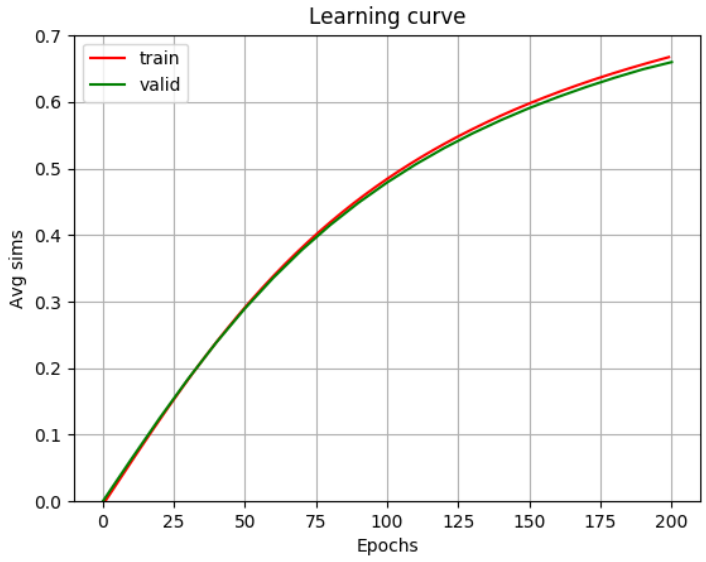

Figure 4.4: Learning curve of experimenting with svd_mode = 0.

4.1.2.2 SVD mode = 1

This experiment was carried out applying SVD several times over the whole learning process. SVD was made on every 50th epoch, i.e. 4 times altogether. Figure 4.5 shows how the learning curve

breaks down every time after applying an SVD on the translation matrices, and, also, how fast it is back once again to the previous high similarity values. Besides, this time the average cosine similarity score was higher even at the beginning than it was after 200 epochs with the previous setting, where no SVD was done. Applying SVD on the transformation matrices seems to accelerate the learning process significantly. The learning curve also shows that SVD-to-SVD fractions have exactly the same trajectory regardless of the number of previous epochs done.

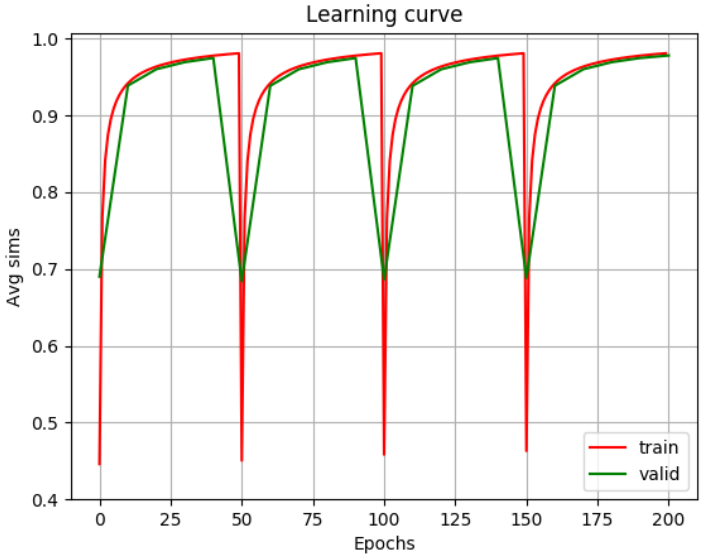

Figure 4.5: Learning curve of experimenting with svd_mode = 1.

4.1.2.3 SVD mode = 2

This experiment was carried out applying SVD only once, at the very beginning. This means, in simple terms, that instead of a random initial transformation matrix, the system tried to adjust an orthogonal one. Figure 4.6 shows that the learning curve is monotone increasing, and owing to the initial SVD it gets fairly high right at the beginning.

4.1.2.4 Dimensionality loss in universal space

The typical pattern of over-fitting in machine learning applications is still increasing similarity scores in parallel with decreasing precision values. One possible explanation is the reduction of dimensionality in the translated space. This also implies information loss that can lead to decreased precision values. One indicator of this problem is the high number of small singular values of the translation matrix. In order to monitor this phenomenon the number of singular values smaller than 0.1 was checked. Table 4.5 shows that as the average similarity is monotone increasing both on the training and the validation sets, the number of small singular values of the translation matrices is increasing as well.

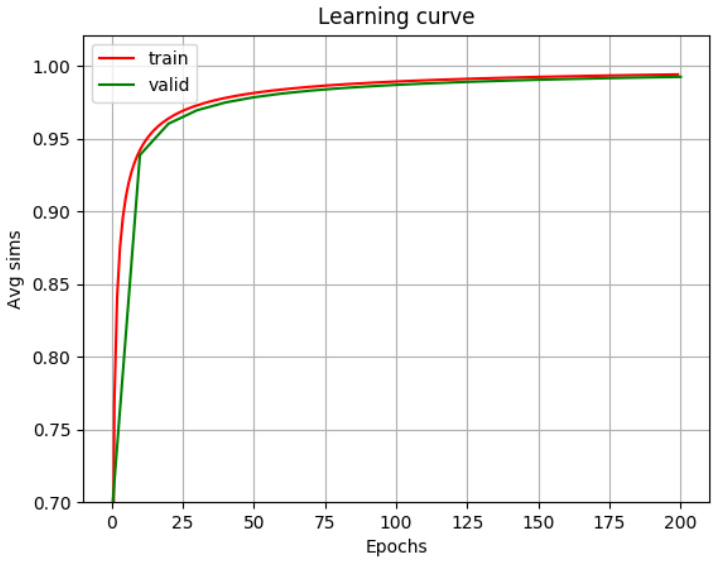

Figure 4.6: Learning curve of experimenting with svd_mode = 2.

Epoch	No.of sing.values <0.1 (eng)	No. of sing.values <0.1 (ita)	train	valid
0	0	0	0.447719	0.687022
1000	24	27	0.998958	0.998392
2000	76	68	0.999627	0.999369
3000	120	113	0.999823	0.999684
4000	157	153	0.999905	0.999824
5000	190	188	0.999946	0.999896
6000	215	215	0.999967	0.999936
7000	237	237	0.999979	0.999959
8000	255	257	0.999987	0.999974
9000	258	270	0.999991	0.999983
10000	278	280	0.999994	0.999988

Table 4.5: Monitoring dimensionality loss in the universal space. Learning rate = 0.1, batch size = 64, SVD mode = 2. The second and the third columns are showing the number of singular values smaller than 0.1 in the English and Italian translation matrices, respectively.

4.1.2.5 Conclusion

As a result of the parameter adjustment process the best learning rate and batch size values have been found, which are 0.1 and 64, respectively. Experiments with SVD have shown that results are the best if SVD is applied only once, at the beginning.

4.2 Testing the baseline system on Dinu's experimental setting

The system was tested on Dinu's original English-Italian data described in Section 3.1.2 using the best parameter setting. First the *fastText* embedding described in Section 3.1.1 was used, and after

that the same embedding that Dinu et al. applied.

4.2.1 Using the *fastText* embedding

Dinu's data originally has 5000 word pairs in the training and 1869 word pairs in the test set. However, in this experiment the system was trained on only 4999 and tested on only 1640 word pairs due to lack of embedding coverage as shown in Table 4.6.

	train	test
eng words	3442	1500
not found	0	97
ita words	4548	1849
not found	1	156
all word pairs	5000	1869
found	4999	1640

Table 4.6: fastText embedding coverage of Dinu's data

Figure 4.7 shows the eng-ita precision scores and Figure 4.8 the ita-eng ones. Unsurprisingly, the English-Italian direction performs better given that some English words in the test set can translate to either the male or female form of the corresponding Italian word. The obtained results are significantly worse than state-of-the-art results on this benchmark data but they are comparable with or even better than some of the previous models discussed in Section 2.4. For comparison see Table 4.7 and Table 4.8.

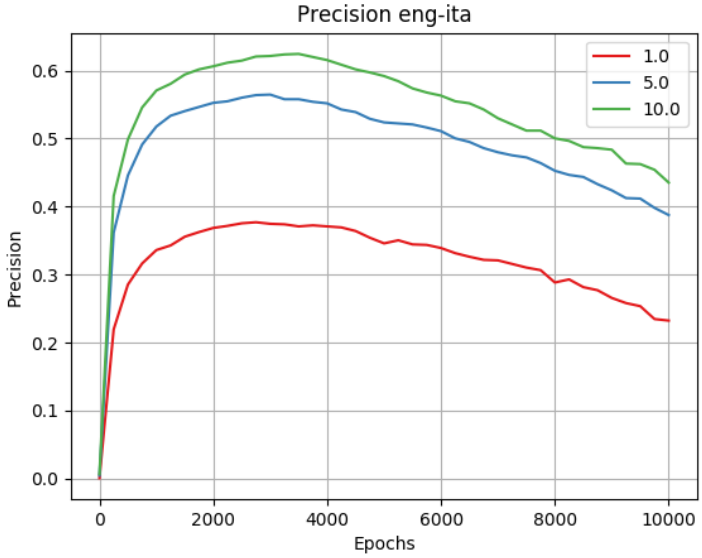

Figure 4.7: eng-ita precision curves of the proposed method on Dinu's data using fastText embedding.

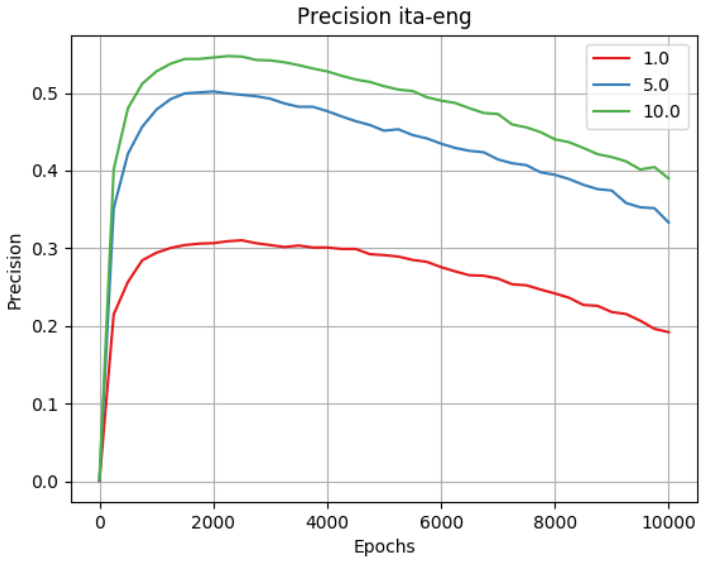

Figure 4.8: ita-eng precision curves of the proposed method on Dinu's data using fastText embedding.

4.2.2 Dinu's word vectors

Next, the system was run with the same embedding that was used by Dinu et al. (2014) in their experiments. These word vectors were trained with *word2vec* and then the 200k most common words in both the English and Italian corpora were extracted. The English word vectors were trained on the WackyPedia/ukWaC and BNC corpora, while the Italian word vectors were trained on the WackyPedia/itWaC corpus. Figure 4.9 shows the eng-ita precision scores and Figure 4.10 the ita-eng ones. Once again, the English-Italian direction performs better than Italian-English direction, as expected. Results with Dinu's word vectors are worse than the previous results using the *fastText* embedding. See Table 4.7 and Table 4.8.

Eng-Ita	@1	@5	@10
Mikolov et al.	0.338	0.483	0.539
Faruqui et al.	0.361	0.527	0.581
Dinu et al.	0.385	0.564	0.639
Smith et al. (2017)	0.431	0.607	0.651
Conneau et al. (2017) - WaCky	0.451	0.607	0.651
Conneau et al. (2017) - fastText	**0.662**	**0.804**	**0.834**
Proposed method - fastText	0.377	0.565	0.625
Proposed method - WaCky	0.220	0.333	0.373

Table 4.7: Comparing English-Italian results on Dinu's data.

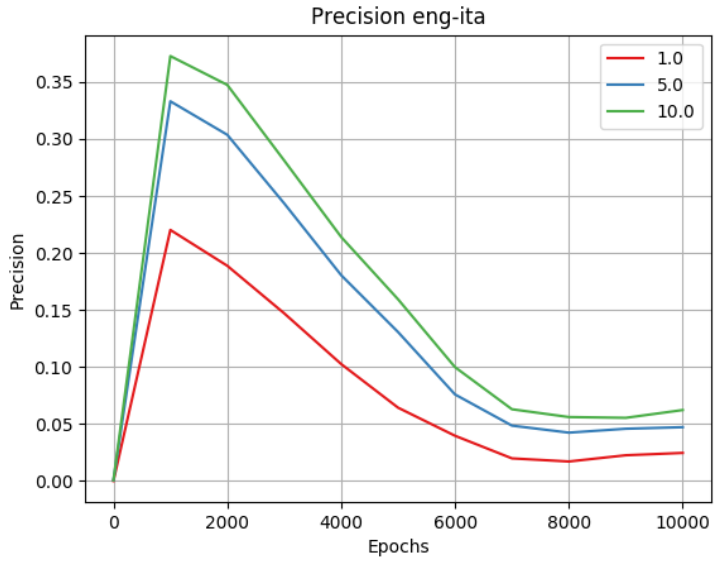

Figure 4.9: eng-ita precision curves of the proposed method on Dinu's data using WaCky embedding.

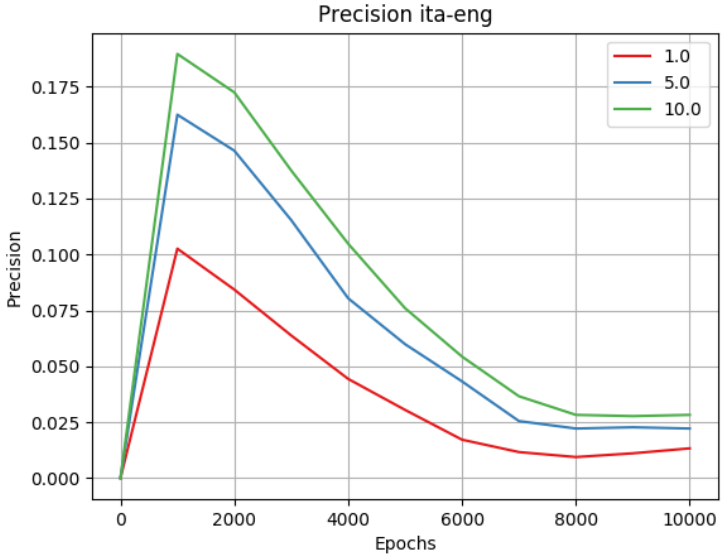

Figure 4.10: ita-eng precision curves of the proposed method on Dinu's data using WaCky embedding.

4.3 English-Italian Panlex experiments

This section describes the experiments carried out on English-Italian word pairs extracted from the Panlex dataset. First, experimental data creation is discussed, followed by the results.

Ita-Eng	@1	@5	@10
Mikolov et al.	0.249	0.410	0.474
Faruqui et al.	0.310	0.499	0.570
Dinu et al.	0.246	0.454	0.541
Smith et al. (2017)	0.380	0.585	0.636
Conneau et al. (2017) - WaCky	0.383	0.578	0.628
Conneau et al. (2017) - fastText	**0.587**	**0.765**	**0.809**
Proposed method - fastText	0.310	0.502	0.547
Proposed method - WaCky	0.103	0.163	0.190

Table 4.8: Comparing Italian-English results on Dinu's data.

4.3.1 Dataset creation

This section describes the dataset creation in detail. First, a brief analysis of the English-Italian data is presented, and then the exact creation process is explained.

4.3.1.1 Analysis of the English-Italian data

In the PanLex database to each translation pair a confidence value is assigned as described in Section 3.1.3. One translation pair is often found with different confident values, therefore during the data extraction process translation duplicates were dropped retaining only the occurrence with the highest value.

After extracting the English-Italian translation pairs from the PanLex database an analysis, shown in Table 4.9, was performed. The second column shows how many entries are contained in the PanLex database with a certain confidence value, while in the third column the number of those entries are listed for which an actual *fastText* word embedding was found. Finally, the last column adds up second column numbers above a certain value. All together there are 187,601 English-Italian word translation pairs in the PanLex database for which a *fastText* word embedding can be found.

score	# wp == score	# wp == score, filtered	# wp >= score
9	1389	66	66
8	4265	514	580
7	163701	69043	69623
6	1085	67	69690
5	79419	26478	96168
4	6045	2836	99004
3	272276	47477	146481
2	126837	36182	182663
1	6893	4938	187601

Table 4.9: Summary of English-Italian PanLex data inspection

4.3.1.2 Creation of 5k English-Italian PanLex data

The procedure applied for extracting a proper data from the PanLex database for training multilingual embedding models roughly follows the steps Dinu et al. (2014) took. After extracting the

raw translation pairs form the PanLex database, a filtered version of entries was formed by dropping translations with a confidence value below 7 and those for which no word vector was found in the *fastText* embedding. This results in an English-Italian word translation set containing 69,623 entries, as can be seen in the corresponding cell of Table 4.9.

For the test set 1,500 English words were taken and split into 5 frequency bins, 300 randomly assigned to each bin. The bins were defined the same way Dinu et al. (2014) defined them, i.e. in terms of rank in the frequency-sorted lexicon: [1-5K], [5K-20K], [20K-50K], [50K-100K], and [100K-200K]. Conneau et al. (2017) published their word vectors sorted by their frequency in descending order, and this order was used as the source of English word frequency data. In the PanLex database it is a common issue that one English word has sometimes as many as 10 different Italian translations. Therefore, in order to avoid having an undesirably huge test set with many Italian synonyms only those English words were selected, for which in the corresponding bin only one Italian translation was present. This way the obtained test set contains exactly 1,500 word pairs, which are made up of 1,500 different English words and their Italian translations.

For the training set, the 69,623 entries were first sorted by the English frequency, then the top 5k entries were extracted and, the same way as by Dinu et al., care was taken to avoid any overlap with test elements on the English side. Then, the top 5k entries were selected in three different ways:

1. Simply the first 5k entries were taken.
2. The first 5k different English words were taken with the most frequent Italian translation.
3. Only those English words were taken for which only one Italian translation was present.

Table 4.10 and Table 4.11 show experiments run with all the three different datasets. Results are the best in the third case, thus this approach was applied later on as well. It is important to note that in the first case the English word of the 5000th translation pair is only the 845th most frequent English word, meaning that there is only 845 different English words in the training set and that, on average, there is 5-6 different Italian translations to each of them. In the second case, where every English word is kept but only with the most frequent Italian translation, this number is 9007. In the last case, however, the 5000th entry is made up of the 39426th most frequent English and the 31543th most frequent Italian words. Still, this last training set provides the best results.

Precision	@1	@5	@10
first 5k entry	0.0093	0.0253	0.0367
first 5k English words with retaining one translation	0.1120	0.2073	0.2427
first 5k English words with one translation	**0.1960**	**0.3087**	**0.3440**

Table 4.10: English-Italian precision values with the different training sets

Precision	@1	@5	@10
first 5k entry	0.0000	0.0007	0.0007
first 5k English words with retaining one translation	0.1114	0.2052	0.2440
first 5k English words with one translation	**0.1838**	**0.3059**	**0.3443**

Table 4.11: Italian-English precision values with the different training sets

4.3.2 Experiments with different training set sizes

The same way the best performing 5k training set was created, experiments were made with different training set sizes. Table 4.12 summarizes the results. The 3k dataset proved to be the best on the English-Italian translation, but on the Italian-English it is only slightly better, than the 5k dataset.

	eng-ita			ita-eng		
Precision	@1	@5	@10	@1	@5	@10
1k	0.1500	0.2847	0.3340	0.1391	0.2761	0.3256
3k	**0.2127**	**0.3473**	**0.3933**	**0.2232**	**0.3650**	**0.4152**
5k	0.1980	0.3193	0.3620	0.2212	0.3555	0.4030
10k	0.1613	0.2807	0.3227	0.1879	0.3012	0.3372

Table 4.12: Experiments with different training set sizes

4.3.3 Comparing PanLex data with Dinu's data

In the next step, some experiments were made to determine which data is more apt for learning linear mappings between embeddings. In order to compare all the experiments objectively subsets of the original test sets were created. These subsets do not contain any English word present either in the Dinu training set or in the PanLex training set. Table 4.13 summarizes the number of word pairs in the old and the new test sets. It should be noted that by this reduction principally the most common English words are affected, and therefore worse scores are expected compared to the previous train-on-Dinu-test-on-Dinu top results, described in Section 4.2.

test set	No. of word pairs in old	No. of word pairs in new
Dinu	1869	1455
PanLex	1500	1242

Table 4.13: Word reduction of the new test sets

Results on Dinu's test set are shown in Table 4.14 and on the PanLex data in Table 4.15. Results show that training on the PanLex data cannot beat the system trained on Dinu's data, which performs better both on Dinu's and on the PanLex test sets. Not even combining the two training sets succeeds in achieving significantly better results, although on the PanLex test set it does improve the scores in the Italian-English direction.

	eng-ita			ita-eng		
Precision	@1	@5	@10	@1	@5	@10
train:PanLex - test:old	0.3770	0.5647	0.6245	0.3103	0.5018	0.5474
train:PanLex - test:new	**0.3560**	**0.5407**	**0.5978**	**0.2917**	**0.4792**	**0.5215**
train:Dinu - test:new	0.1360	0.2309	0.2594	0.1361	0.2556	0.2965
train:Dinu+PanLex - test:new	0.2930	0.4349	0.4861	0.2910	0.4556	0.5090

Table 4.14: Comparing Dinu's and PanLex data on Dinu's test set

	eng-ita			ita-eng		
Precision	@1	@5	@10	@1	@5	@10
train:PanLex - test:old	0.1960	0.3087	0.3440	0.1838	0.3059	0.3443
train:PanLex - test:new	0.1812	0.2858	0.3196	0.1668	0.2835	0.3213
train:Dinu - test:new	**0.2295**	**0.4171**	**0.4839**	0.2227	0.3763	0.4199
train:Dinu+PanLex - test:new	0.2295	0.3712	0.4275	**0.2498**	**0.4026**	**0.4495**

Table 4.15: Comparing Dinu's and PanLex data on the PanLex test set

4.4 Continuing the baseline system with PanLex data

Another experiment was conducted to continue the baseline system trained on Dinu's data with the PanLex data. In other words, it is the same as initializing the translation matrices of the PanLex training process with previously learned ones. The baseline system reaches its best performance between 2000 and 4000 epochs, depending on which precision value is regarded as Figure 4.7 and Figure 4.8 show. Therefore, continuation was done with three different settings: the translation matrices were initialized with the one obtained from the baseline system after 2000, 3000, and 4000 epochs. Table 4.16 summarizes the results. On the English-Italian task there is no improvement at all, while on the Italian-English task with the best setting slightly better scores are achieved on precision @1 and @10 values. Figure 4.11 and Figure 4.12 show the precision curves of the 2000 epoch continuation experiment.

	eng-ita			ita-eng		
Precision	@1	@5	@10	@1	@5	@10
original	**0.3770**	**0.5647**	**0.6245**	0.3103	**0.5018**	0.5474
cont from 2000	0.3426	0.5256	0.5802	**0.3229**	0.4882	**0.5535**
cont from 3000	0.3535	0.5416	0.5970	0.3229	0.4840	0.5465
cont from 4000	0.3510	0.5273	0.5911	0.3118	0.4701	0.5243

Table 4.16: Continuing the baseline system with the PanLex data.

4.5 Multilingual Panlex experiments

This section describes the experiments carried out on a multilingual dataset extracted from Panlex. In these experiments instead of only two languages the system was trained on three different languages at the same time: English, Italian, and Spanish.

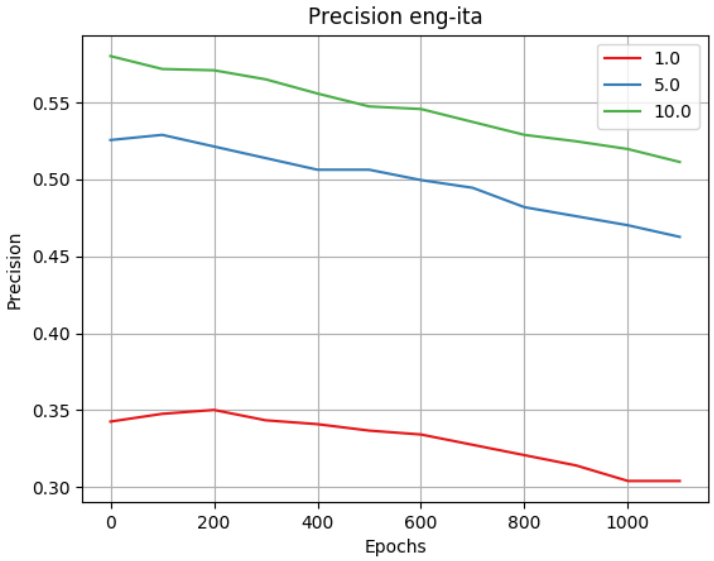

Figure 4.11: eng-ita precision curves of the 2000 epoch continuation of the baseline system.

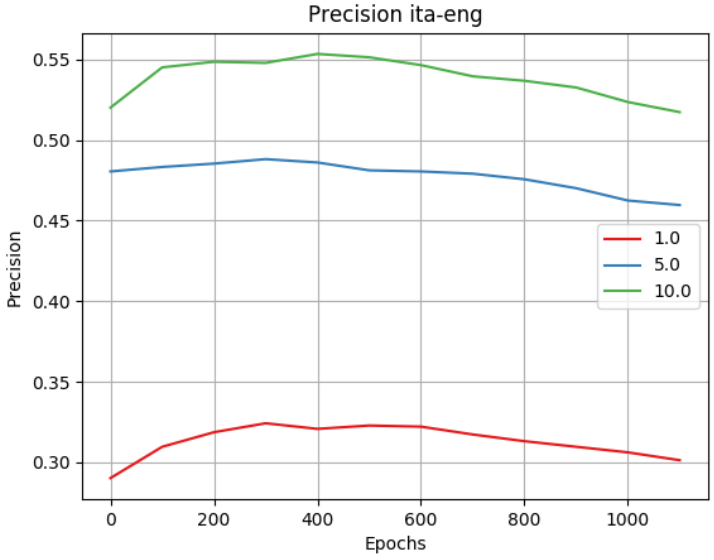

Figure 4.12: ita-eng precision curves of the 2000 epoch continuation of the baseline system.

4.5.1 Dataset creation

When training the system on n number of languages it requires $\binom{n}{2}$ number of bilingual data, i.e. a gold dictionary is needed in all different language pairs. The advantage of using the PanLex database is that it contains translations between many languages. For the {eng, ita, spa} language set pairwise word translation dictionaries were extracted with the same method as described in Section 4.3.1.2. The frequency sort on the Italian-Spanish dataset was done according to the frequency order of Italian words. This information was extracted from the Italian *fastText* embedding word order, in

the same way as it was done with the English word frequencies. Naturally the English-Spanish dataset was sorted by the English frequency order. For these experiments a new English-Italian dataset was created, which was extracted the same way but may distribute the words differently.

4.5.2 Experiment results

During training the system learns three different translation matrices, one for English-universal, one for Italian-universal, and one for Spanish-universal space mapping. For example, in order to learn the English-universal translation matrix, both the English-Italian and the English-Spanish dictionaries are used according to Equation 3.2. Batches are homogeneous, but two following batches are always different in terms of the language origins of the contained data. That is, first an English-Italian batch is fed to the system, then an English-Spanish one, after that an Italian-Spanish one, and so on.

First, bilingual models were trained in order to compare them later with the multilingual system. The results of the bilingual models are summarized in Table 4.17. Results are best on the Italian-Spanish task. Next, the system was trained using all the three languages at the same time. During the training process the model was evaluated on the bilingual test datasets of which the results are shown in Table 4.18. The obtained results show that no advantage was achieved by extending the number of languages, since the multilingual model performs worse than any of the pairwise bilingual models.

	eng-ita			ita-eng		
Precision	@1	@5	@10	@1	@5	@10
eng-ita	0.2080	0.3280	0.3687	0.2082	0.3386	0.3904
eng-spa	0.2840	0.4320	0.4800	0.2883	0.4331	0.4836
spa-ita	0.3920	0.5340	0.5813	0.3655	0.5291	0.5750

Table 4.17: Results of bilingual models trained pairwise on the three different languages.

	eng-ita			ita-eng		
Precision	@1	@5	@10	@1	@5	@10
eng-ita	0.1573	0.2667	0.3127	0.1638	0.2942	0.3386
eng-spa	0.1947	0.2973	0.3447	0.2350	0.3538	0.4064
spa-ita	0.2520	0.3640	0.4160	0.2568	0.3723	0.4162

Table 4.18: Bilingual results of the multilingual model trained using three different languages at the same time.

Chapter 5

Conclusion and future work

5.1 Summarizing the contributions of this thesis

This thesis work proposes a novel method for finding linear mappings between word embeddings in different languages. First, the field of Natural Language Processing (NLP) was introduced by describing its main motivation and by giving some examples of real life applications that are taking advantage of NLP technologies. Then, the topic of word embeddings was discussed in more detail, concentrating principally on most recent advances in multilingual word vector representations. Different word vector mapping approaches, multilingual models and their evolution over the last 5 years were summarized, and their results on Dinu's English-Italian benchmark data (Dinu et al., 2014) were reported. After that, multilingual resources were introduced, which were utilized in this thesis work for multilingual embedding learning. A new method, which aims to learn translations between word embeddings in different languages by mapping them all into a universal space, was proposed. A widely configurable experimentation framework was implemented which was first used for running parameter adjustment experiments, and then for testing the best setting on different datasets.

Parameter adjustment experiments showed that a learning rate of 0.1 and a batch size of 64 are the best configuration options. Another interesting finding is that the system learned much faster when an initial singular value decomposition (SVD) was applied on the translation matrices. Results obtained with these settings on Dinu's data showed that the proposed model did learn from the data. The obtained precision scores, though, are far from current state-of-the-art results on this benchmark data, they are comparable with results of previous attempts. The proposed model performed much better using the *fastText* embeddings (Conneau et al., 2017), than using Dinu's WaCky embeddings (Dinu et al., 2014).

Thereafter, an English-Italian dataset was extracted from the PanLex database, from which training and test datasets were constructed roughly following the same steps that Dinu et al. (2014) did. The system was trained on this dataset as well. This way two different translation matrices were obtained, one which was trained on Dinu's data, and one which was trained on the PanLex

data. The performance was tested on both Dinu's and PanLex test sets, and in both cases the former matrices, the ones trained on Dinu's data, were the ones reaching higher scores. On the PanLex data experiments with different training set sizes were executed, out of which the 3k training set gave the best results. After that, the training of the matrices obtained with Dinu's data was continued with the PanLex dataset. A slight improvement was experienced on the Italian-English scores, but the English-Italian ones only got worse.

Finally, the system was trained on three different languages, on English, Italian, and Spanish, at the same time. The obtained pairwise precision values are worse than the results obtained when system was trained in bilingual mode. However, these results are still promising considering that a completely new approach was implemented, and they showed that the system definitely learned from a data which is available for a wide range of languages.

5.2 Future work

This thesis work proposed a novel method for obtaining translation matrices between word embeddings. As a proof of concept a framework was developed which enabled basic parameter adjustments and flexible configuration for initial experimentation.

The approach is quite promising but in order to reach state-of-the-art performance the system has to deal with some mathematical issues, for example dimension reduction in the universal space. Further collaboration with mathematicians is encouraged, since they might be able to propose modifications to help the current model overcome its problems.

Further experimentation in multilingual mode with an extended number of languages could also provide meaningful outputs. By involving expert linguistic knowledge various sets of languages could be constructed using either only very close languages, or, on the contrary, using very distant languages. Thanks to the PanLex database, bilingual dictionaries can easily be extracted, which can, then, be directly used for multilingual experiments.

Bibliography

J. Acs, K. Pajkossy, and A. Kornai. Building basic vocabulary across 40 languages. In *Proceedings of the Sixth Workshop on Building and Using Comparable Corpora*, pages 52–58, 2013.

E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, and A. Soroa. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27. Association for Computational Linguistics, 2009.

W. Ammar, G. Mulcaire, Y. Tsvetkov, G. Lample, C. Dyer, and N. A. Smith. Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01925*, 2016.

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*, pages 2289–2294, 2016.

M. Bansal, K. Gimpel, and K. Livescu. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 809–815, 2014.

M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43 (3):209–226, 2009.

R. Beckwith, C. Fellbaum, D. Gross, and G. A. Miller. Wordnet: A lexical database organized on psycholinguistic principles. *Lexical acquisition: Exploiting on-line resources to build a lexicon*, pages 211–232, 1991.

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

G. Dinu, A. Lazaridou, and M. Baroni. Improving zero-shot learning by mitigating the hubness problem. *arXiv preprint arXiv:1412.6568*, 2014.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

T. Economist. Finding a voice. *The Economist*, 2017. `www.economist.com/technology-quarterly/2017-05-01/language`, accessed: 17.05.2018.

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*, pages 462–471, 2014.

C. Fellbaum. *WordNet*. Wiley Online Library, 1998.

L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM, 2001.

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

P. Gärdenfors. *Conceptual spaces: The geometry of thought*. MIT press, 2004.

P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

F. Hill, R. Reichart, and A. Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695, 2015.

K. Huang, M. Gardner, E. Papalexakis, C. Faloutsos, N. Sidiropoulos, T. Mitchell, P. P. Talukdar, and X. Fu. Translation invariant word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1084–1088, 2015.

D. Jurafsky and J. H. Martin. *Speech and language processing*. Pearson London:, 2017.

D. Kamholz, J. Pool, and S. M. Colowick. Panlex: Building a resource for panlingual lexical translation. In *LREC*, pages 3145–3150, 2014.

A. Kornai. The algebra of lexical semantics. In *the Mathematics of Language*, pages 174–199. Springer, 2010.

A. Kornai. Eliminating ditransitives. In *Formal Grammar*, pages 243–261. Springer, 2012.

G. Lakoff. *Women, fire, and dangerous things*. University of Chicago press, 2008.

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)*, volume 1, pages 270–280, 2015.

T. Luong, H. Pham, and C. D. Manning. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159, 2015.

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

T. Mikolov, Q. V. Le, and I. Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013b.

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

G. A. Miller and W. G. Charles. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28, 1991.

T. Naseem, B. Snyder, J. Eisenstein, and R. Barzilay. Multilingual part-of-speech tagging: Two unsupervised approaches. *Journal of Artificial Intelligence Research*, 2009.

A. Ng. Machine learning. `https://www.coursera.org/learn/machine-learning`, 2016.

D. Picca, A. M. Gliozzo, and S. Campora. Bridging languages by supersense entity tagging. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 136–142. Association for Computational Linguistics, 2009.

D. Radev, W. Fan, H. Qi, H. Wu, and A. Grewal. Probabilistic question answering on the web. *Journal of the Association for Information Science and Technology*, 56(6):571–583, 2005.

K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th international conference on World wide web*, pages 337–346. ACM, 2011.

G. Recski, E. Iklódi, K. Pajkossy, and A. Kornai. Measuring semantic similarity of words using concept networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 193–200, 2016.

E. Rosch and C. B. Mervis. Family resemblances: Studies in the internal structure of categories. *Cognitive psychology*, 7(4):573–605, 1975.

H. Rubenstein and J. B. Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633, 1965.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

S. L. Smith, D. H. Turban, S. Hamblin, and N. Y. Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*, 2017.

Stoyanov and Ayan. Under the hood: Multilingual embeddings. *facebook code*, 2018. `https://code.facebook.com/posts/550719898617409/under-the-hood-multilingual-embeddings/`, accessed: 17.05.2018.

M. Swadesh. Lexico-statistic dating of prehistoric ethnic contacts: with special reference to north american indians and eskimos. *Proceedings of the American philosophical society*, 96(4):452–463, 1952.

D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1555–1565, 2014.

J. Tiedemann. Parallel data, tools and interfaces in opus. In *LREC*, volume 2012, pages 2214–2218, 2012.

J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

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*, pages 1006–1011, 2015.

H. Youn, L. Sutton, E. Smith, C. Moore, J. F. Wilkins, I. Maddieson, W. Croft, and T. Bhattacharya. On the universal structure of human lexical semantics. *Proceedings of the National Academy of Sciences*, 113(7):1766–1771, 2016.

K. Zhao, H. Hassan, and M. Auli. Learning translation models from monolingual continuous representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1527–1536, 2015.