Universidad del País Vasco Euskal Herriko Unibertsitatea

A Common Semantic Space for Monolingual and Cross-Lingual Meta-Embeddings

Author: Iker García Ferrero

Advisors: German Rigau i Claramunt

hap/lap

Hizkuntzaren Azterketa eta Prozesamendua
Language Analysis and Processing

Final Thesis

September 2019

Departments: Computer Systems and Languages, Computational Architectures and Technologies, Computational Science and Artificial Intelligence, Basque Language and Communication, Communications Engineer.

Abstract

This master's thesis presents a new technique for creating monolingual and cross-lingual meta-embeddings. Our method integrates multiple word embeddings created from complementary techniques, textual sources, knowledge bases and languages. Existing word vectors are projected to a common semantic space using linear transformations and averaging. With our method the resulting meta-embeddings maintain the dimensionality of the original embeddings without losing information while dealing with the out-of-vocabulary (OOV) problem. Furthermore, empirical evaluation demonstrates the effectiveness of our technique with respect to previous work on various intrinsic and extrinsic multilingual evaluations.

Acknowledgments

We thank Rodrigo Agerri for the help given to evaluate our meta-embeddings in the Sequence Labelling task as well as for the help provided for the writing of the research paper derived from this work. We thank Mikel Artexe for his help to set up VecMap and use it in our experiments. We thank Eneko Agirre and Mikel Artetxe for their suggestions during the early stages of this work on how to improve it and possible research lines.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Context

This master's thesis is framed in the area of Natural Language Processing (NLP). Natural language processing is a field of research in computer science, artificial intelligence and linguistics, which studies how to model the human language by computational means. NLP final purpose is making computers able to understand, interpret and manipulate human language. Among the main challenges of natural language processing are speech recognition, natural language comprehension and language generation. Among these great challenges are tasks such as automatic translation or the extraction of information from texts.

1.2 Introduction

Humans are really good understanding language and conversations. When someone says a word like "bank" we are able to understand if the word is used in the context of a financial institute or a river bank. At the same time, we are able to understand that works such as "beautiful" and "pretty" have roughly the same meaning. We are also able to conclude that if something is "good" is not "bad" since it cannot be both at the same time. We do all these things thanks to logical, linguistic, commonsense reasoning and understanding.

To build machines able to understand natural language like we do, the first task is to teach them the meaning of words. Encoding the meaning of words in a computer is the most basic and at the same time more challenging task in NLP. Fortunately, during the past few years, significant progress have been made in this field.

In classic NLP systems based on rules and statistics word are represented as discrete and atomic units. If we use as example [1] the sentences "I like icecream" and "I love icecream", if we use a discrete variables to represent each word such one-hot vectors and we consider that our vocabulary contains 4 words, we will represent "'I" as [0,0,0,1], "like" as [0,0,1,0], "love" as [0,1,0,0] and "icecream" as [1,0,0,0], using this encoding, "like" and "love" will be two different words without any relation between them, they will be as different as "I" and "icecream". This type of representation does not encode any information about the relations that could exits between different words, all that can encode is the presence or absence of words in a sentence. Representing words as unique discrete variables does not provide sufficient information to be able to develop systems that can successfully perform complex natural language processing tasks.

Taxonomies or semantic networks are other classic approaches trying to represent the meaning of words. WordNet (Miller, 1995) or ConceptNet (Speer et al., 2017) are examples of such lexical knowledge bases. In the case of WordNet it stores hyperonymic relations such as "A cat is an animal" and sets of synonyms. However, in WordNet if we search for the synonyms of the word "expert"', we will find the words "adept", "good", "practiced", "proficient", "skillful". But the sentences "I am an *expert* in NLP" and "I am *skillful* in NLP"

[1] Example inspired by "Introduction to Word Embeddings and Word2Vec" by Dhruvil Karani

do not share the exact same meaning. Another problem is that we cannot measure that "good" can be more similar to "skilful" than to "expert" [2]. These representations usually suffer from another problem, they need to be developed and maintained by humans, which require a huge amount of labour, makes very difficult to maintain the knowledge base updated, and since the knowledge base is created by humans subjectivity becomes also a problem.

Modern word semantic representation methods are based on the Distributional Hypothesis (Harris, 1954). This hypothesis states that words that occur in the same contexts, tend to have similar meanings. We can know the meaning of the word bank by the words that appear in the same context, for example, if we refer to the financial entity sense, we will find words such as "money", "goverment", "debs", "regulation", "crisis", etc. The underlying idea that "a word is characterized by the company it keeps" was popularized by Firth (1957) and is one of the most successful ideas in NLP.

Based on the distributional hypothesis, many word representation methods have been proposed. One of the first methods is the so-called "bag of words". Words are represented a large vector of numbers, in with each dimension is a measure of association between words and a particular type of information, such as the document in which it appears, the words next to which it appears, etc. This usually generates a very large matrix, where almost all positions are zeros (sparse matrix), meaning that the dot product between almost all the vectors will equal zero, providing us no useful information. This problem is solved by applying dimensionality reduction algorithms (i.e Single Value Decomposition) that transforms the large sparse matrix in a dense matrix with fewer dimensions (Deerwester et al., 1990).

The breakthrough in word representations methods come with neural networks (Rumelhart et al., 1988; Bengio et al., 2003; Mnih and Hinton, 2009; Collobert et al., 2011). These methods become popular after the word done by Tomas Mikolov and Dean (2013), who developed word2vec, a toolkit for learning neural vector representations of words. These type of models are called *predictive models*. Predictive models are based on the idea that if we can predict in what context a word appears, then it means that we understand the meaning of the word in its context. Predictive models attempt to directly predict a word from its neighbours in terms of small, dense vectors that are learned during training. Words are represented in vector spaces where semantically similar words will be found close to each other. Figure 1 shows an example of a word embeddings in a 2-D vector space, we can see that words that appear in similar contexts are close to each other. This type of representations are known as "Word Embeddings".

Using word embeddings as input, Neural Networks and Deep Learning methods have become the most successful approach in current natural language processing. These models have been successfully applied to many complex tasks such as Question Answering (Wang et al., 2017; Santoro et al., 2017; Kamath et al., 2019; Yoon et al., 2019), Machine Translation (Bahdanau et al., 2014; Gehring et al., 2017; Vaswani et al., 2017), Word Sense

[2]Example borrowed from: "Natural Language Processing with Deep Learning" by Christopher Manning and Richard Socher

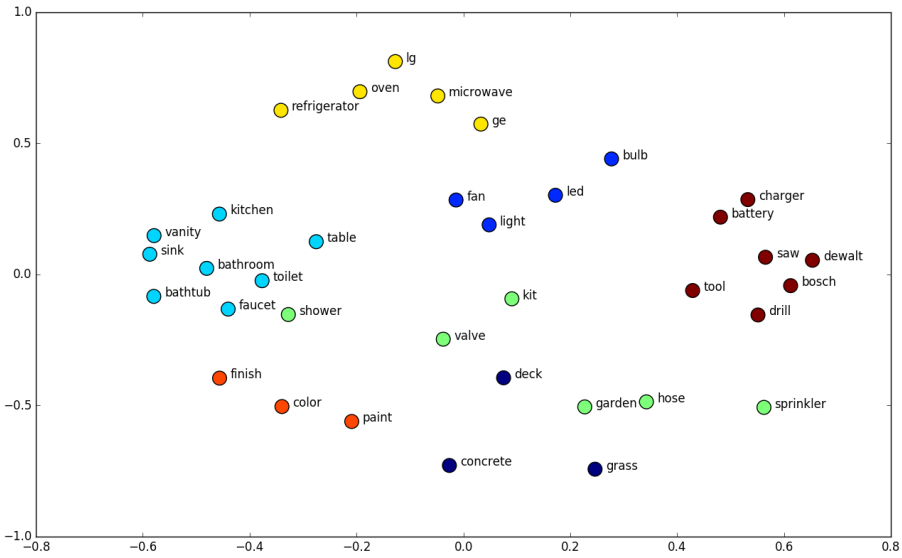

Figure 1: Example of Word Embeddings represented in a 2-D vector space (source: `https://blog.kaggle.com/2016/05/18/home-depot-product-search-relevance-winners-interview-1st-place-alex-andreas-nurlan/`)

disambiguation (Papandrea et al., 2017; Vial et al., 2019) or Semantic Role Labelling (Zhou and Xu, 2015; Kozhevnikov and Titov, 2013; Marcheggiani and Titov, 2017) among many others (Otter et al., 2018). While these models provide superior possibilities that classic NLP approaches, almost all the models are supervised, this is, they require annotated data for training. For many languages and domains there is not enough annotated data available or even there is not annotated data at all, this leads to inferior performance compared to English. This is why recently the development of cross-lingual embedding has received great attention. Cross-lingual embeddings are vector spaces that contain words for two or more languages, words that have a similar meaning in both languages will be found close to each other. Cross-lingual embedding provides a mechanism for transfer learning, that is, train a model for a resource-rich language and transfer it to a low-resource language (Upadhyay et al., 2016; Artetxe et al., 2018b; Goikoetxea et al., 2018).

On the other hand, there exists another line of research that follows the hypothesis that different knowledge sources contain complementary semantic information (Goikoetxea et al., 2016). Thus, several authors have tried to enhance the quality of distributional word representations by incorporating information for knowledge bases (Halawi et al., 2012b; Faruqui et al., 2015; Bollegala et al., 2016; Speer et al., 2017), languages (Vulić and Korhonen, 2016; Artetxe et al., 2017, 2018c) or both (Speer and Lowry-Duda, 2017; Goikoetxea et al., 2018). Another way to improve word representations is to obtain an ensemble of distinct word embeddings each trained using different methods and resources, and possibly

containing complementary knowledge. This ensembles are know as "Meta-Embeddings" (Yin and Schütze, 2016). While meta-embeddings show promising results, learning a single meta-embedding from multiple sources remains a challenging task. The algorithms, resources and languages used for learning the source word embeddings often differ significantly and it is now obvious how to reconcile their differences. Previous approaches based on concatenation (Goikoetxea et al., 2016) produce word meta-embeddings that suffer from a much larger dimensionality, alternatives such as averaging (Coates and Bollegala, 2018) or dimensionality reduction (Raunak, 2017) suffer or from loss of information.

The objective of our research is to explore the combination of word embeddings which have been independently learned applying different techniques to different textual sources, knowledge bases and languages by projecting all word embeddings to a common semantic space using linear transformations and averaging.

1.3 Objectives

This Master's Thesis is the continuation of the work done for in my final degree work "Estudio de Word Embeddings y métodos de generación de Meta Embeddings" (Study of Word Embeddings and methods of generating Meta-embeddings) (García-Ferrero, 2018), where we studied the performance of different pre-trained word embeddings, normalization methods and meta embedding generation methods in the word similarity task. The main approach of this work was the development of a meta embedding generation method using linear transformations Artetxe et al. (2018b) and averaging. The main objective of this work is to extend the proposed method to the multi-lingual domain with two intentions. The first one being able to improve the quality of the generated meta-embeddings by ensembling representations in different languages. The second one is to use it as a transfer learning mechanism, where pre-trained embeddings trained in a rich-resource language can improve the quality of the pre-trained embedding from a low-resource language. In the original research, we evaluated our embeddings in the word similarity task. Now, we will extend the results evaluating the generated mono-lingual and cross-lingual meta-embeddings in more challenging tasks: Semantic Textual Similarity (STS), Part-of-Speech (POS) and Named Entity Recognition (NER).

The objectives of the research also include the development of the necessary software for generating and evaluating word-embeddings. This software will be available under a copyleft license in GitHub [3].

1.4 Structure of the Document

This document is divided in 7 sections bing the first one this brief introduction. The rest of the sections are structured as follows:

- Section 2 presents and discuss the current state-of-the-art in the field of research.

[3] `https://github.com/ikergarcia1996/`

- Section 3 presents and explains in-depth our proposed method for meta-embedding generation.
- Section 4 presents an evaluation of different pre-trained word embeddings our meta-embeddings and alternative meta-embedding generation method in the word similarity task. We include monolingual and cross-lingual evaluations in English and Spanish.
- Section 5 present a multilingual (Spanish and English) evaluation of different pre-trained word embeddings and our meta-embeddings in the Semantic Text Similarity task. In this section, we will also explore the transfer-learning capabilities of our meta-embeddings.
- Section 6 presents a multilingual (Spanish and English) evaluation of different pre-trained word embeddings and our meta-embeddings in the Sequence labelling task.
- Section 7 concludes, presents the main contributions of the work and discusses future work.

2 State of the art

This chapter will focus on the background of the project. We will divide it into four sections. Section 2.1 explains in-depth word embeddings, their influence and applications, the most relevant embedding generation methods and word embedding evaluation methods. Section 2.2 describes the current state-of-the-art in meta-embedding generation methods. In our project we made use of cross-lingual embeddings and word mapping methods, section 2.3 explains them. Finally section 2.4 describes the current state-of-the-art in contextual embeddings methods.

2.1 Word Embeddings

Word embedding approach to distributional semantics by representing words as real number vectors. This representation has useful grouping properties. Semantically similar words will appear closer in the vector space. For example, we expect words such as "cat" and "dog" to be close, since both are animals, mammals and pets, but "cat" and "train" not to be close since there is no a strong relationship between them. Therefore, words are represented as vectors of real values, where each value captures a dimension of the word meaning. The numerical value en each dimension captures the closeness of the association of the word to that meaning. This causes semantically similar words to have similar vectors.

Vector space models have been used in distributional semantics since the 1990s. Since then, different models have been developed to estimate continuous representations of words, one of the most relevant examples in Latent Semantic Analysis (LSA). The term word embedding was originally conceived by Bengio et al. (2003), who trained a vector space model using neuronal probabilistic model. However Collobert and Weston (2008) where probably the first ones to demonstrate the power of word embeddings, they probed that word embeddings are a highly effective tool in different types of tasks such as Semantic Role Labelling (SRL), Named Entity Recognition (NER), Part-of-speech tagging (POS), chunking and language modelling. They also proposed a neural network architecture in which is the base for many modern approaches.

However, word embeddings were popularized and became an indispensable resource for NLP largely thanks to the work done by Tomas Mikolov and Dean (2013) who released "'Word2Vec", a tool to train and use word embeddings. A year later Pennington et al. (2014) released GloVe, another toolkit for word embedding generation. From this moment on, word embeddings have become one of the main building blocks in NLP.

2.1.1 Influence and Applications

Word embeddings are able to successfully capture lexical and semantic information about words in terms of vectors than can be used as input for any machine learning algorithm. Therefore, their use have spread rapidly and they are now a fundamental piece in the architecture of all type of models that perform NLP tasks. The list of possible applications

for word embeddings is huge. We will describe just some of the applications where word embeddings have been very successful and have had a big impact.

- Automatic machine translation has been a topic of great interest since the very beginning of computer science. For example, in 1954, the IBM 701 computer successfully translated 49 sentences on the topic of chemistry from Russian into English. However, classic machine translation approaches such as rule-based systems or statistics-based systems were not able to produce wide coverage good quality machine translations for all languages. In conjunction with neural network architectures, word embeddings have massively improved the quality of the machine translations (Edunov et al., 2018). Modern neural machine translations systems such as Google Translate, DeepL or Modela are able to provide high quality machine translations for many language pairs.

- Text classification accuracy has been a field that has also massively improved thanks to word embeddings and neural network architectures (Kowsari et al., 2019). This has resulted in the development of customer service, spam detection, document classification systems or information retrieval systems with a a very high accuracy for many domains.

- Question Answering systems is one of the faster-growing topics in NLP. Question Answering has benefited enormously from modern deep learning techniques as well as from the improvement made in the text classification and information retrieval fields. This has resulted in very famous systems such as "Siri", "Alexa" or "Google assistant" that are used by millions of people around the globe.

- Automatic Text Summarisation is a field which has been gaining popularity in recent years as a way to avoid the overload of information on the internet. Many modern systems that made use of word embeddings and neural networks architectures are able to achieve very good results in this task. Pointer-Generator Networks by Reddy et al. (2019) is an example of these models.

These are just some examples of the influence and applications of word embeddings among many others. Word embeddings have become the base of almost any NLP task.

2.1.2 Word Embedding Generation Methods

Since word embeddings began to popularize, many different methods to generate word embeddings have emerged. In this section, we will present the most relevant methods to generate words embeddings among the ones that we will use later in our empirical study. The methods for generating word embeddings are usually accompanied by pre-calculated word vectors. We used these pre-calculated word vectors in our experiments, so we will also specify them in this section. We also include a short description of alternative meta-embeddings that we used in our experiments.

- **Word2Vec** (W2V) ((Tomas Mikolov and Dean, 2013)) implements two neural models: CBOW and Skip-gram. In the first, given the context of the target word, it tries to predict it. In the second, given the word it tries to predict the context. The word embedding of a word corresponds to the internal layers of the neural network. We use the embeddings from Google News (100 billion words).

- **GLOVE** (GV) (Pennington et al. (2014)) unlike Word2Vec, is a counting-based model. GloVe generates a large matrix where the information of the concurrency between words and contexts is stored. That is, for each word we count how many times that word appears in some context. The training goal of this matrix is to learn vectors so that the scalar product between the words is equal to the logarithm of the probability of co-occurrence between the words. As the number of contexts is very high, a factorisation of the matrix is performed to obtain one of smaller dimensions. In this way a vector that represents each word is obtained. In this report we include the results of the Common Crawl pre-trained vectors (pre-trained on a corpus of 600 billion words). As the authors recommend, we have applied $l2$ normalization to its variables to improve performance.

- **FastText** (FT) (Mikolov et al., 2018) is an extension of word2vec that treats each word as the sum of its composing characters (ngrams). For the example, we will calculate the vector for the word "apple" as the sum of the vectors for the ngrams that form it '<ap, app, appl, apple, apple>, ppl, pple, pple>, ple, ple>, le>". In this way it is expected to obtain better representations for " rare " words, which have very few appearances in the corpus of texts, and thus be able to generate vectors for words that are not found in the vocabulary of the word embeddings. For English, we used Common Crawl (pre-trained on a corpus of 600 billion words). For Spanish, the model is trained on the Spanish Billion Word Corpus Cardellino (2016) with around 1.4 billion words.

- **LEXVEC** (LV) (Alexandre Salle and Villavicencio, 2016) is a model that seeks to obtain better results thanks to the combination of GloVe and Word2Vec. The most recent version uses context vectors. Context vectors seek to improve the performance of word embeddings in analogy tasks. Usually, word embedding generation models only take into account words within the context of the objective word. For example in the sentence: "'The big **dog** barked loudly" if the objective word is "'dog", the context will be "'(the, big, barked, loudly)". Context vectors also take into account the relative position of each word in the context relative to the objective word, for example, the context in the previous example will be encoded as "(The^{-2}, big^{-1}, barked^{1}, loudly^{2})." It also implements the ngram approach from FastText. We have used the vectors trained in the common crawl corpus using around 58 billion words.

- **RWSGwn** (UKB) (Goikoetxea et al., 2015): is a model that combines random walks over WordNet (Fellbaum, 1998) with the skip-gram model. The random walks over WordNet generate sentences of the type: "amphora wine nebuchadnezzar bear retain

long". The phrase begins with amphora, *a container that is usually filled with wine*. Wine is the second word. Nebuchadnezzar is a particular bottle size. And the final words are related to storage. In the original paper, the generated corpus is processed by the skip-gram model to generate word embeddings. We have used the vectors trained using WordNet3.0 plus gloss relations. Apparently using GloVe instead of skip-gram improves the performance of the final embeddings, so by default we used GloVe instead.

- **Attract Repel** (AR) Mrkšić et al. (2017) is an algorithm for improving the semantic knowledge encoded in word embeddings created from textual corpora by injecting synonymy and antonym constraints extracted from monolingual and cross-lingual lexical resources. Attract-Repel is also able to use semantic relations extracted from BabelNet (Navigli and Ponzetto, 2012) to inject constraints between words in different languages to map vector spaces of multiple languages into a single vector space. We used the English vocabulary from the published four-lingual (English, German, Italian, Russian) vector space.

- **Paragram** (P) (Wieting et al., 2015) are pre-trained word vectors learned using word paraphrase pairs from PPDB (Ganitkevitch et al., 2013) using a modification of the skip-gram objective function. The hyperparameters were tuned using the wordsim-353 dataset. The word embeddings of the default model are initialized with GloVe word vectors.

- **Numberbatch** (N) (Speer et al., 2017): they claim to be the best pre-trained word embeddings available. They combine knowledge encoded in ConcepNet, word2vec, GloVe and OpenSubtitles 2016 using concatenation, dimensionality reduction (truncated SVD) and a variation on retrofitting (Faruqui et al., 2015). We used Numberbatch version 17.06 in the experiments.

- **JOINTChyb** (J) (Goikoetxea et al., 2018) combines Random Walks over multilingual WordNets and bilingual corpora as input for a modified skip-gram model that forces equivalent terms in different languages to come closer during training. We used the English-Spanish bilingual embeddings publicly available.

During our experiments we also tested other word embeddings for creating our meta-embeddings such as PDC/HDC (Sun et al., 2015) or context2vec (Melamud et al., 2016) without showing significant improvements over the ones shown in this section.

2.1.3 Word Embeddings Evaluation Methods

In the section 2.1.2 we have described many word embeddings generation methods, but there are many more. Since word embeddings are the base for almost every NLP model, a wide range of different approaches have been proposed for generating them. Evaluation methods to determine the quality of the different word representations are necessary.

These evaluation methods can be divided into two main groups: Intrinsic and Extrinsic evaluations.

Intrinsic evaluations compare word embeddings against human judgements on words relations. Manually generated sets of words are used to obtain human assessments. Then, these assessments are compared with the ones obtained from the word embeddings. There are two main alternatives to collect these assessments. The first one is using a limited set of experts in the field to make judgements on a given data. The other one uses crowd-sourcing platforms such as "Mechanical Turk", where assessments can be collected from an unlimited number of participants. Crowd-sourcing platforms allow the generation of huge datasets thanks to the collaboration of a large number of persons. However, using experts in the field can result in a better quality evaluation dataset. Many intrinsic evaluation methods exist Bakarov (2018). We will describe the most common ones: word similarity, word analogy, concept categorization, synonym detection and outlier word detection.

- **Word similarity**: This method is based on the idea that distances between words could be evaluated through the human heuristic judgements on the actual semantic distances between these words. For example, the distance between "house" and "flat" defined in the continuous interval [0,1] could be 0.7 since both words are quite similar although not exactly the same thing. The distances between pairs are also calculated in the word embedding space. The more similar these results are to the human judgements, the better are the word embeddings. This method will be explained in-depth together with the datasets available in section 4 where we will perform an evaluation of word- and meta-embeddings using this evaluation method.

- **Word analogy**. Word embeddings also seems to capture semantic regularities such as word analogies as shown by (Mikolov et al., 2013b). A word analogy holds between two word pairs: $a{:}a^* :: b{:}b^*$ (a is to a^* as b is to b^*). For example, having the puzzle *Tokyo* is to *Japan* as *Paris* is to b^* the relation $a{:}a^*$ seems to be capital:country. Thus, the word b^* that we are looking for is *France*. Some of the most well-kown dataset for word analogy are WordRep (Gao et al., 2014), BATS (Gladkova et al., 2016), Google Analogy (Tomas Mikolov and Dean, 2013), SemEval-2012 (Jurgens et al., 2012), MSR (Mikolov et al., 2013b), SAT (Turney et al., 2003) and JAIR (Turney, 2008).

- **Concept Categorization** or word clustering. Given a set of words, the task is to split them into subsets of words belonging to different categories. For example two cluster seem to appear in *dog*, *cat*, *hammer*, *screwdriver*. One with the words *dog* and *cat* that corresponds to animals and another one with *hammer* and *screwdriver* that correspond to tools. The number of clusters is given by the dataset. The most well-known datasets for the Concept Categorization task are BM (Baroni et al., 2010), AP (Almuhareb, 2006), BLESS (Baroni and Lenci, 2011) and ESSLLI-2007 (Baroni et al., 2008).

- **Synonymy Detection** is very close to the word similarity task. However, instead of calculating a value for the similarity for a pair of words, the task is to determine

which word from the list is more similar to a given one. For example, the most similar word to *cat* given the list of words including *dog*, *car*, *church* and *moon* is *dog* since both *dog* and *cat* are animals. The most well-known datasets for this tasks are RDWP (Jarmasz and Szpakowicz, 2004), TOEFL (Landauer and Dumais, 1997) and ESL (Turney, 2001).

- **Outlier word detection** is a similar task to concept categorization. However, the goal is not to divide a set of words in a certain amount of clusters, but to identify a semantically anomalous word in an already formed cluster. For example given the cluster including *elephant*, *car*, *cat*, *dog*, *giraffe* and *lion* the outlier word is seems to be *car* since all the words in the cluster are animals expect this one. The most well-known datasets for this tasks are 8-8-8 Dataset (Camacho-Collados and Navigli, 2016) and WordSim-500 (Blair et al., 2016).

Extrinsic evaluations compare word embeddings based on the ability to be used as vectors of characteristics on supervised automatic learning algorithms used in various NLP tasks. The performance of the supervised method on a particular dataset is taken as a measure of the quality of the word embedding. By definition, any NLP task where we have a model that takes as input word embeddings and a dataset available for evaluating the model is an extrinsic evaluation method. Therefore, the list of possible extrinsic evaluation methods is very large since it combines every existing NLP task and system using word embeddings. In our research we carried out some experiments on Semantic Textual Similarity (Section 5) and Sequence labelling (Section 6) as evaluation methods for different word- and meta-embeddings. Tasks, datasets and systems used are in-depth described in their corresponding sections.

2.2 Word Meta-Embeddings

In this section, we revise previous research with meta-embeddings. Meta-embeddings have been a field of study since the moment word embeddings became popular. Hill et al. (2014) showed that word embeddings trained from monolingual or bilingual corpora capture different nearest neighbours, proving that they encode different knowledge. Yin and Schütze (2016) where the first ones that proved the possibilities of meta-embeddings. By meta-embedding five different pre-trained word embeddings, they show that we can overcome the out-of-vocabulary problem, while improving the accuracy of cross-domain part-of-speech (POS) tagging.

Although word embeddings are mainly constructed by exploiting information from text corpora only as we shown in the section 2.1.2, some researchers tried to combine them with the knowledge encoded in lexical resources such as WordNet (Halawi et al., 2012b; Bollegala et al., 2016; Goikoetxea et al., 2016), PPDB (Faruqui et al., 2015) or ConceptNet (Speer et al., 2017). Their work show improvements in performance when combining text corpora and lexical resources, which seems to prove the hypotheses that they encode different knowledge that can be combined to create new word representations with improved

quality. Moreover, Goikoetxea et al. (2016) show that simply concatenating word embeddings derived from text and WordNet outperform alternative methods such as retrofitting (Faruqui et al., 2015) at the cost of increasing the dimensionality of the meta-embeddings. Trying to overcome this issue Coates and Bollegala (2018) found that averaging is in some cases better than concatenation, with the additional benefit of reduced dimensionality. The most popular approach to address the dimensionality problem is to apply dimensionality reduction algorithms such as SVD (Yin and Schütze, 2016), PCA (Ghannay et al., 2016) or DRA (Raunak, 2017). In this line of work Numberbatch (Speer et al., 2017) claims to be the best meta-embedding model generated so far. The meta-embedding combines the knowledge encoded in ConcepNet, word2vec, GloVe and OpenSubtitles corpus using concatenation, dimensionality reduction and a variation on retrofitting.

Other alternative approaches have also been proposed. Bollegala et al. (2017) proposed an unsupervised locally linear method for learning meta-embeddings from a given set of pretrained source embeddings. Bollegala and Bao (2018) propose three types of autoencoders to learn meta-embeddings. Autoencoders are a type of artificial neural network used to learn efficient data encoding in an unsupervised manner.

2.3 Cross-lingual embeddings

In previous sections, we have discussed how monolingual word embeddings have proven to be extremely useful across a wide range of NLP applications. However, these monolingual word representations are only suitable for mono-lingual tasks. Cross-lingual word embeddings is a research line that has attracted a lot of attention in recent times. Cross-lingual word embeddings represent words from different languages in the same vector space in such a way that words with similar meaning in different languages may appear close to each other. Section 2 shows an English-Italian example. In the left image, we can see how the two monolingual word embeddings are unaligned. This representation does not provide any useful knowledge about the relations between words in both languages. However, in the image to the right both monolingual embeddings have been projected into a joint cross-lingual embedding space. Words with similar meaning in both languages tend to appear close to each other. This can provide us with a lot of information about the relationship between words in both languages. For example, we can use these cross-lingual word representations to measure which is the most similar word in English for a given Italian word.

Cross-lingual embeddings are useful for two main reasons. The first one is that they enable us to compare word meaning across languages, which is key for bilingual lexicon induction, machine translation, or cross-lingual information retrieval among others. Second, although many NLP models based on deep learning techniques have achieved very good results for many tasks, they require huge amounts of annotated data for training. For many languages and domains, there is not enough annotated data available or even there is not annotated data at all. This leads to inferior performance compared to English. Cross-lingual embeddings enable model transfer learning between languages, that is, transfer the knowledge learned in a resource-rich language to a low-resource language.

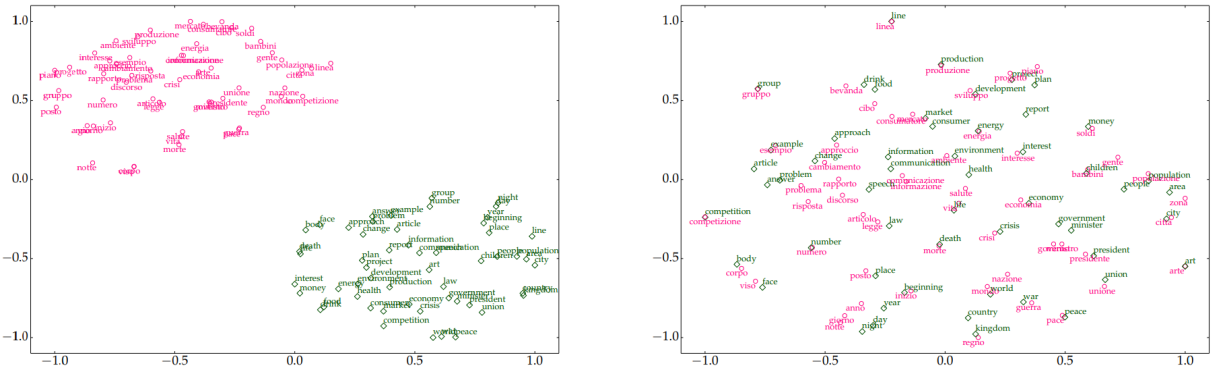

Figure 2: Unaligned monolingual word embeddings (left) and word embeddings projected into a joint cross-lingual embeddings space (right). Source: "A survey of cross-lingual word embedding models" by Ruder et al. (2017)

Transfer-learning allows using a resource-rich language to improve the accuracy of a task in a low-recourse language where not enough data is available to successfully train the required NLP models. Cross-lingual transfer learning (Lin et al., 2019) has been proved to improve the performance in several NLP task, including machine translation (Neubig and Hu, 2018; Nguyen and Chiang, 2017; Johnson et al., 2016), parsing (Ponti et al., 2018; Ahmad et al., 2018), part of-speech or morphological tagging (Plank and Agic, 2018; Malaviya et al., 2018), named entity recognition (Xie et al., 2018; Agerri et al., 2018) and entity linking (Rijhwani et al., 2018) among others.

Many models for learning cross-lingual embeddings have been proposed in recent years (Ruder et al., 2017). One of these methods is using parallel corpora in different languages (Luong et al., 2015) or some sort of bilingual signal (Mogadala and Rettinger, 2016). However, these resources may not available for many language pairs. An alternative approach seeks to remove the necessity of using bilingual data. This approach independently trains the embeddings for each language using monolingual corpora and then learns a linear transformation to map one embedding into the other based on a bilingual dictionary. In our research, we take advantage of these mapping methods to generate our meta-embeddings. More precisely, we apply VecMap by Artetxe et al. (2016) for creating meta-embeddings.

2.3.1 Mapping Word Embedding

In this section, we discuss different approaches applied for mapping word embeddings learned independently from different languages.

Word embeddings mapping methods seek to project independently learned mono-lingual word representations to a common cross-lingual vector space. Mikolov et al. (2013a) learns a linear transformation minimizing the sum of squared Euclidean distances of the word vectors. This approach was later improved and expanded by other authors Zhang et al. (2016); Lazaridou et al. (2015); Xing et al. (2015). Instead of using a linear transformation

from the source language to the target language, other authors use canonical correlation analysis (CCA) to map both languages to a common space (Faruqui and Dyer, 2014; Lu et al., 2015). Artetxe et al. (2016) propose a general framework that clarifies the relation between the previous work as variants of the same core optimization problem. They propose a new method called VecMap that surpasses them all. However, these methods made use of large bilingual dictionaries, that may not be available for many language pairs. Recent research has focused on using much smaller dictionaries (Artetxe et al., 2017) or even no dictionary at all (Conneau et al., 2017; Artetxe et al., 2018b).

Word Embeddings mapping methods have been successfully applied to cross-lingual meta embedding generation. Doval et al. (2018) tries to improve the cross-lingual word embeddings generated by VecMap (Artetxe et al., 2018c) and MUSE (Conneau et al., 2017) by averaging the source embeddings after its mapping to a common vector space. Our method extends this approach by combining into a single meta-embedding any number of monolingual and cross-lingual source embeddings created from complementary techniques, textual sources, knowledge bases and languages. Existing word vectors are projected to a common semantic space using linear transformations and averaging. We also extend the method to generate word representations that are missing in some of the source embeddings. This allows us to improve the cross-lingual mapping and the quality of the final meta-embeddings. As a consequence, the performance improves for both intrinsic and extrinsic mono-lingual and cross-lingual evaluations. In addition, we can use cross-lingual transfer learning to exploit pre-trained source embeddings from a resource-rich language in order to improve the word representations for under-resourced languages.

2.4 Contextual Embeddings

Word embeddings have been proven useful and even essential in many NLP task. However, all the embeddings generation methods such as the ones described in section 2.1.2 generate a unique and a static representation for each word. For example, we only have a representation for the word *bank*. However, *bank* has a very different meaning in the following sentences [4]:

1. A **bank** of clouds was building to the northeast.
2. Her **bank** account was rarely over two hundred.
3. She sat on the river **bank** across from a series of wide, large steps leading up a hill to the park where the Arch stood, framed against a black sky.
4. Deidre's gaze was caught by the **bank** of windows lining one side of the penthouse.
5. Mr. Catlin at the **bank** says he's as honest as they come.
6. ATMs replaced human **bank** tellers, so they are called "Automated Teller Machines.

[4]Examples borrowed from `https://sentence.yourdictionary.com/bank`

In each sentence the word *bank* have a different meaning. However, using static word embeddings, all instances of *bank* will share the same word representation. Alternatively, we can also use sense embeddings (Iacobacci et al., 2015). Furthermore, even if they refer to the same sense (e.g. financial institution), two appearances of the word *bank* can still not have exactly the same meaning. *Contextualized word embeddings* obtain different word representations depending on the context. The sequence tagger of Li and McCallum (2005) is one of the first works employing contextualized word representations. The model infers context-sensitive latent variables for each word using clustering and integrates them, as additional features, to a sequence tagger. However, contextual word representations become a research field after the introduction of word embeddings and their proven efficacy as input for neural networks. Context2vec by Melamud et al. (2016) is one of the first and one of the most relevant proposals for contextualized word representations. The model represents the context of a target word by extracting the output embedding of a multi-layer perceptron built on top of a bi-directional LSTM language model. Context2vex focuses on using contextualized word representations as input for downstream tasks with the objective of improving their accuracy. Context2vec is the base for many of the subsequent works.

The work that popularized and proved the power of contextualized word representations is ELMo by Peters et al. (2018). ELMO uses use language models to obtain embeddings for individual words while taking the entire sentence or paragraph into account. These representations are later used as input in a wide variety of NLP downstream tasks. Concretely, ELMo uses a pre-trained, multi-layer, bi-directional, LSTM-based language model and extract the hidden state of each layer for the input sequence of words. LSTM-based language models is a model that can predict how likely a certain sequence of words is a real piece of text. ELMo trains this LSTM-based language model by trying to predict the next word in a given sequence of words. ELMo is trained using a massive amount of unlabelled training data. ELMo suppose a significant step forward in NLP achieving state-of-the-art results in a wide variety of tasks including Question Answering, Natural Language Inference, Semantic Role Labelling, Coreference Resolution, Named entity recognition and Sentiment Analysis.

LSTMs in which ELMo is based, have been successfully applied in NLP for many years. Recently, instead of LSTMs Vaswani et al. (2017) presented a new architecture called transformer for machine translation. Transformer seems to deal better than LSTMs with long-term dependencies. BERT by Devlin et al. (2019) is a language model similar to ELMo that replaces the LSTM architecture by a transformer. BERT supposed another step forward in NLP improving the state-of-the-art results previously achieved by ELMo.

After ELMo and BERT proved the efficacy of contextualized word representations many other models emerged. All of them introducing improvements in the transformer architecture and mainly increasing the computational resources and training data used. The most well-known models are GPT2 by Radford et al. (2019). They generated a great controversy in the NLP community after they decided not to release their biggest pre-trained model claiming that it was so good that it could be dangerous (its text generation capabilities could be used to generate a huge amount of artificial content on the internet). XLNet (Yang et al., 2019), an improvement of the BERT architecture, achieves even better results.

Figure 3: Contextualized word embeddings give words a different emebdding based on the meaning they carry in the context of the sentence. Source: "The Illustrated BERT, ELMo, and co." by Jay Alammar. `http://jalammar.github.io/illustrated-bert/`

Contextualized word representations have achieved new state-of-the-art results for every NLP task while taking into account the context of the words is necessary. However, training these models require a huge amount of data and computational resources that may be not easily accessible. Although these models are usually provided as pre-trained models to be incorporated into downstream tasks, they are not available for every language and domain.

We tested our meta-embeddings in the Semantic Text Similarity task (Section 5) and in the Sequence labelling task (Section 6) in order to compare our new meta-embeddings with respect existing word- and meta-embeddings, not for obtaining new state-of-the art performances on these tasks.

3 A Common Semantic Space for Meta-Embeddings

In this chapter we describe our method for word meta embedding generation (Section 3.1) and its advantages against alternative methods (Section 3.2).

In the work done for my final degree work *Estudio de Word Embeddings y métodos de generación de Meta Embeddings* García-Ferrero (2018), we found that current methods for creating meta embedding face multiple problems. Concatenation is able to combine word embedding from different sources without losing information. However, the resulted meta embedding has as many dimension as the sum of the dimension of the source embeddings. The resulting meta-embeddings have a very high dimensionality and are not suitable for its use in a real NLP tasks. Other methods, such as dimensionality reduction or averaging are able to overcome this problem at the cost of losing information and reducing its performance.

We proposed a new algorithm to generate meta-embeddings which consists of three steps: (i) aligning the vector spaces using VecMap (Artetxe et al., 2018b), (ii) creating new representations for OOV words and, (iii) averaging the resulting word embeddings.

In this thesis, the original algorithm has been generalized to create *multi-lingual meta-embeddings* by combining mono-lingual and multi-lingual source embeddings in different languages. Additionally, the original algorithm has also been completely rewritten and optimized. First, it takes advantage of modern GPUs or CPUs with multiple cores. Second, the new algorithm offers much lower memory usage. It does not require to load all the word embeddings at the same time. For instance, the original algorithm last 2 days[5] combining the English pre-trained FastText embedding and the English pre-trained UKB embedding (the ones described in the section 2.1.2). The new code can combine these two embeddings in less than 30 minutes using the same system. This allowed us to make many more experiments on the same amount of time.

Now, our method can combine any number of word embeddings generated with any technique, source or language as long as there is some common vocabulary between them. The resulting meta-embedding vocabulary will be the union of the vocabularies of the word embeddings used.

3.1 Method explanation

Our approach to create meta-embeddings consists of three steps: (i) aligning the vector spaces using VecMap, (ii) creating new representations for OOV words and, (iii) averaging the resulting word embeddings.

3.1.1 VecMap

Artetxe et al. (2016) presents a method that learns bilingual mappings between word embeddings. VecMap first applies a normalisation to the word embeddings and then

[5]On a system with 2x Intel Xeon E5-2640 v4 (40 threads in total) and a GTX Titan X and 64GB of RAM

an orthogonal transformation. Orthogonality allows monolingual invariance during the mapping, preserving vector dot products between words. Monolingual invariance makes Vecmap a suitable method for generating meta-embeddings since it is not causing loss of information after the mapping. Recent versions of VecMap introduce multiple steps to improve bilingual mappings (Artetxe et al., 2018a). However, for meta-embedding generation, the best approach seems to be just applying, as a preprocessing step, length normalisation and mean centering to the original word embeddings (Step 0 as described in Artetxe et al. (2016)), and then just applying an orthogonal mapping (Step 2). In this way, we are able to sequentially project to a common space any number of word embeddings[6].

Although VecMap has the ability to induce bilingual dictionaries (Artetxe et al., 2018c) for meta-embedding generation, our source word embeddings will always have some vocabulary in common (otherwise we cannot generate a meta-embedding). Thus, we generate the mapping dictionaries as the intersection of the vocabulary of the word embeddings to be ensembled.

3.1.2 OOV treatment

When combining word embeddings we need to deal with the problem of having a missing word embedding in one of the source embeddings. In this case, one source embedding E1 has a representation for the word W while another source embedding E2 does not have a representation for that word. Inspired by Speer et al. (2017), once placed in a common space, we approximate a synthetic representation for a missing word (W) in one source embedding (E2) by using the ten most similar word representations of W in the other source embedding (E1). That is, first we calculate the set of ten nearest neighbours of W by cosine similarity in E1. Second, we obtain the corresponding embeddings in E2 of the set of nearest neighbour words. Finally, we assign the centroid in E2 of this set of nearest neighbours as the new word representation for W in E2. In this way, we ensure that both source representations have the same vocabulary placed in a common space.

3.1.3 Averaging

Finally, once projected two source word embeddings to a common space, we still need to combine them into a single word representation. The simplest way is by averaging the two projected word representations.

It should be noted that when generating cross-lingual meta-embeddings we also apply exactly the same process described in Section 3.1 with at least one source cross-lingual embedding. That is, projection (3.1.1), OOV generation (3.1.2) and averaging (3.1.3). After projecting both embeddings to the same common space using VecMap, for every missing word in both source embeddings, we first apply the same OOV generation process described above and, secondly, we apply averaging. Thus, when combining a monolingual Spanish embedding and a cross-lingual one, we will also create new synthetic embeddings for all missing words in the corresponding sets (possibly new English word embeddings in

[6] As a target common space we can use any of the ones provided by the original word embeddings.

the Spanish set and possibly some new Spanish word embeddings in the cross-lingual set). Thanks to this, the meta-embedding creation process can help to improve both languages. This is specially interesting when good representations are available only for one language.

3.2 Advantages

We consider that in addition to the experimental results that we will provide in the following sections, it is also important to describe the rationale behind the method and why it is superior to alternative approaches.

3.2.1 No information loss

We know that the concatenation (assuming that all the vectors are normalized to have the same length) can combine any number of source embeddings without loss of information, since none of the source embeddings is altered in the final embedding. Coates and Bollegala (2018) proved that the concatenation and the average hold a strong relationship. More precisely, lets suppose that we have the embeddings E_1 and E_2 with the same number of dimensions d and we want to calculate the euclidean distance between the words u and v. We denote the euclidean distance between u and v in both embeddings as D_{E_1} and D_{E_2}. If we concatenate both embeddings, the resulting euclidean distance will be $\sqrt{(D_{E_1})^2 + (D_{E_2})^2 - 2D_{E_1}D_{E_2}cos(\Theta)}$. We can express the concatenation of two vectors as the sum of the two vectors padding one of them d position to the right and the other one d to the left, this means that the vectors are orthogonal to each other, that is if we calculate the dot product between them we will obtain 0, so in the previous expression $-2D_{E_1}D_{E_2}cos(\Theta)$ equals 0. We can simplify it as follows $\sqrt{(D_{E_1})^2 + (D_{E_2})^2}$. On the other side, if we average both embeddings the resulting euclidean distance will also be $\sqrt{(D_{E_1})^2 + (D_{E_2})^2 - 2D_{E_1}D_{E_2}cos(\Theta)}$, however in the case of averaging we can not assume that the vector are orthogonal, so the term $-2D_{E_1}D_{E_2}cos(\Theta)$ if what differentiates the concatenation and the average. Coates and Bollegala (2018) mathematically prove that this term should be a small number without a big impact in the final result, but in the experiments that we did in our previous research work García-Ferrero (2018) we found that in practice, average offers significantly worse performance than concatenation.

Mapping the source embeddings to the same vectors space the average becomes equal to the concatenation. Thanks to this we can obtain the same results than the concatenation without increasing the dimensionality of the resulted meta embedding.

3.2.2 OOV handling

Each source word embedding use to have a different vocabulary since it has been trained on different resources. They use to share many common words, but many uncommon words are not present in every word embedding set making the combination difficult. The easier approach is to just discard them, but this approach results on a meta embedding with a smaller vocabulary that the source embeddings. A good meta embedding combination

method should be able to handle OOV words correctly and generate a meta embedding that contains all the words from all the source embeddings (the union of the vocabularies of the source embeddings).

Let's imagine that we have the source embeddings E_1, E_2, E_3 and E_4 and we want to combine them by averaging, but E_3 does not have a representation for the word u. If we just average the representations for u from the other 3 source embeddings, the result vector will be located in a different vector space than the vectors for the rest of the words, making this word vector useless for any future calculation. However, if we normalize all the embeddings to have the same length and we map all the embeddings to the same vector space, we expect the representation for u in all the embeddings to be very close. This means that if E_3 would have had a representation for u it would have been very similar to the representation for u in the rest of the source embeddings, so now the average of the representations for u in E_1, E_2 and E_4 will be a good approximation of the result that we would have obtained if E_3 would have had a representation for u. The embedding mapping by its own is a good OOV handling method. In our research we found that we can further improve the results using the OOV generation algorithm described in Section 3.1.2 to generate an approximate representation for u in E_3 (Section 4.2.1).

3.2.3 Combine everything!!

Our methods allows combining any word embedding from any source, method and language. We can combine embeddings trained using text corpora and embeddings trained using knowledge-based methods. Or even we can combine monolingual embeddings in different languages with multi-lingual embeddings. The combination can be done by just writing one command in a terminal and we just need to decide one hyperparameter, which is the vector space to which we will map all the source embeddings (can be any of the source pre-trained word embeddings). In our experiments, we found that the vector space chosen to map all the embeddings does not have a significant impact on the performance. Also, our method will generate the meta-embeddings very fast compared to other previously proposed methods. None of the meta embedding generated in this research took more than 6 hours.

4 Word Similarity Experiments

In this chapter we will describe the word similarity task (Section 4.1), the datasets that we use for this task (Section 4.1.1), the methodology for evaluating word embeddings and meta-embeddings (Section 4.1.2) and the results obtained (Section 4.2.2). Finally Section 4.3 concludes with a short summary of the chapter.

4.1 Word Similarity

Word Similarity is an intrinsic evaluation task. We compare our word embeddings against human judgements on words relations. This method is based in the idea that distances between words could be evaluated through the human judgements on the actual semantic distances between these words. For example the distance between "house" and "flat" defined in the continuous interval [0,1] could be 0.7 since both words are similar, but not exactly the same thing. The distances between pairs are also calculated in the word embedding space. The more similar these results are to the human judgements, the better are the embeddings. This method roots back to 1965 when trying to test the distributional hypothesis. Rubenstein and Goodenough (1965) conducted a test on human judgements on word semantic similarity. This way, datasets to evaluate word similarity provide a set of pairs of words with the corresponding score given by humans. This score is known as gold standard. Listing 4.1 shows the first ten pairs of words from the MTurk-287 dataset as an example of a word similarity dataset. In this dataset the similarity is defined in the continuous interval [0,5]. In the example we can see that the words *mistake* and *error* receive a high similarity score, since they are almost synonyms. However, the score is not 5 because they do not have the exact same meaning. On the other side, *latin* and *credit* receive a low similarity score because their meaning is not similar or related. This dataset evaluates semantic relatedness as shown by the pair *plays* and *losses*. These words have not a similar meaning. However they have a high similarity score in this dataset because there is a strong relation between them (when you play a game, you can loose).

```
episcopal   russia  2.75
water   shortage    2.714285714
horse   wedding 2.266666667
plays   losses  3.2
classics    advertiser  2.25
latin   credit  2.0625
ship    ballots 2.3125
mistake error   4.352941176
disease plague  4.117647059
sake    shade   2.529411765
```

Listing 1: Example of a word similarity dataset: 10 first pairs of words from the MTurk-287 dataset

This method has become one of the most popular methods for evaluating word embeddings because it is computationally inexpensive and fast. However it is also criticized, specially for the subjective factor in of the human judgements. Linguistic, psychological,

social factors or even the assessors getting tired can influence the judgement. Another strong criticism is the ambiguity of the task since different definitions for "semantic similarity" are given by different authors (Faruqui et al., 2016).

4.1.1 Datasets

In this section we describe the datasets that we use for evaluating word embeddings and meta-embeddings.

As explained in the Section 4.1 different authors give different definitions for *semantic similarity*. This means that datasets notably differ from each other. The datasets available for the task can be divided in two main groups. The ones that measures semantic similarity and the ones that measure semantic relatedness. Datasets that study the similarity between words generally capture synonymy relationships between words. Sometimes also hyperonymy and hyponymy relationships. Let's use as an example the pair of words *coast* and *shore* that are synonyms. In a dataset that measures semantic similarity this pair will have a very high score. In a dataset that measures semantic relatedness they will also have a very high scores since being synonyms means that a strong relationship between exists. However, if we use as an example the pair of words *clothes* and *closet* in a dataset measuring semantic word similarity they will have a very low score. clothes and closet are not similar in any way since they are completely different objects. However in a dataset measuring semantic relatedness they will have a high score since they hold a strong relationship, people generally store their clothes in a closet. [7]. Section 1 at the end of this chapter summarizes this classification.

In our evaluation we used all the datasets available for the word similarity task. We will briefly describe them:

1. SimVerb-3500 (Gerz et al., 2016). 3500 pairs of verbs, judged by semantic similarity, on a scale from 0 to 4.

2. MEN Bruni et al. (2014) (Acronym for Marco, Elia, and Nam). 3000 pairs of words judged by semantic relatedness on a scale from 0 to 50. This dataset is divided into three sets, one of them with the objective to be used as training for different types of algorithms, another with the objective to be used as a test, and another that combines both, we use the last one in our evaluation.

3. RW (Luong et al., 2013) (Acronym for Rare Words), 2034 pairs of words, with the peculiarity that they are rarely used words, with few appearances in the training corpus. The pairs are judged by semantic similarity on a scale from 0 to 10.

4. SimLex-999 (Hill et al., 2015). 999 pairs of words judged very strictly by semantic similarity on a scale of 0 to 10.

5. MTurk-771 (Halawi et al., 2012a) (Acronym for Mechanical Turk), 771 pairs of words judged by semantic relatedness on a scale from 0 to 5.

[7] Example borrowed from: `https://fh295.github.io/simlex.html`

6. MTurk-287 (Radinsky et al., 2011). (Acronym for Mechanical Turk), 287 pairs of words judged by semantic relatedness on a scale from 0 to 5. The pairs of word in this dataset differ from the ones in the MTurk-771 dataset.

7. WordSim-353 (Finkelstein et al., 2002). 353 pairs judged by semantic similarity on a scale of 0 to 10. The original, in which the dataset is divided into two sets, called set1 and set2. However, many researchers consider that the instructions given to the judges on similarity and association were ambiguous, causing both to be mixed in the evaluation. So there is a second division proposed by Agirre et al. (2009) where the 353 words are divided into two subsets. One whose pairs are focused on measuring the semantic similarity between words and the other whose pairs are focused on measuring the semantic relatedness between words. We use the latest proposed subsets in our results.

8. Verb-142 (Baker et al., 2014). 143 pairs of verbs judged by semantic similarity on a scale from 0 to 4.

9. YP-130 (Yang and Powers, 2005) (Acronym for Yang and Powers), 130 pairs of verbs judged by semantic similarity on a scale from 0 to 4.

10. RG-65 (Rubenstein and Goodenough, 1965). (Acronym for Rubenstein and Goodenough). 65 pairs of words judged by semantic similarity on a scale of 0 to 4.

11. MC-30 (Miller and Charles, 1991). (Acronym for Miller and Charles) a subset of RG-65 containing 10 pairs with high semantic similarity, 10 with medium semantic similarity and 10 with low similarity.

12. SemEval-2017 (Camacho-Collados et al., 2017). 500 pairs assessed by semantic similarity with a scale from 0 to 4. This dataset was prepared for the SemEval-2017 Task 2 (Multilingual and Crosslingual Semantic Word Similarity). This dataset contains not only words, but also collocations such as "autonomous car", since our embeddings contain representation for individual words, not collocations, we exclude collocations from the evaluation reducing the dataset to 388 pairs.

4.1.2 Methodology

In this section we describe the methodology that we follow to evaluate word embeddings or meta-embeddings.

We follow the same well-standardized methodology used in previous research. The similarity between two words has been calculated as the cosine similarity between their word representations. The correlation betweeen the similarities calculated using the word embeddings and the gold scores calculated by humans provided by the datasets has been calculated using Spearman Correlation Spearman (1904).

$$\text{Cosine similarity} = cos(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{||\boldsymbol{x}|| \cdot ||\boldsymbol{y}||}$$

Datasets	
Semantic Similarity	Semantic Relatedness
SimVerb-3500	MEN
RW	MTurk-287
SimLex999	MTurk-771
WS353 similarity	WS353 relatedness
Verb 143	
RG 65	
MC30	
SemEval-2017	

Table 1: Datasets classified by the semantic properties that they measure (Semantic similarity or Semantic Relatedness)

However while this procedure is well-standardized, the methodology to deal with Out-of-Vocabulary (OOV) words is not. When evaluating word embeddings in a word similarity dataset we usually face the OOV, since the embedding may not have a representation for every word in the dataset.

Embedding	Cov. All	Cov. Rel	Cov. Sim
W2V	97.4	98.3	97.3
GV	99.8	100	99.7
FT	99.8	100.0	99.7
UKB	89.0	94.4	87.3
AR	94.2	99.7	92.1
P	91.8	97.1	90.2
LV	99.6	100.0	99.5

Table 2: Coverage in the word similarity datasets

Table 2 shows the coverage of the word embeddings in the word similarity datasets. UKB presents the lower coverage while FastText almost full coverage. Previous research differ on how to deal with this problem. Some researchers assign an arbitrary vector for the missing word (a zero vector or the average of all word embeddings in the vocabulary). Other researchers prefer to assign a predefined similarity (i.e. 0.5) to all the pairs where we have an OOV word. Others prefer not taking into account the pairs of words when there is no representation for at least one of the words. We adopt this last option. We found that when assigning arbitrary vectors or similarity, the scores are probably bad and significantly differ from the gold standard. When generating a meta embedding, the final vocabulary increases. So, even if the new word representations are worse than the ones from the source embeddings, we can obtain a higher Spearman correlation just because we have artificially introduce less arbitrary results. This can lead to the wrong conclusion that a meta embedding generation method produces representations with an improved quality when in reality it is not. Since what we want is to test if the new representations generated

by our method are better than the ones in the source embeddings we discard those pairs of words when there is no representation for at least one them. However, It is also important to mention that we are using pre-trained embeddings with large vocabularies than can go up to millions of words, and the generated meta-embeddings combining this vocabularies obtain even larger ones. This mean that as we can see in Table 2 all the embeddings and meta-embeddings achieve almost one hundred percent coverage. In fact, this decision has a very little impact in the final results since there are few OOV words.

4.2 Results

In this section we show the results obtained in the word similarity task. First, in Section 4.2.1 we compare our meta-embeddings against some baseline methods (concatenation, dimensionality reduction and average) and alternative meta-embeddings on English. We also evaluate our meta-embeddings in a multilingual and cross lingual setting (Section 4.2.2).

4.2.1 English Word Similarity

In this section we first compare our meta-embeddings against some baseline ensemble methods, and then compare our method against other meta-embeddings methods. For this comparison we have used all the datasets described in section 4.1.1. Since there is a large number of datasets and it would be very difficult to compare them all, we just report the average of the results of all the datasets (Av), the average of those datasets that measure word similarity (Sim), and the average of the ones that measure word relatedness (Rel). WS353 dataset is divided in two subsets Agirre et al. (2009) measuring semantic similarity and semantic relatedness. We use the corresponding one to calculate the average of the datasets that measure word similarity and the average of the ones that measure word relatedness. To calculate the average of all the datasets we use the WS353 dataset containing both subsets. We do not take into account the MC-30 dataset for the averages since it is a subset of the RG-65 dataset. In this section all the meta-embeddings have been mapped to the vector space of the English FastText (Common Crawl, 600B tokens).

We compare our meta-embeddings with three baseline methods.

- **Concatenation**: For each word we concatenate the representation for that word in each source embedding. To ensure that every source embedding contributes equally in the meta-embedding we normalize them using the l_2 norm. This method produces meta-embeddings of high quality but very difficult to manage because its very high dimensionality.

- **Dimensionality reduction**: The most popular approach to solve the dimensionality problem is to apply dimensionality reduction algorithms. We report the results obtained using DRA Raunak (2017) when reducing the dimensionality to 300 dimensions of the meta-embeddings generated by concatenation. We also tested the

implementations of PCA and truncated SVD using sklearn Pedregosa et al. (2011) but DRA obtains the best results.

- **Averaging**: For each word, we calculate the average of the representation for that word. To ensure that every source embedding contributes equally, we normalize the source embeddings using l_2 norm. This is another way to approximate the results of the concatenation as described in the section 3.2.1.

For a fair comparison we also apply the OOV approach described in Section 3.1.2 when concatenating or averaging word embeddings. We also test our meta-embeddings with respect to alternative approaches.

- **Autoencoding Meta-Embeddings** (AAEME) (Bollegala and Bao, 2018) uses autoencoders to generate meta-embeddings. Autoencoders are an unsupervised learning algorithm that first compress the input in a space of latent variables and then reconstructs the input based on the information encoded in those latent variables. This method aims to learn meta-embeddings by reconstructing multiple source embeddings. This method comes in three flavours, DAEME, CAEME and AAEME. We used the last one because it obtains better results. We tested this method with the default parameters and enabling the option to generate OOV word representations.

- **Locally Linear Meta Embedding Learning** (LLE) (Bollegala et al., 2017) consist in two steps. A reconstruction step and a projection step. In the reconstruction step the embeddings of each word is represented by the linear weighted combination of the embeddings of its nearest neighbours. In the projection step they compute the meta embedding of each word such that the nearest neighbours in the source embedding spaces are embedded closely to each other in the meta-embedding space. We tested this method with the same parameters used in the paper.

- **Numberbatch** (N) (Speer et al., 2017) claims to be the best pre-trained word embeddings available. It combines knowledge encoded in ConcepNet, word2vec, GloVe and OpenSubtitles 2016 using concatenation, dimensionality reduction (truncated SVD) and a variation on retrofitting (Faruqui et al., 2015). Results using Numberbatch version 17.06 are reported.

- We also tested several configurations of contextual embeddings such as those from ELMo (Peters et al., 2018) or BERT (Devlin et al., 2019) but as this task does not include context, they obtain much lower results than the static ones. Using BERT[8], the configuration that provided the best results was using the sum of the last four layers of the model. This configuration scored an average result of 33.3, much lower than the results from any other static word embedding tested in this section.

[8]"bert-base-uncased" model

Embedding	Mturk 287	RW
FastText	**72.6** (100)	**59.5** (98.1)
UKB	64.6 (80.1)	45.0 (68.9)
Concatenation	71.5 (100)	51.4 (99.9)
Concatenation (Gen)	**72.7**	55.9
Average	66.2	50.2
Average (Gen)	68.1	54.8
Mapping + Average	71.6	54.5
Mapping + Average (Gen)	71.8	**58.7**

Table 3: Results of the different methods to handle OOV during the meta embedding generation method. (Gen) indicate that we are using the OOV generation algorithm. The coverage of each embeddings for each dataset is indicated between brackets after the results.

In our first experiment, we focused on evaluating our word approximation algorithm (Section 3.1.2) and the effect of the mapping in the handling of the OOV words during the generation of the meta embedding (Section 3.2.2). Table 3 reports the results of our meta-embeddings combining the embedding with larger (FastText) and smaller (UKB) coverage. We present the results in the two datasets where UKB has smaller coverage (Mturk287 and RW). In the case of concatenation, we compare our OOV generation algorithm with the most common approach appearing in the literature. That is, using a vector of zeros. Regarding average, we compare our approach against averaging using the representations available. As we can observe, our OOV generation algorithm significantly improves the results. Interestingly, the mapping using VecMap by itself is a good method for handling OOVs, increasing by a large margin the results of the average without mapping and the concatenation using zeros. We obtain even better approximations for OOV words by combining the mapping method and our word generation algorithm.

Table 4 reports the results of the single pre-trained embeddings, the baseline meta-embeddings and our meta-embeddings. As we can see in the table, in general concatenation scores are significantly better than the ones of the best pre-trained word embedding used in the concatenation. This method also outperforms the other baseline methods, averaging and dimensionality reduction. We can also see that dimensionality reduction sightly outperforms averaging, specially when combining a large number of source embeddings. Averaging causes more knowledge loose as we increase the number of word embeddings. We think that this happens because averaging multiple source-embeddings (each one in a different vector space) results in vectors that cancel each other. That is, averaging an infinite number of source-embeddings will be a embedding where all vectors are zeros. On the other side, while dimensionality reduction seems to be a better alternative to averaging, computing meta embedding with this method require much more computational effort.

Our approach leverages the advantages of the concatenation and the average or dimensionality reduction. It produces meta-embeddings of similar or higher quality than the ones produced by concatenation, with the advantage of not increasing the dimensionality of the word vectors.

1. Embeddings	Av	Rel	Sim
W2V	60.7	68.7	57.9
GV	62.4	75.3	57.8
LV	62.6	73.4	58.7
UKB	62.5	70.2	60.3
AR	65.0	61.4	**67.0**
FT	66.6	**76.5**	63.2
P	**68.2**	74.9	65.9
2. Concatenation			
W2V+UKB	68.6	75.8	66.4
W2V+AR	69.1	71.4	68.6
GV+UKB	69.4	79.0	66.3
GV+AR	70.1	75.4	68.5
FT+W2V	65.1	74.5	61.8
FT+GV	65.7	77.6	61.5
FT+UKB	70.9	78.8	68.3
FT+AR	71.5	74.6	70.8
FT+UKB+P	72.8	**81.0**	70.1
FT+UKB+AR	72.7	78.2	71.1
FT+UKB+AR+LV	72.4	79.4	69.9
FT+UKB+AR+P	**74.1**	79.9	**72.5**
3. Average			
W2V+UKB	66.8	72.4	64.9
W2V+AR	67.1	68.6	67.1
GV+UKB	67.8	76.6	65.0
GV+AR	70.5	76.2	68.8
FT+W2V	63.3	72.3	60.1
FT+GV	63.6	75.0	59.6
FT+UKB	67.1	74.8	64.9
FT+AR	69.0	70.9	68.6
FT+UKB+P	68.9	**77.0**	66.5
FT+UKB+AR	69.6	72.6	69.2
FT+UKB+AR+LV	69.2	74.1	67.7
FT+UKB+AR+P	**71.6**	75.9	**70.5**
4. Dim. reduction			
W2V+UKB	64.5	69.9	62.7
W2V+AR	67.2	67.6	67.0
GV+UKB	68.0	77.8	64.6
GV+AR	70.9	77.0	68.7
FT+W2V	65.1	73.9	61.8
FT+GV	66.0	77.5	61.8
FT+UKB	68.9	76.9	66.3
FT+AR	72.1	75.1	70.9
FT+UKB+P	71.3	**78.5**	69.0
FT+UKB+AR	71.9	76.1	70.6
FT+UKB+AR+LV	72.5	78.4	70.6
FT+UKB+AR+P	**72.8**	77.3	**71.4**
5. Mapping + Average			
W2V+UKB	68.7	77.0	66.25
W2V+AR	69.4	72.8	68.4
GV+UKB	69.4	78.6	66.4
GV+AR	70.6	75.5	69.1
FT+W2V	65.2	74.9	61.8
FT+GV	66.4	77.7	62.3
FT+UKB	71.8	79.4	69.6
FT+AR	72.1	75.4	71.2
FT+UKB+P	73.2	**81.0**	70.7
FT+UKB+AR	74.0	78.6	72.9
FT+UKB+AR+LV	73.5	79.8	71.6
FT+UKB+AR+P	**74.5**	80.0	**72.9**

Table 4: English Word Similarity results

Another interesting behaviour that we can see in Table 4 is that meta-embeddings generate better word representations when ensembling pre-trained word embeddings that encode complementary and different knowledge. The more different the method or the corpus used to generate the source embeddings, the better meta-embeddings are obtained. For instance, the combination of word representations learned from WordNet and text result in higher performance (Goikoetxea et al., 2016). This proves that meta-embeddings are able to combine the knowledge encoded in different word embeddings.

Embeddings	Av	Rel	Sim
N	73.7	78.9	71.6
AAEME(FT+UKB+P)	65.8	74.3	62.7
AAEME(FT+UKB+AR+P)	65.0	73.6	61.9
LLE(FT+UKB+P)	63.2	71.5	60.0
LLE(FT+UKB+AR+P)	64.9	71.4	62.3
FT+UKB+P	73.2	81.0	70.7
FT+UKB+AR+P	74.5	80.0	72.9
FT+UKB+N	74.1	**81.3**	71.8
FT+UKB+AR+N	**75.4**	80.4	**73.9**

Table 5: Comparison of different meta-embedding approaches

Table 5 presents the results comparing different meta-embedding approaches. For AAEME and LLE we have generated two meta-embeddings using two of the source embedding combination that resulted in better results in Table 4. For Numberbatch we use the pre-trained word embedding provided by the authors (version 17.06). Our meta-embeddings obtain similar or better results that Numberbatch and outperforms by far LLE and AAEME approaches when using the same pre-trained embeddings. We also found that, apparently, Numberbatch encoded complementary knowledge not encoded in the rest of pre-trained embeddings (probably thanks to the knowledge not included in the rest of embeddings). Adding Numberbatch to our meta-embeddings results in even better meta-embeddings. To the best of our knowledge these are the best results on the English word similarity task.

4.2.2 Multilingual and Crosslingual Word Similarity

In this section we will test our meta-embedding generation method in multilingual and crosslingual word similarity tasks. We used English and Spanish since for these languages there are more pre-trained word embeddings and evaluation datasets. First, we evaluate our cross-lingual English-Spanish meta-embeddings in the English version of the SimLex999, WS353, RG65 and SemEval datasets, and them we evaluate the same meta-embeddings in the Spanish version of the datasets. In the cross-lingual English-Spanish task we used the same cross-lingual meta-embeddings but this time for each pair in the datasets, the first word in English and the second in Spanish. In our preliminary research we found out that the best approach to generate cross-lingual meta-embeddings using our method

ME	EN	ES
ME1	Jen+FTen+Aten+Pen	Jes+FTes
ME2	Jen+Nen+FTen+ATen+Pen+UKBen	Jes+Nes
ME3	Jen+Nen	Jes+Nes
ME4	Jen+Nen+FTen+ATen	Jes+Nes

Table 6: Cross-lingual meta-embeddings

is to independently generate the best possible meta-embeddings in English and the best possible meta-embeddings in Spanish, and project them to the same space. In this section we map all of our meta-embeddings to the English-Spanish cross-lingual space offered by JOINTChyb (J). Mapping our meta-embeddings to an already cross-lingual vector space allows us to skip the dictionary induction process and the generation of the cross-lingual meta-embeddings is much faster. Also, since we are averaging the source embeddings to merge then in a common vector space, the best possible cross-lingual mapping is not necessary, although it could provide a small performance increase thanks to reducing the loss of information during the averaging.

Table 6 summarises the information of our best performing meta-embeddings. Suffixes (en) for English or (es) for Spanish indicate the language of the monolingual embeddings selected for creating the meta-embeddings. For instance, FTen corresponds to the English FastText. In the case of cross-lingual word embeddings, the suffix indicate the language of the words selected for creating the meta-embeddings. For instance, Jen corresponds to the English vocabulary from JOINTChyb. Thus, our meta-embedding ME1 includes 6 word embeddings, four English and two Spanish, all projected to the space of JOINTChyb (English-Spanish cross-lingual embedding) which is the same space for both English and Spanish. As the generated meta-embeddings show, our approach for creating meta-embeddings can ensemble a very large number of word embeddings.

Table 7 shows the multilingual results for English and Spanish, and the cross-lingual English-Spanish. We include the results of some monolingual embeddings, and the cross-lingual JOINTChyb (J) and NumberBach (N). Compared to Numberbach, our cross-lingual meta-embeddings obtain the best results for English, slightly better for Spanish. In the case of the cross-lingual results our meta-embeddings obtain a slightly lower results than Numberbach. However, our meta embedding has a much larger vocabulary than Numberbach, which means that they give an answer for almost every pair of word in the datasets, while Numberbach due to a smaller vocabulary skips giving answer to some of pair of words that include non-commonly used words. Since these words are not used commonly there are few instances of them during the training of the source embeddings resulting in low quality representations and more inaccurate results for the similarity task. Skipping the answers for these pairs Numberbach obtains an advantage over our meta-embeddings. Providing answers for the same word-pairs as Numberbach, our meta-embedding *M4enes* obtains in the cross-lingual English-Spanish a result of 80.4 which is a slightly better result than Numberbach. Thanks to these results we can probe that our meta-embedding generation method is suitable for different languages. To the best of our knowledge, these

Embedding	EN-EN	ES-ES	EN-ES
UKBen	71.9	-	-
FTen	73.5	-	-
FTes	-	60.2	-
Pen	73.6	-	-
ARen	75.7	-	-
Jenes	73.5	68.7	71.7
Nenes	80.7	75.6	**80.0**
ME1enes	81.6	69.1	74.6
ME2enes	82.4	**75.7**	79.6
ME3enes	81.1	**75.7**	79.5
ME4enes	**82.6**	**75.7**	79.6

Table 7: Multilingual and Cross-lingual word similarity results using cross-lingual meta-embeddings

are the best published results for English and Spanish in the mono-lingual word similarity task. At the same time, the approach is also suitable for a cross-lingual task achieving similar results to the current state-of-the-art.

4.3 Conclusions

In this section we have evaluated our meta-embeddings on the mono-lingual and cross-lingual word similarity task. Our method achieve a strong performance in mono-lingual word similarity, surpassing different baseline methods and other alternative approaches proposed by different authors. To the best of our knowledge, we have achieved the best results for English and Spanish in word similarity task. Our method also offers a robust performance in the English-Spanish cross-lingual word similarity task with a performance on par with the current state-of-the-art.

5 Semantic Text Similarity Experiments

In this chapter we describe the Semantic Text similarity task (Section 5.1), the dataset that we use for this task (Section 5.1.1), the methodology for evaluating word embeddings and meta-embeddings (Section 5.1.2) and the results obtained (Section 5.2). Finally section 5.3 concludes with a short summary of the chapter.

5.1 Semantic Text Similarity

Semantic Text Similarity is an extrinsic task. We compare word embeddings based on the ability to be used as vectors of characteristics of supervised automatic learning algorithms used in various NLP task. The performance of the supervised method, usually measured in a dataset for NLP tasks, is taken as a measure of the quality of the word embedding. In this case the task is Semantic Text Similarity (STS).

Semantic Text Similarity aims to calculate the degree of semantic similarity between two texts. This task is a key component of many NLP systems such as Machine Translation and evaluation, Summarization, Machine Reading, Question Answering, etc. and has received a lot of attention in recent years. For instance, if we provide the following texts *A man is cutting up a cucumber* and *A man is slicing a cucumber*, the similarity between them in the continuous interval [0,1] could be 0.8 since they are very similar but the meaning is not exactly the same. A supervised automatic learning algorithm that uses as input the word embedding for each word in the sentence is trained using these type of examples, and them the similarity between these sentences is calculated using the trained model. The more similar the results provided by the model and by the dataset are, the better is the model. If we use the same system, with the same training data, but different word embeddings as input we can compare the quality of the word embeddings based on the performance of the supervised model. Listing 5.1 shows as example of this type of dataset corresponding to the first 5 pairs of texts from the STS Benchmarks test set dataset. This dataset is scored in the continuous interval [0,5]. In this example we can see that the sentences *One woman is measuring another woman's ankle* and *A woman measures another woman's ankle.* are annotated with the highest possible similarity because their meaning is exactly the same. They just differ in one word only that does not affect the meaning in any way. On the other side, the sentences *A man is playing a harp.* and *A man is playing a keyboard.* are annotated with a low similarity. Even if they just differ in one word, playing an harp and playing a keyboard is not the same. They are very different instruments, so the meaning of the sentence has completely changed. They are not annotated with the lower possible score because, in both sentences a man is playing an instrument, so still exists some similarity between them.

```
2.500 A girl is styling her hair. A girl is brushing her hair.
3.600 A group of men play soccer on the beach. A group of boys are playing soccer on the beach.
5.000 One woman is measuring another woman's ankle. A woman measures another woman's ankle.
4.200 A man is cutting up a cucumber. A man is slicing a cucumber.
1.500 A man is playing a harp. A man is playing a keyboard.
```

Listing 2: Example of a semantic text similarity dataset: 5 first pairs of texts from the STS bechmark dataset

The gold scores provided by the semantic text similarity datasets are usually calculated by humans judges. Deciding the semantic similarity between two texts is a very time consuming task. Instead of just compare two words as in semantic word similarity (Section 4) we need to read, understand and compare two texts. Due to this for a long time only few very small datasets where available. For instance, Li et al. (2006) dataset, containing 65 sentence pairs corresponding to the dictionary definitions for the 65 word pairs in the RG-65 dataset (Rubenstein and Goodenough, 1965) or Lee et al. (2005) containing 60 documents ranging from 51 to 126 words. Choosing pairs of sentences or texts is also a challenging task, since taken random sentences usually generate unrelated pairs of sentences. Thanks to the word done by Agirre et al. (2012) and crowdsourcing platforms such as Amazon Mechanical Turk where a large number of persons can evaluate a small number of pairs of texts, large high-quality STS datasets are now available.

5.1.1 Dataset

In this section we describe the semantic texual Similarity (STS) dataset that we use for evaluating word embeddings and meta-embeddings. We use STS benchmark (Cer et al., 2017). This is a set of multilingual and cross-lingual datasets used in the STS task organized in the context of SemEval between 2012 and 2017. It includes text from image captions, news headlines and user forums. For English we used the training, development and test set provided by STSbenchmark. For Spanish we collected all the Spanish data provided in previous SemEval editions to create a new training and development dataset. This dataset was divided in two. 85% of sentences were used for training and 15% for development. We used as test the dataset provided in the SemEval 2017 STS task 1. The number of pairs for each dataset is shown in Table 8. It is important to highlight that the Ensligh dataset is almost 5 times larger than the Spanish dataset.

	Train	Dev.	Test
English	5749	1500	1379
Spanish	1296	324	250

Table 8: Number of words for each STS dataset

5.1.2 Methodology

In this section we describe the methodology followed to evaluate the word embedding and meta-embeddings. In order to evaluate our embeddings in STS we have used the system proposed by Shao (2017). The system is composed by a convolutional neural network (CNN), a comparison of semantic vectors and a fully-connected neural network (FCNN). The reason for choosing this system is a light model that can be trained very fast without the need for large computing power while obtaining very good results in the task. The system achieved the 3rd place on the primary task of SemEval 2017. The model also includes a pre-process step.

- **Pre-process**: The preprocessings include multiple steps.
  1. All punctuations are removed.
  2. All words are lower-cased, all sentence are tokenized using Natural Language Toolkit (NLTK) Bird et al. (2009).
  3. All words are replaced by the pre-trained word embeddings (or meta-embeddings). Out-of-vocabulary words are set to zero vector.
  4. All sentences are padded to a static length 30 with zero vectors.
  5. If a word appears in both sentences, a True flag is added to the word vector, otherwise, a False flag.
  6. If a word is a number, and the same appears in the other sentence, add a True flag to the word vector of the matching number in each sentence, otherwise, a False flag is added.
  7. The part-of-speed (POS) tag of every word according to NLTK is added as one-hot vector.

- **Convolutional Neural Network (CNN)**: The aim of the CCN is to transform the word level word embedding from the input to sentence level embeddings. This CNN consists on 300 one dimensional filters. The length of these filters is the same as the dimensions of the enhanced word vectors. Sentence level embeddings are calculated by max pooling over every dimension of the transformed word level embedding.

- **Comparison of semantic vectors**: To calculate the semantic similarity score of two sentences, a semantic difference vector is generated by concatenating the element-wise absolute difference and the element-wise multiplication of the corresponded paired sentence level embeddings.

$$\vec{SDV} = (|\vec{SV}1 - \vec{SV}2|, \vec{SV}1 \circ \vec{SV}2)$$

$\vec{SDV}$ is the semantic difference vector, $\vec{SV}1$ and $\vec{SV}2$ are the semantic vectors of the two sentences and $\circ$ is the Hadamard product which generates the element-wise multiplication of two semantic vectors.

Sentence pad length	30
Dimension of embeddings	300
Number of CNN layers	1
Dimension of CNN filters	1
Number of CNN filters	300
Activation function of CNN	relu
Initial function of CNN	he_uniform
Number of FCNN layers	2
Dimension of input layer	600
Dimension of the first layer	300
Dimension of second layer	6
Activation of first layer	tanh
Activation of second layer	softmax
Initial function of layers	he_uniform
Optimizer	ADAM
Batch size	339
Max epoch	6
Run Times	8

Table 9: Hyperparamers of the Neutal Network used for STS

- **Fully-connected neural network (FCNN)**: An FCNN is used the transfer the 600 dimension semantic difference vector to a probability distribution over the labels used by STS.

The system is trained 8 times. All the weights and parameters are reinitialised reach time we train the model. The model is evaluated in the development test after each epoch. The model and epoch that achieves the higher result in the development test is selected and evaluated in the test set. We used the default hyperparemeters that are shown in Table 9. To compare the results obtained from the model with respect to those from the gold standards Pearson correlation is used.

5.2 Results

Table 10 shows the results when training and evaluating the model in English. Some of our meta embedding achieve better results compared to other word embeddings evaluated. The original system presented at SemEval used GloVe pre-trained embeddings. Using our meta-embeddings the same system outperforms the original one by a large margin, proving the potential of our meta-embedding. In the same way Table 11 shows the results when training and evaluating in Spanish. For Spanish our meta-embeddings also outperform other mono-lingual embeddings, including Numberbatch selecting the Spanish vocabulary. Compared to English, in Spanish we have much less training data available thus obtaining lower results. Interestingly, the best results on the test data for Spanish are obtained when including into the meta-embeddings not just Spanish source embeddings, but also English

Embedding	Dev	Test
UKB	76.3	70.7
FT	81.7	76.1
GV	81.8	78.1
AT	81.3	75.3
P	82.4	78.8
Nen	83.5	79.6
FT+UKB+AT+P	83.5	79.7
FT+UKB+AT+N	**84.0**	80.1
FT+N+AT+P	83.9	**80.5**

Table 10: STS results training and testing on English

Embedding	Dev	Test
GVes	76.6	73.7
FTes	80.4	75.1
Jes	76.0	73.1
Nes	76.3	73.8
Nes+FTes	**81.8**	75.3
Nenes+FTen+FTes+Pen	81.0	**78.4**

Table 11: STS results training and testing on Spanish

ones. Our meta-embedding generation method is able to improve the knowledge encoded in one language with the knowledge from another language. That is, we are successfully performing transfer learning from English (where we have very good word meta-embeddings trained with very large corpora) to Spanish (where we have lower quality word embeddings).

In order to further explore the transfer-learning capabilities of our meta-embeddings, we trained a model in English and then, without seeing any Spanish example, we test the model in Spanish. That is, using the training parameters learned from English and applying them to Spanish. Our meta-embedding MEenes have been obtained by ensambling Jenes, Nenes, FTes, FTen, ATen and Pen. Our meta-embeddings outperforms other cross-lingual alternatives and they exhibit a very robust cross-lingual capabilities. Thanks to these capabilities we can achieve better results for the STS task in Spanish when training the model with English data (7K sentences) than when training the model with the smaller Spanish dataset (1K sentences) only. Our meta-embeddings are not only able to improve the results in mono-lingual STS task. They are also capable of transferring the knowledge from a resource-rich language to another language, allowing us to successfully perform STS for languages where training data is very limited or not available at all.

Embedding	Test EN	Test ES
Jenes	73.8	69.3
Nenes	**79.6**	81.8
MEenes	78.8	**82.8**

Table 12: STS results training on English and testing on Spanish.

5.3 Conclusions

In this section we have proved that our meta-embeddings exhibit solid performance in a semantic textural similarity task. Although the STS leaderboard is currently headed by contextual embeddings, these experiments confirm that our meta-embeddings outperform

other existing static embeddings and meta-embeddings. Furthermore, since contextual embeddings require huge resources, our cross-lingual approach could be useful for under-resourced languages.

6 Sequence Labelling Experiments

In this chapter we describe the Sequence Labelling tasks (Section 6.1), the datasets that we use (Section 6.1.1), the methodology for evaluating word embeddings and meta-embeddings (Section 6.1.2) and the results obtained (Section 6.2). Finally section 6.3 concludes with a short summary of the chapter.

6.1 Sequence Labelling

Sequence labelling in NLP is a type of patter recognition task that involves classifying each member of a sequence of words with a linguistic label. A very large family of NLP tasks can be formulated as sequence labelling problems. The most popular sequence labelling tasks in NLP are Part-of-speech tagging, chunking and named entity recognition:

- Part-of-speech tagging (POS tagging or POST), also known as grammatical tagging or word-category disambiguation, is a task that consists of assigning a particular part of speech for each word in sequence based on its definition and context. A Part of speech is a category of words which share a similar grammatical properties. Commonly listed English parts of speech are noun, verb, adjective, adverb, pronoun, preposition, conjunction, interjection, article and determiner. Figure 4 shows an example of Part-of-speech tagging for the sentence: *She sells seasshells on the seashore*. As we can see, each word is annotated with a POS tag. *She* is annotated as a personal pronoun. *sells* is annotated as 3rd person singular present Verb. *seashells* is annotated as a plural Noun. *on* is annotated as a preposition. *The* is annotated as a determiner. Finally, *seashore* is annotated as a singular noun.

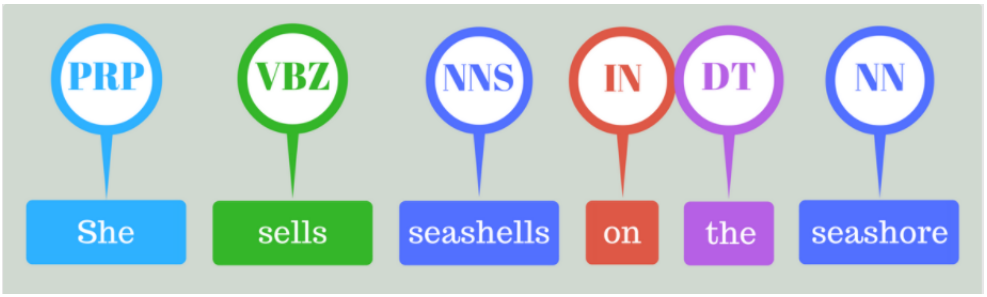

Figure 4: Example of POS tagging application on a example sentence. Source: `https://nlpforhackers.io/training-pos-tagger/`

- Chunking, also known as shallow parsing, is a task on top of POST tagging. The task consist of generating a tree whose leaves will hold POS tags and then links them to higher order units such as noun or verb phrases. Figure 5 shows as example of chunking using the sentence *John hit the ball*. We can see that each word has been first assigned a Post tag. *John* has been tagged as a Noun Phrase. *hit* has been tagged as a verb. *the* has been tagged as a determiner. And *ball* has been tagged as

a Noun. Then the leaves are linked to higher order units. *The* (Det) and *ball* (N) form a Noun Phrase. *hit* (V) and *the ball* (NP) form a Verb Phrase. Finally, *John* (NP) and *hit the ball* (VP) form a sentence.

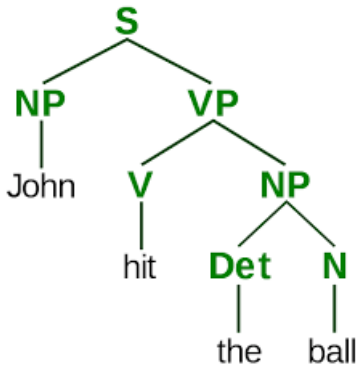

Figure 5: Example of chunking application on a example sentence. Source: `https://courses.cs.ut.ee/LTAT.01.001/2017_fall/uploads/Main/Lecture7.pdf`

- Named Entity Recognition (NER), also known as entity chunking or entity extraction is a task that consists of locating and classifying named entity mentions in unstructured text into pre-defined categories such as persons, names, locations, time expressions, monetary values, medical codes, etc. Figure 6 shows a example of NER for the text: *The Mona Lisa is 16th century oil painting created by Leonardo. It's held at the Leuvre in Paris. Mona Lisa* has been tagged as a *word* since we are speaking about the painting, not the person. *oil painting* has been tagged as a concept. *Leonardo* has been tagged as a person. Finally *Louvre* and *Paris* have been tagged as places. The system used to tag this sentence has not tagged *16th century*. However, we could have tagged it as a time expression. Note that named entities can consist of multiple words. The task it not limited to determine if a given word is an entity of any type, entities can also be multi-word expressions that we need to tag as only one named entity (e.g. oil painting).

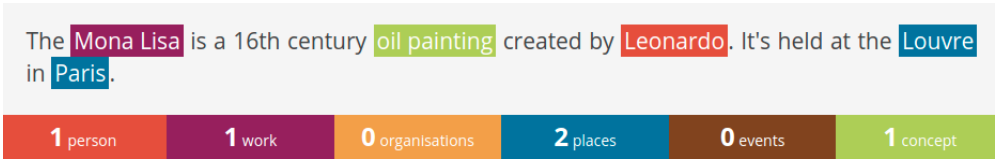

Figure 6: Example of NERC application on a example sentence. Source: `https://dandelion.eu/semantic-text/entity-extraction-demo/`

Sequence labelling, as Semantic Text Similarity (Section 5) is an extrinsic task. The performance of a supervised method for sequence labelling is taken as a measure of the quality of the word embeddings. A model is trained and them evaluated using examples annotated by humans. The more similar the model predictions are to the

ones annotated by humans the better the model is. If we use the same system with same training data, but different word embeddings as input we can evaluate the performance of the word embeddings based on the performance of the supervised model.

We experiment with POS tagging and NER in English and Spanish.

6.1.1 Datasets

For POS tagging we experiment with the Universal Dependencies 2.0 data Nivre et al. (2017). This dataset was used in the *CoNLL 2017 Multilingual Parsing from Raw Text to Universal Dependencies* shared task [9]. In Listing 6.1 we have an example of a labelled sentence from the dataset: *President Bush on Tuesday nominated two individuals to replace retiring jurists on federal.* In the example we can see how the dataset provides for each word its lemma and the POS tag, as well as other useful information. The dataset contains data form multiple languages, although we restrict our experiments to English and Spanish.

```
# sent_id = weblog-blogspot.com_nominations_20041117172713_ENG_20041117_172713-0002
# text = President Bush on Tuesday nominated two individuals to replace retiring jurists on
            federal courts in the Washington area.
1   President   President   PROPN   NNP Number=Sing                          5   nsubj   _   _
2   Bush        Bush        PROPN   NNP Number=Sing                          1   flat    _   _
3   on          on          ADP     IN  _                                    4   case    _   _
4   Tuesday     Tuesday     PROPN   NNP Number=Sing                          5   obl     _   _
5   nominated   nominate    VERB    VBD Mood=Ind|Tense=Past|VerbForm=Fin     0   root    _   _
6   two         two         NUM     CD  NumType=Card                         7   nummod  _   _
7   individuals individual  NOUN    NNS Number=Plur                          5   obj     _   _
8   to          to          PART    TO  _                                    9   mark    _   _
9   replace     replace     VERB    VB  VerbForm=Inf                         5   advcl   _   _
10  retiring    retire      VERB    VBG VerbForm=Ger                         11  amod    _   _
11  jurists     jurist      NOUN    NNS Number=Plur                          9   obj     _   _
12  on          on          ADP     IN  _                                    14  case    _   _
13  federal     federal     ADJ     JJ  Degree=Pos                           14  amod    _   _
14  courts      court       NOUN    NNS Number=Plur                          11  nmod    _   _
15  in          in          ADP     IN  _                                    18  case    _   _
16  the         the         DET     DT Definite=Def|PronType=Art             18  det     _   _
17  Washington  Washington  PROPN   NNP Number=Sing                          18  compound_   _
18  area        area        NOUN    NN  Number=Sing                  14  nmod _ SpaceAfter=No
19  .           .           PUNCT   .   _                                    5   punct   _   _
```

Listing 3: Example sentence from the Universal Dependencies 2.0 english data

For NER we used the well-known CoNLL 2002 data for Spanish and the 2003 one for English. In Listing 6.2 we can see an example of a labelled sentence from the dataset: *U.N. official Ekeus heads for Baghdad.*. In this sentence the word *U.N* is labelled as and organization entity (I-ORG). The word *Ekeus* as a Person entity (I-PER) and the word *Baghdad* as a location entity (I-LOC). The rest of the words are not classified as an entity that the system must recognize (O).

[9] http://universaldependencies.org/conll17/

Embedding	en	es
Flair	92.34	93.12
Flair+kom (Komninos and Manandhar, 2016)	94.07	-
Flair+GV (news)	93.78	94.37
Flair+FT (news)	93.84	**94.77**
Flair + Jenes+N+FTes+FTen+ATen+Pen	**94.18**	94.68

Table 13: Word accuracy POS results for English and Spanish evaluated on UD 2.0

```
U.N.         NNP  I-NP  I-ORG
official     NN   I-NP  O
Ekeus        NNP  I-NP  I-PER
heads        VBZ  I-VP  O
for          IN   I-PP  O
Baghdad      NNP  I-NP  I-LOC
.            .    O     O
```

Listing 4: Example sentence from the CoNLL 2003 data for English

6.1.2 Methodology

We use the Flair system[10] for the Part-of-Speech (POS) and Named Entity Recognition (NER) sequence labelling tasks. Flair implements a recurrent neural network (RNN) architecture (Cho et al., 2014) to represent documents, modelling text as a sequence of characters passed to the RNN which at each point in the sequence is trained to predict the next character (Akbik et al., 2018). We used this architecture leveraging the pre-trained word embedding and meta-embedding models combined with the Flair character-based contextual embeddings for English and Spanish. Flair has been successfully applied to sequence labelling tasks obtaining best results for a number of public benchmarks Akbik et al. (2018), outperforming current successful approaches such as BERT and ELMO (Devlin et al., 2019; Peters et al., 2018) in NER and POS evaluations. For the NER task we compare the results produced by the model and the results provided by the dataset gold standards using F1 score. For the POS tagging task we use word accuracy.

6.2 Results

For POS tagging we experiment with the Universal Dependencies 2.0 data. For NER we used the well-known CoNLL 2002 data for Spanish and the 2003 one for English. Both datasets provide train, development and test partitions. We train the neural network off-the-shelf using the parameters used in Akbik et al. (2019). We compare the previous best configurations of Flair (Akbik et al., 2019) replacing the static word embeddings with our own meta-embeddings.

[10] `https://github.com/zalandoresearch/flair`

Embedding	en	es
Flair	90.85	86.47
Flair + GV (news)	**92.74**	87.70
Flair + FT (news)	92.35	87.60
Flair + Jenes+Nes+FTes	92.59	**88.19**

Table 14: F1 micro NER results for English (CoNLL 2003) and Spanish (CoNLL 2002).

In Table 13 we can see that for POS tagging the best performing meta-embedding combines the cross-lingual embeddings JOINTChyb, Numberbatch (full vocabulary), both Spanish and English FastText, Attract Repel and Paragram.

Table 14 shows that the best meta-embeddings we used for NER is a combination of the cross lingual embedding JOINTChyb, the Spanish vocabulary of Numberbatch and FastText trained in Spanish.

The results show that using our meta-embeddings help to improve results for POS tagging in English and for NER in Spanish. To the best of our knowledge, these are the best results published so far for these two benchmarks.

6.3 Conclusions

In this section we have proved that our meta-embeddings in combination with the Flair model, can also help to improve the results for POS tagging in English and for NER in Spanish. As for STS, we would like to highlight that the cross-lingual character of our meta-embeddings helps also the monolingual tasks.

7 Conclusions

We have presented a new-meta embedding generation approach that can integrate efficiently and effectively combine multiple word embeddings derived from complementary techniques, textual sources, knowledge bases and languages. Our method is able to combine these sources without increasing the dimensionality of the representations and without producing a loss of information. We also propose a method to successfully deal with the out-of-vocabulary problem, which allows combining source embeddings with different vocabularies. The vocabulary of the generated meta-embedding will be the union of the vocabularies of the source embeddings. Our OOV method also allows us to combine word embeddings in different languages, which gives to our meta-embeddings interesting transfer-learning capabilities. Our meta-embeddings help to improve performance over previous meta- and static word embeddings, obtaining excellent results in several tasks, including Word Similarity, Semantic Textual Similarity and Sequence Labelling.

The main contributions of our work are:

- We have carried out an in-depth study of the current state-of-the-art regarding word embeddings and meta-embeddings.
- We propose a new method to efficiently and effectively combine word embeddings from complementary techniques, textual sources, knowledge bases and languages.
- The code and some of our best meta-embeddings are publicly available[11] under a copyleft licence in the hope that they will be useful in future research and initiatives.
- A summary of the research carried out in this Master's thesis have also resulted in a paper submitted to the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20). The paper is currently under review and we are waiting for feedback.

7.1 Future work

To conclude we want to present some of the future research lines that might be interesting. Our meta-embedding have shown interesting transfer-learning capabilities. For both Semantic Textual Similarity and Sequence Labelling, the knowledge encoded in the English pre-trained word embeddings seems to help the performance in Spanish evaluation task when combined with Spanish pre-trained word embeddings. We plan to investigate further this technique to improve performance for less-resourced languages by applying cross-lingual meta-embeddings. We are especially interested in testing the transfer-learning capabilities of our meta-embeddings in the Basque language. We also plan to extend the work done to enable the generation of cross-lingual embeddings just from monolingual ones taking advantage of the dictionary induction capabilities of VecMap.

[11] `https://github.com/ikergarcia1996`

References

Rodrigo Agerri, Yiling Chung, Itziar Aldabe, Nora Aranberri, Gorka Labaka, and German Rigau. Building named entity recognition taggers via parallel corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan, May 2018. European Languages Resources Association (ELRA). URL `https://www.aclweb.org/anthology/L18-1557`.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor 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.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada, 7-8 June 2012. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/S12-1051`.

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard H. Hovy, Kai-Wei Chang, and Nanyun Peng. Near or far, wide range zero-shot cross-lingual dependency parsing. *CoRR*, abs/1811.00570, 2018. URL `http://arxiv.org/abs/1811.00570`.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018*, 2018.

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. Pooled contextualized embeddings for named entity recognition. In *NAACL*, pages 724–728, 2019.

Marco Idiart Alexandre Salle and Aline Villavicencio. Matrix factorization using window sampling and negative sampling for improved word representations. *ACL*, 2016.

Abdulrahman Almuhareb. *Attributes in lexical acquisition*. PhD thesis, University of Essex, 2006.

Mikel Artetxe, Gorka Labaka, and Eneko 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.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, 2017.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5012–5019, 2018a.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, 2018b.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, 2018c.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Amir Bakarov. A survey of word embeddings evaluation methods. *arXiv preprint arXiv:1801.09536*, 2018.

Simon Baker, Roi Reichart, and Anna Korhonen. An unsupervised model for instance level subcategorization acquisition. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 278–289, 2014.

M Baroni, S Evert, and A Lenci. Esslli 2008 workshop on distributional lexical semantics. *Hamburg, Germany: Association for Logic, Language and Information*, 2008.

Marco Baroni and Alessandro Lenci. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10. Association for Computational Linguistics, 2011.

Marco Baroni, Brian Murphy, Eduard Barbu, and Massimo Poesio. Strudel: A corpus-based semantic model based on properties and types. *Cognitive science*, 34(2):222–254, 2010.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=944919.944966`.

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

Philip Blair, Yuval Merhav, and Joel Barry. Automated generation of multilingual clusters for the evaluation of distributed representations. *arXiv preprint arXiv:1611.01547*, 2016.

Danushka Bollegala and Cong Bao. Learning word meta-embeddings by autoencoding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1650–1661, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1140.

Danushka Bollegala, Alsuhaibani Mohammed, Takanori Maehara, and Ken-ichi Kawarabayashi. Joint word representation learning using a corpus and a semantic lexicon. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2690–2696. AAAI Press, 2016. URL http://dl.acm.org/citation.cfm?id=3016100.3016278.

Danushka Bollegala, Kohei Hayashi, and Ken-ichi Kawarabayashi. Think globally, embed locally - locally linear meta-embedding of words. *CoRR*, abs/1709.06671, 2017. URL http://arxiv.org/abs/1709.06671.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47, 2014.

José Camacho-Collados and Roberto Navigli. Find the word that does not belong: A framework for an intrinsic evaluation of word vector representations. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 43–50, 2016.

Jose Camacho-Collados, Mohammad Taher Pilehvar, Nigel Collier, and Roberto Navigli. Semeval-2017 task 2: Multilingual and cross-lingual semantic word similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 15–26, 2017.

Cristian Cardellino. Spanish Billion Words Corpus and Embeddings, March 2016. URL https://crscardellino.github.io/SBWCE/.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2001. URL https://www.aclweb.org/anthology/S17-2001.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL http://arxiv.org/abs/1406.1078.

Joshua Coates and Danushka Bollegala. Frustratingly easy meta-embedding–computing meta-embeddings by averaging source word embeddings. In *Proceedings of the 2018*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 194–198, 2018.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390177. URL `http://doi.acm.org/10.1145/1390156.1390177`.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November 2011. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1953048.2078186`.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *ACL 2019*, 2019.

Yerai Doval, Jose Camacho-Collados, Luis Espinosa-Anke, and Steven Schockaert. Improving cross-lingual word embeddings by meeting in the middle. In *Proceedings of EMNLP*. Association for Computational Linguistics, 2018.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*, 2018.

Manaal Faruqui and Chris 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.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, 2015.

Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. Problems with evaluation of word embeddings using word similarity tasks. *arXiv preprint arXiv:1605.02276*, 2016.

Christiane Fellbaum. *WordNet: An electronic lexical database*. Cambridge, MA: MIT Press, 1998.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. *ACM Transactions on information systems*, 20(1):116–131, 2002.

John Rupert Firth. A synopsis of linguistic theory 1930-1955 in studies in linguistic analysis, philological society, 1957.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, 2013.

Bin Gao, Jiang Bian, and Tie-Yan Liu. Wordrep: A benchmark for research on learning word representations. *arXiv preprint arXiv:1407.1640*, 2014.

Iker García-Ferrero. Estudio de word embeddings y métodos de generación de meta embeddings. *ADDI http://hdl.handle.net/10810/29088*, 2018.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org, 2017.

Daniela Gerz, Ivan Vulić, Felix Hill, Roi Reichart, and Anna Korhonen. Simverb-3500: A large-scale evaluation set of verb similarity. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2173–2182, 2016.

Sahar Ghannay, Benoit Favre, Yannick Esteve, and Nathalie Camelin. Word embedding evaluation and combination. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 300–305, 2016.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In *Proceedings of the NAACL Student Research Workshop*, pages 8–15, 2016.

Josu Goikoetxea, Aitor Soroa, and Eneko Agirre. Random walks and neural network language models on knowledge bases. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1434–1439, Denver, Colorado, May–June 2015. Association for Computational Linguistics. doi: 10.3115/v1/N15-1165. URL `https://www.aclweb.org/anthology/N15-1165`.

Josu Goikoetxea, Eneko Agirre, and Aitor Soroa. Single or multiple? combining word representations independently learned from text and wordnet. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2608–2614. AAAI Press, 2016. URL `http://dl.acm.org/citation.cfm?id=3016100.3016266`.

Josu Goikoetxea, Aitor Soroa, and Eneko Agirre. Bilingual embeddings with random walks over multilingual wordnets. *Know.-Based Syst.*, 150(C):218–230, June 2018. ISSN 0950-7051. doi: 10.1016/j.knosys.2018.03.017. URL `https://doi.org/10.1016/j.knosys.2018.03.017`.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1406–1414. ACM, 2012a.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1406–1414. ACM, 2012b.

Z. Harris. Distributional structure. *Word, 10(23): 146-162.*, 1954.

F Hill, Kyunghyun Cho, Sebastien Jean, C Devin, and Yoshua Bengio. Not all neural embeddings are born equal. In *NIPS 2014 Workshop on Learning Semantics, December 2014*, 2014.

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

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. Sensembed: Learning sense embeddings for word and relational similarity. 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)*, pages 95–105, 2015.

Mario Jarmasz and Stan Szpakowicz. Roget's thesaurus and semantic similarity. *Recent Advances in Natural Language Processing III: Selected Papers from RANLP*, 2003:111, 2004.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *CoRR*, abs/1611.04558, 2016. URL `http://arxiv.org/abs/1611.04558`.

David A Jurgens, Peter D Turney, Saif M Mohammad, and Keith J Holyoak. Semeval-2012 task 2: Measuring degrees of relational similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 356–364. Association for Computational Linguistics, 2012.

Sanjay Kamath, Brigitte Grau, and Yue Ma. Predicting and integrating expected answer types into a simple recurrent neural network model for answer sentence selection. In *20th International Conference on Computational Linguistics and Intelligent Text Processing*, 2019.

Alexandros Komninos and Suresh Manandhar. Dependency based embeddings for sentence classification tasks. In *NAACL 2016*, 2016.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. Text classification algorithms: A survey. *Information*, 10 (4):150, 2019.

Mikhail Kozhevnikov and Ivan Titov. Cross-lingual transfer of semantic role labeling models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1190–1200, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P13-1117`.

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

Angeliki Lazaridou, Georgiana Dinu, and Marco 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)*, pages 270–280, 2015.

Michael D Lee, Brandon Pincombe, and Matthew Welsh. An empirical evaluation of models of text document similarity. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 27, 2005.

Wei Li and Andrew McCallum. Semi-supervised sequence modeling with syntactic topic models. In *AAAI*, volume 5, pages 813–818, 2005.

Yuhua Li, David McLean, Zuhair Bandar, James O'Shea, and Keeley Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering*, 18:1138–1150, 09 2006. doi: 10.1109/TKDE.2006.130.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, and Xuezhe Ma. Choosing Transfer Languages for Cross-Lingual Learning. *arXiv e-prints*, art. arXiv:1905.12688, May 2019.

Ang Lu, Weiran Wang, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Deep multilingual correlation for improved word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 250–256, 2015.

Thang Luong, Richard Socher, and Christopher Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, 2013.

Thang Luong, Michael Kayser, and Christopher D Manning. Deep neural language models for machine translation. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 305–309, 2015.

Chaitanya Malaviya, Matthew R. Gormley, and Graham Neubig. Neural factor graph models for cross-lingual morphological tagging. *CoRR*, abs/1805.04570, 2018. URL `http://arxiv.org/abs/1805.04570`.

Diego Marcheggiani and Ivan Titov. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515, 2017.

Oren Melamud, Jacob Goldberger, and Ido Dagan. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, 2016.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013a.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, 2013b.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the 11th Language Resources and Evaluation Conference*, Miyazaki, Japan, May 2018. European Language Resource Association. URL `https://www.aclweb.org/anthology/L18-1008`.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

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

Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2009.

Aditya Mogadala and Achim Rettinger. Bilingual word embeddings from parallel and non-parallel corpora for cross-language text classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 692–702, 2016.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association for Computational Linguistics*, 5:309–324, 2017.

Roberto Navigli and Simone Paolo Ponzetto. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250, 2012.

Graham Neubig and Junjie Hu. Rapid adaptation of neural machine translation to new languages. *CoRR*, abs/1808.04189, 2018. URL `http://arxiv.org/abs/1808.04189`.

Toan Q. Nguyen and David Chiang. Transfer learning across low-resource, related languages for neural machine translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 296–301, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. URL `https://www.aclweb.org/anthology/I17-2050`.

Joakim Nivre, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Eckhard Bick, Cristina Bosco, Gosse Bouma, Sam Bowman, Aljoscha Burchardt, Marie Candito, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Silvie Cinková, Çağrı Çöltekin, Miriam Connor, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Marhaba Eli, Ali Elkahky, Tomaž Erjavec, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Petter Hohle, Radu Ion, Elena Irimia, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, Phuong Lê Hồng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shunsuke Mori, Bohdan Moskalevskyi, Kadri Muischnek, Nina Mustafina, Kaili Müürisep, Pinkey Nainwani, Anna Nedoluzhko, Luong Nguyễn Thị, Huyền Nguyễn Thị Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Petya

Osenova, Lilja Øvrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Martin Popel, Lauma Pretkalniņa, Prokopis Prokopidis, Tiina Puolakainen, Sampo Pyysalo, Alexandre Rademaker, Livy Real, Siva Reddy, Georg Rehm, Larissa Rinaldi, Laura Rituma, Rudolf Rosa, Davide Rovati, Shadi Saleh, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Lena Shakurova, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Antonio Stella, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Takaaki Tanaka, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Gertjan van Noord, Viktor Varga, Veronika Vincze, Jonathan North Washington, Zhuoran Yu, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. Universal dependencies 2.0 – CoNLL 2017 shared task development and test data, 2017. URL `http://hdl.handle.net/11234/1-2184`. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Daniel W Otter, Julian R Medina, and Jugal K Kalita. A survey of the usages of deep learning in natural language processing. *arXiv preprint arXiv:1807.10854*, 2018.

Simone Papandrea, Alessandro Raganato, and Claudio Delli Bovi. Supwsd: A flexible toolkit for supervised word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 103–108, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *NAACL 2018*, 2018.

Barbara Plank and Zeljko Agic. Distant supervision from disparate sources for low-resource part-of-speech tagging. *CoRR*, abs/1808.09733, 2018. URL `http://arxiv.org/abs/1808.09733`.

Edoardo Maria Ponti, Roi Reichart, Anna Korhonen, and Ivan Vulić. Isomorphic transfer of syntactic structures in cross-lingual NLP. In *Proceedings of the 56th Annual Meeting*

of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1531–1542, Melbourne, Australia, July 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P18-1142.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul 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.

Vikas Raunak. Simple and effective dimensionality reduction for word embeddings. In *"Learning with Limited Labeled Data: Weak Supervision and Beyond" workshop at the Conference on Neural Information Processing Systems*, 2017.

Siva Reddy, Danqi Chen, and Christopher D Manning. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7: 249–266, 2019.

Shruti Rijhwani, Jiateng Xie, Graham Neubig, and Jaime G. Carbonell. Zero-shot neural transfer for cross-lingual entity linking. *CoRR*, abs/1811.04154, 2018. URL http://arxiv.org/abs/1811.04154.

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

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. A survey of cross-lingual word embedding models. *arXiv preprint arXiv:1706.04902*, 2017.

David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976, 2017.

Yang Shao. Hcti at semeval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 130–133, 2017.

Charles Spearman. The proof and measurement of association between two things. *American journal of Psychology*, 15(1):72–101, 1904.

Robyn Speer and Joanna Lowry-Duda. Conceptnet at semeval-2017 task 2: Extending word embeddings with multilingual relational knowledge. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 85–89, 2017.

Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Fei Sun, Jiafeng Guo, Yanyan Lan, Jun Xu, and Xueqi Cheng. Learning word representations by jointly modeling syntagmatic and paradigmatic relations. 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)*, pages 136–145. Association for Computational Linguistics, 2015. URL `http://aclweb.org/anthology/P15-1014`.

Greg Corrado Tomas Mikolov, Kai Chen and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781v3*, 2013.

Peter D Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *European conference on machine learning*, pages 491–502. Springer, 2001.

Peter D Turney. The latent relation mapping engine: Algorithm and experiments. *Journal of Artificial Intelligence Research*, 33:615–655, 2008.

Peter D Turney, Michael L Littman, Jeffrey Bigham, and Victor Shnayder. Combining independent modules to solve multiple-choice synonym and analogy problems. *arXiv preprint cs/0309035*, 2003.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. Cross-lingual models of word embeddings: An empirical comparison. *CoRR*, abs/1604.00425, 2016. URL `http://arxiv.org/abs/1604.00425`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. Sense vocabulary compression through the semantic knowledge of wordnet for neural word sense disambiguation. *CoRR*, abs/1905.05677, 2019. URL `http://arxiv.org/abs/1905.05677`.

Ivan Vulić and Anna Korhonen. On the role of seed lexicons in learning bilingual word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 247–257, 2016.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198, 2017.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics*, 3:345–358, 2015.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime G. Carbonell. Neural cross-lingual named entity recognition with minimal resources. *CoRR*, abs/1808.09861, 2018. URL `http://arxiv.org/abs/1808.09861`.

Chao Xing, Dong Wang, Chao Liu, and Yiye 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.

Dongqiang Yang and David M. W. Powers. Measuring semantic similarity in the taxonomy of wordnet. In *Proceedings of the Twenty-eighth Australasian Conference on Computer Science - Volume 38*, ACSC '05, pages 315–322, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc. ISBN 1-920-68220-1. URL `http://dl.acm.org/citation.cfm?id=1082161.1082196`.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019. URL `http://arxiv.org/abs/1906.08237`.

Wenpeng Yin and Hinrich Schütze. Learning word meta-embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1351–1360, Berlin, Germany, August 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P16-1128`.

Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. A compare-aggregate model with latent clustering for answer selection. *CoRR*, abs/1905.12897, 2019. URL `http://arxiv.org/abs/1905.12897`.

Yuan Zhang, David Gaddy, Regina Barzilay, and Tommi Jaakkola. Ten pairs to tag – multilingual POS tagging via coarse mapping between embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1307–1317, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1156. URL `https://www.aclweb.org/anthology/N16-1156`.

Jie Zhou and Wei Xu. End-to-end learning of semantic role labeling using recurrent neural networks. 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)*, pages 1127–1137, 2015.