Cross-lingual Word Embeddings Beyond Offline Mapping

Aitor Ormazabal Oregi

Master of Science
School of Informatics
University of Edinburgh
2020

Abstract

Cross-lingual Word Embeddings are a way of representing words of two languages as points in a shared semantic vector space, that is, a vector space where distances and geometric relations between points are semantically meaningful. These cross-lingual embeddings enable tasks such as unsupervised machine translation and cross-lingual transfer learning. Unsupervised methods that learn these embeddings without any cross-lingual supervision have attracted a lot of interest in recent years, since they can in theory work in low-resource settings, where cross-lingual transfer is specially interesting.

Recent research on unsupervised cross-lingual word embedding methods has been dominated by mapping approaches, which first learn the embeddings for each language independently, and then align them into a shared space through a linear transformation. These methods work well under ideal conditions, but their performance depends heavily on the monolingual embeddings having a similar structure. However, recent work has shown that this assumption doesn't hold when learning the embeddings separately for each language, hindering the performance of mapping methods.

In this thesis, we design and implement a novel method to address this problem. Instead of aligning two fixed embedding spaces, we learn the embeddings in two steps: first we learn the representation for a target language, and then we train the source language embeddings in such a way that they are aligned with the target space, which is kept fixed. By taking this approach, our method learns the representations directly in a shared space, thus sidestepping the need for mapping.

Our experiments on bilingual lexicon induction over six language pairs confirm the effectiveness of our method, surpassing the state-of-the-art baseline on every pair, with an average improvement of over 2 percentage points.

Acknowledgements

I would like to thank my supervisors, Ivan Titov, Mikel Artetxe and Eneko Agirre, for their valuable feedback and contributions throughout this project.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Aitor Ormazabal Oregi*)

Table of Contents

Chapter 1

Introduction

In the past decade, deep learning has transformed the field of natural language processing. This paradigm achieves high-level abstraction by stacking multiple layers of processing together, usually in the form of neural networks. Thanks to it, the field has seen unprecedented advances in areas such as translation, generation and understanding [Edunov et al., 2018, Radford et al., 2018, Devlin et al., 2019].

An area where deep learning has been particularly successful is unsupervised pre-training. When pre-training, we learn vector representations for language units, such as words or sentences, by training a model to perform a certain self-learning task, such as language modelling, over large corpora. The idea is that, since these models are trained on very general linguistic tasks over large amounts of text, they learn general patterns of language, and that this knowledge can be exploited by fine-tuning the model for many different NLP tasks. Since the amount of labeled data for most NLP tasks is limited, starting from a pre-trained representation usually yields much better results than training a model from scratch [Devlin et al., 2019], and fine-tuning large pre-trained models such as BERT has become common practice in the field.

Pre-trained representations can be broadly classified into two categories: static, where a fixed representation is learned for each unit, or dynamic, where the vector representation for a given unit depends not only on the unit itself, but on its context too. A widely known example of pre-trained static representations are word embedding algorithms such as Skip-gram, proposed by Mikolov et al. [2013b], with an accompanying implementation called *Word2Vec* [1]. These algorithms train a shallow neural network on a simple language modelling task, and extract vector representations for each word from the trained weights. These embeddings are widely used in the literature as an

[1] `https://github.com/tmikolov/word2vec`

alternative to randomly initializing the weights for the initial layer of a neural network in NLP taks, and are specially helpful when training data is scarce [Kim, 2014, Chen and Manning, 2014].

The success of static word embeddings has also spawned a lot of research on cross-lingual word embeddings, as they can be used for cross-lingual transfer. Cross-lingual word embeddings (CLWE) represent units of multiple languages as vectors in a shared semantic space. By semantic space we mean that geometric relationships in the vector space will be semantically significant not only within each language, but across languages. For example, words in both languages that have similar meaning should be close together in the vector space. If we have such multilingual representations available, we can train a model for a certain task in a high-resource language, for which we have labeled data available, and then simply switch the embeddings for this language with aligned embeddings for another language. Then, due to the multilingual nature of the representations, the model should be able to perform well on this task in the new language, for which we had no labeled training data. That is, we can ”transfer” what the model has learned in one language to another language. This can help make many NLP tools and models that are currently only available in English accessible to millions of minority language speakers.

Even though contextualized models usually show the best results for cross-lingual transfer learning, static CLWE are also used in areas such as unsupervised machine translation [Artetxe et al., 2018b], and are an active area of research [Wang et al., 2019, Patra et al., 2019, Ormazabal et al., 2019].

In recent years, unsupervised cross-lingual word embedding algorithms have attracted a lot of attention. In unsupervised embedding learning, aligned representations are learned for two or more languages using monolingual resources only (i.e. non-related monolingual corpora), without any cross-lingual signal. These algorithms are particularly promising in the context of transfer learning, as low-resource languages, for which cross-lingual transfer learning would be specially useful, rarely have cross-lingual training resources available. In the static scenario, recent research has been mostly focused on mapping based algorithms. These methods first learn monolingual embeddings for each language using regular monolingual algorithms, and they then align them into a shared space through a linear transformation.

These unsupervised mapping methods have achieved very promising results, comparable to those of supervised methods that use fairly large bilingual dictionaries to learn the embeddings [Artetxe et al., 2018a], and have been key components of advances

such as unsupervised machine translation. However, they have some critical drawbacks. The linear transformation used to align the representations is usually learned by exploiting geometric similarities between the monolingual embeddings, and thus these embeddings have to be similar enough in structure for this to be possible. The assumption that independently trained monolingual embeddings will be similar in structure is known as the isometry hypothesis [Miceli Barone, 2016], and it has been shown to break under unfavorable conditions. Søgaard et al. [2018] argue that using different domain corpora or linguistically distant languages can lead to strong divergences in the independently learned embeddings, which in turn causes mapping methods to fail. In previous work [Ormazabal et al., 2019], we showed that this divergence in structure is a particular limitation of mapping methods, as it isn't shared by another class of supervised "joint" methods that learn representations for both languages simultaneously. However, these joint methods require very strong cross-lingual supervision in the form of parallel corpora [Luong et al., 2015]. This suggests that unsupervised methods that can learn representations directly in a shared space are worth exploring, as they could help sidestep the issues of mapping methods without requiring large amounts of supervision. This is precisely the goal of this thesis.

In this work we will introduce a novel approach to learn CLWE. Instead of learning the representations for each language independently as in mapping methods, we instead learn them in two steps: first we learn the embeddings for one of the languages, which we refer to as the "target" language, using monolingual algorithms, and then we learn the representations for the other language, called the "source" language, in such a way that they are aligned with the target embeddings, which are kept fixed. The alignment with the target embeddings is achieved through two components: a reassignment step, where target embedding vectors are assigned to source language words according to a given bilingual dictionary, and a refinement step, where the reassigned source vectors are fine-tuned through regular Skip-gram, while keeping representations for reliable translations frozen. These additions allow us to learn the embeddings directly in a shared space, without need for a mapping step. However, a mapping method will still be used at first to obtain the initial dictionary used for reassignment. In order to remove this dependency, we will introduce several improvements to this core method that will allow us to use very poor dictionaries for the initialization, such as dictionaries based on identically spelled words or numerals in both languages, thus removing the need for a mapping method to obtain the initial dictionary.

Empirically, our method outperforms the previous mapping based state-of-the-art

by a significant margin, obtaining the best results for every language pair. It thus serves as a plug-in replacement for systems that currently use cross-lingual word embeddings, such as unsuperivsed machine translation frameworks, and as foundation for a new way of learning cross-lingual embeddings in an unsupervised way.

1.1 Outline

This report is structured as follows:

- In chapter 2, we introduce previous work on CLWE necessary to understand our method: monolingual word embedding algorithms, currently dominant cross-lingual embedding approaches, and evaluation tasks for CLWE. Additionally, we cover related work that also attempts to address issues of current methods.
- In chapter 3, we introduce our novel approach to learn CLWE. The key idea will be to learn embeddings in two steps: first the representations for the target language are learned using regular monolingual Skip-gram, and then the source embeddings are learned, while retaining alignment with the target space. We will describe two additions to the Skip-gram algorithm that make this alignment possible: reassignment and refinement with context freezing. We will also design an experimentation process to analyze the performance of our method as compared to the state-of-the-art, and discuss the results.
- In chapter 4, we identify several issues with the initial method presented in chapter 3, the main one being the dependency on a mapping method, wich is used to obtain the initial dictionary for the reassignment step. In order to alleviate this issue, we introduce two improvements to the method: iterative re-induction and random restarts. Through these additions, our method is able to achieve good results even when initialized with very poor heuristic based dictionaries, and therefore a mapping method is no longer needed for the initialization. We also experiment with the final method to analyze its performance as compared to the literature and the previous method.
- Finally, in chapter 5, we present the main conclusions drawn from our results, and we discuss possible future lines of research.

The contents of chapter 3 are currently under review in the form of a short paper in the EMNLP 2020 conference.

Chapter 2

Background

We will first describe the Skip-gram algorithm used to train monolingual embeddings, followed by a brief exposition of the current state-of-the-art in unsupervised mapping methods. Finally, we will outline the BLI metric used throughout the thesis to evaluate CLWE.

2.1 Monolingual word embeddings and the Skip-gram algorithm

Monolingual word embeddings are an example of static pre-trained word representations. They represent each word in a language by a point in a semantic vector space (i.e. the relations between vector representations are semantically significant). Given a lexicon from some language of size V, we can represent the embeddings for that lexicon as a $V \times N$ matrix X, where N is the dimension of the vector space, and X_{i*} represents the vector for the ith word in the lexicon. We will use this notation throughout this section.

Work on learning dense representations has a long history, and there exist many algorithms to learn monolingual word embeddings [Mikolov et al., 2013b, Bojanowski et al., 2017, Pennington et al., 2014]. We will focus on the Skip-gram with Negative Sampling (referred to as SGNS throughout this thesis) algorithm proposed by Mikolov et al. [2013b], which is widely used in the literature. We refer the reader to the student's Informatics Research Review [Aitor Ormazabal, 2020] for a deeper survey of the topic.

Most embedding learning algorithms such as Skip-gram fall within the general distributional semantics framework. The core idea behind distributional semantics is the distributional hypothesis, which broadly states that one can learn properties about

linguistic units based on their distributional properties in a large corpus, commonly expressed through the quote "You shall know a word by the company it keeps" [Firth, 1957].

The basic idea is to train a shallow neural network to perform a certain task on large amounts of monolingual text, so that the learned weights from the network can be used as vector representations. The network consists of two $V \times N$ dimensional matrices, X and C. We call X the word matrix and C the context matrix, and X_{i*} and C_{i*} denote the word and context vectors for the ith word. Then, a corpus $C = \{w_1, ..., w_K\}$ of length (token count) K is used to train the weights in these matrices. Specifically, the following training objective is minimized:

$$H = -\sum_{n=1}^{K} \sum_{-c \leq j \leq c, j \neq 0} \log P(w_{n+j}|w_n).$$

That is, for each word w_n in the training corpus, we look at the surrounding words in a context window of size c, and use the word w_n to try and predict the context words w_{n+j}. In regular skip-gram, the word and context vectors are used to estimate $P(w_{n+j}|w_n)$ in the neural network through a softmax:

$$P(w_{n+j}|w_n) = \frac{exp\{C_{w_{n+j}*} \cdot X_{w_n*}\}}{\sum_{i=1}^{V} exp\{C_{i*} \cdot X_{w_n*}\}}.$$

However, computing the softmax denominator over all words in the vocabulary can be computationally expensive. For that reason, the negative sampling loss replaces the expensive softmax evaluation with the following loss:

$$P(w_{n+j}|w_n) = \log \sigma(C_{w_{n+j}*} \cdot X_{w_n*}) + \sum_{i=1, w_i \sim P(w)}^{k} \log \sigma(-C_{w_i*} \cdot X_{w_n*}).$$

Intuitively, this means that the similarity (as measured by the dot product) between the word w_n and the true context should be high, while the similarity between the word and a random context w_i sampled from the distribution $P(w)$ should be low. These sampled k words are called the negative samples. This negative sampling loss is much quicker to compute, and was shown by Mikolov et al. [2013b] to achieve very good embedding quality. The distribution $P(w)$ from which the negatives are sampled is a free parameter of the model, and is usually a modified version of the unigram distribution, so that more frequent words are sampled with higher frequency.

Once the objective H is minimized through stochastic gradient descent, the word matrix X is used as the word embedding. One could also choose to use C or some

combination of X and C, but using only the word vectors is a common choice in the cross-lingual embedding literature [Conneau et al., 2018, Artetxe et al., 2017, Ormazabal et al., 2019], which we have followed in this thesis.

2.2 Cross-lingual word embedding methods

Just like monolingual word embeddings represent each word in a language as a point in a semantic vector space, cross-lingual word embeddings represent words of two languages in a shared semantic space. By shared space we mean that the geometric relations between word vectors will be semantically significant not only within each language, but across languages. For example, given a certain word in one language, we would expect the closest word in the other language to be its translation.

In this section we will describe the general mapping framework that most unsupervised methods follow, and then we will briefly describe the alternative paradigm of joint training. Just as we used a $V \times N$ matrix X to represent the embeddings for one language in the previous section, here we suppose that we are working with two languages, L_1 and L_2, and represent the embeddings for them by X and Z, respectively.

2.2.1 Mapping methods

In mapping based methods, the embeddings X and Z for each language are learned independently on monolingual corpora, using algorithms such as Skip-gram. Then, in the mapping step a $N \times N$ matrix W is learned, such that XW and Z are aligned in a shared space.

In supervised mapping, a dictionary of source-target word pairs is usually used to learn the transformation: W is chosen so that the distance between the translation pairs in XW and Z according to some metric is minimized. This idea to align embedding spaces through a linear transformation was introduced by Mikolov et al. [2013a], who proposed to minimize the sum of square distances between translation pairs:

$$W^* = \arg\min_W \sum_{(i,j)\in H} ||X_{i*}W - Z_{j*}||^2,$$

where H is a dictionary composed of (i, j) pairs, and $i \in L_1$ and $j \in L_2$ are translations of each other.

However, in unsupervised methods there is no such dictionary available, and therefore W has to be learned through some unsupervised heuristic. Thus the main difference

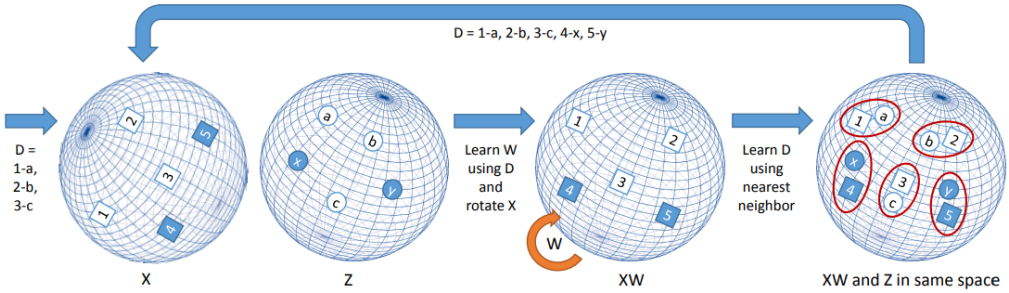


Figure 2.1: The self-learning step in the unsupervised mapping algorithm illustrated. Image from Artetxe et al. [2017].

between mapping methods is how this transformation is learned. We will outline the method of [Artetxe et al., 2018a], as we use it as a baseline in this project, and we will then briefly describe several alternatives proposed in the literature.

To learn a mapping W without any cross-lingual signal, they propose a heuristic based on the monolingual similarity distributions of each word. The basic idea is as follows: if the embeddings X and Z for L_1 and L_2 were perfectly isometric (i.e. there is a one-to-one mapping of words from L_1 to L_2 and the corresponding word vectors differ only by a rotation), then the similarity matrices XX^T and ZZ^T would be the same up to a permutation of its rows and columns, where the permutation would be given by the dictionary between L_1 and L_2. In practice, the embeddings won't be perfectly isometric, but we could still find an initial dictionary by finding the permutation that makes XX^T and ZZ^T match the best. Since checking all permutations is computationally infeasible, they solve an approximate version of this problem to build an initial dictionary. Although this initial dictionary is very weak (achieving only 0.53% accuracy according to the authors), it is enough to bootstrap a self-learning process.

After an initial dictionary is obtained, they propose a self-learning step that iteratively improves the quality of the resulting mapping. The following two steps are alternated until convergence:

1. Computing the optimal mapping based on the current dictionary. Given a dictionary in the form of a binary matrix D, where $D_{ij} = 1$ if and only if $j \in L_2$ is a translation of $i \in L_1$, they find matrices W_X and W_Z such that XW_X and ZW_Z are aligned according to the dictionary. Specifically, they optimize

$$\underset{W_X\, W_Z\, orthogonal}{\arg\max} \sum_i \sum_j D_{ij} S((X_{i*}W_X), (Z_{j*}W_Z)),$$

where S is a similarity metric between vectors, so that the sum of similarities between translation pairs is maximized[1]. For the similarity metric, they use the cosine similarity described in Section 2.3, which intuitively measures the cosine of the angle between the vectors, discarding the length.

2. Computing a new dictionary based on the current mapping. In this step the current aligned embeddings XW_X and ZW_Z are used to induce a new dictionary D, such that $D_{ij} = 1$ if $j = \arg\max_k S((X_{i*}W_X), (Z_{k*}W_Z))$, and $D_{ij} = 0$ otherwise. Here S is a similarity metric, so that the closest word in the opposite language is chosen as the translation for each $i \in L_1$. For the similarity metric, they use the CSLS function, which we describe in detail in the evaluation Section 2.3, as it yields better results than standard cosine similarity.

The idea behind the self-learning process is that the embeddings resulting from a certain dictionary D can be used to induce new dictionary D' that is of better quality than D, and that this process can be iterated to obtain good quality cross-lingual embeddings even when the initial dictionary is very poor. This process is illustrated in figure 2.1.

The authors also proposed several improvements to the self-learning step to improve its robustness and quality, such as randomly setting some values of the dictionary D to zero, and inducing the dictionary only for the most frequent words in the vocabulary. Their resulting method achieved very strong results, comparable to the supervised state-of-the-art [Artetxe et al., 2018a, Glavaš et al., 2019].

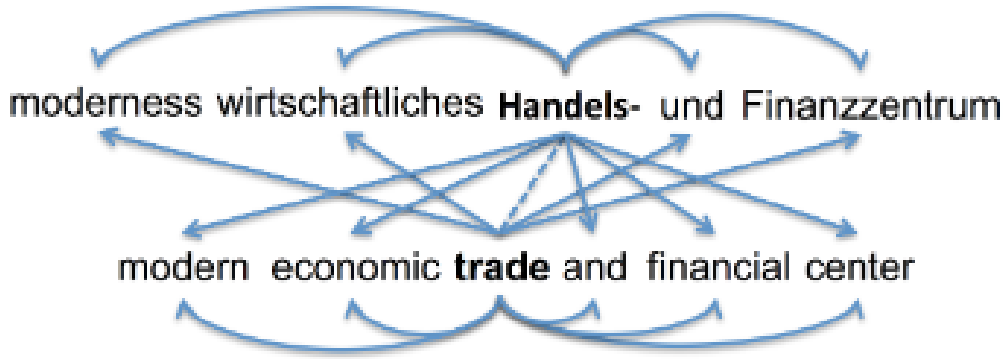
Multiple other approaches to learn an initial transformation W without any supervision have been proposed: Conneau et al. [2018] use a Generative Adversarial Network approach, and Hoshen and Wolf [2018] use an adapted version of the Iterative Closest Point (ICP) algorithm from the 3D point cloud literature. Both of these methods use a refinement step similar to the self-learning we just described to improve the initial mapping, although Conneau et al. [2018] report good results using only a single iteration.

2.2.2 Joint training

Although the previously described unsupervised mapping methods achieve very good results under adequate conditions, they often fail when the monolingual embeddings X

[1]Note that finding W_X and W_Z such that XW_X and ZW_Z are aligned is conceptually similar to finding W such that XW and Z are aligned. In fact, when W_X and W_Z are orthogonal, one can take $W = W_X W_Z^T$, and the distances and dot products between word vectors will be the same in both configurations.

(a) Monolingual Skip-gram



(b) Extended cross-lingual form of Skip-gram used in BiVec

Figure 2.2: Difference in context prediction between monolingual Skip-gram and BiVec. See text for details. Original image from Luong et al. [2015]

and Z aren't similar enough, since the heuristic methods fail to obtain a good enough initial transformation W [Søgaard et al., 2018, Artetxe et al., 2018a].

An alternative to mapping methods is to do "joint" training, where the embeddings for both languages X and Z are learned simultaneously, and directly in a shared space. One example of this is the BiVec algorithm introduced by Luong et al. [2015], which is an extension of the monolingual Skip-gram algorithm presented in Section 2.1. BiVec is trained on parallel corpora, that is, a collection of pairs of sentences that have the exact same content in two languages, and are aligned at the word level. When going through a word w in the corpus, instead of only using w to predict its context in the same language as in regular Skip-gram, BiVec also uses w to predict the context of the corresponding word in the opposite language. This cross-lingual prediction term in the loss provides an incentive for word and context vectors for both languages to be learned aligned in a shared space. Figure 2.2 illustrates the difference between regular Skip-gram and BiVec. In the monolingual case, the word *trade* is used to predict its context. However, in the cross-lingual case, the word *trade* is aligned with the German word *Handels* in the parallel corpus, and thus the word *trade* will also be used to predict the context of *Handels* in the German side of the corpus.

As mentioned in the introduction, in Ormazabal et al. [2019] we showed that this joint training method doesn't suffer from the structural mismatch issue that often causes mapping methods to yield poor results. However, large parallel corpora usually don't exist for low-resource language pairs, and are very expensive to obtain. This motivates our goal to develop a method capable of learning representations directly in a shared space, without requiring large amounts of supervision.

2.3 Evaluation of CLWE

There are many ways to evaluate cross-lingual word embeddings, but due to time constraints we have focused on the Bilingual Lexicon Induction (BLI) task for this project, which is the most widely used evaluation task in the literature. We refer the reader to the student's Informatics Research Review [Aitor Ormazabal, 2020] for a deeper review of other evaluation tasks.

Suppose we have two languages, L_1 and L_2, and cross-lingual embeddings X and Z for each language. Then, we first define a translation function from L_1 to L_2 using the embeddings:

$$\text{translation}(i) = \arg\max_{j \in L_2} S(X_{i*}, Z_{j*}),$$

where S is a similarity metric that measures how close together two vectors are in the vector space. That is, given a word from L_1, the translation function simply chooses the closest word from L_2 as its translation. One common metric S used in the literature is the cosine similarity, which intuitively measures the cosine of the angle between two vectors, discarding their lengths:

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \, \|\mathbf{y}\|}.$$

However, when using cosine similarity to retrieve the translation, word embeddings are known to suffer from the *hubness problem* [Radovanović et al., 2010]. The hubness problem is a phenomenon that causes a few points in high-dimensional point clouds to be the nearest neighbors of many other points, and it is known to affect CLWE [Lazaridou et al., 2015]. In the setting of BLI, this means that a few words from L_2 will be chosen as the translations of many L_1 words, which has a negative impact on the induced dictionary's quality. Conneau et al. [2018] proposed to use an alternative similarity metric, called CSLS, that is designed to reduce the hubness issue. It is given by the following formula:

$$\text{CSLS}(\mathbf{x},\mathbf{y}) = 2\cos(\mathbf{x},\mathbf{y}) - \frac{1}{k}\sum_{\mathbf{y}'\in\mathcal{N}_Z(\mathbf{x})}\cos(\mathbf{x},\mathbf{y}') - \frac{1}{k}\sum_{\mathbf{x}'\in\mathcal{N}_X(\mathbf{y})}\cos(\mathbf{x}',\mathbf{y}),$$

assuming that $\mathbf{x}$ comes from the L_1 embedding X and $\mathbf{y}$ from the L_2 embedding Z. The symbol $\mathcal{N}_Z(\mathbf{x})$ denotes the set of k closest neighbors to the vector $\mathbf{x}$ in the embedding Z (that is, among the word vectors for the L_2 language). Similarly, $\mathcal{N}_X(\mathbf{y})$ is the set of closest neighbors to the vector $\mathbf{y}$ among the word vectors for the L_1 language. Intuitively, the CSLS metric will penalize word vectors y for which the k closest neighbors are very close, since these words would have a high chance of being hubs. Empirically, this similarity function yields very good results and is currently the most commonly used retrieval metric in the literature [Joulin et al., 2018, Artetxe et al., 2018a, Hoshen and Wolf, 2018].

Once the translation function is defined, it is used to induce a dictionary, and this dictionary is compared to a gold standard to obtain the final evaluation metric. Specifically, we measure the precision at one (referred to as P@1 throughout this thesis): the percentage of L_1 words in the gold standard dictionary for which our model generates a correct translation.

There are multiple sets of gold standard dictionaries freely available. In this thesis we use the MUSE set of dictionaries [2], as it is widely used in the literature, which allows us to compare directly to other works. The collection was created by Conneau et al. [2018] using internal tools, and contains 1500 entries for each test dictionary.

It is worth noting that the evaluation dictionaries are usually one-to-many, as each word can have multiple correct translations. When calculating the precision at one, we count a word as correctly translated if the generated translation is one of the options in the gold standard.

2.4 Related work

We have covered the currently dominant mapping approach to unsupervised CLWE learning in the background section, and have identified several issues with it. In this section, we describe other works that have tried to address these issues by utilizing alternative approaches.

Nakashole [2018] attempt to mitigate the structural mismatch issue of mapping methods by learning neighborhood sensitive maps. They use a model that jointly

[2] https://github.com/facebookresearch/MUSE

discovers neighborhoods in the monolingual spaces, and learns specific mappings for each of them, thus learning a mapping that is not globally linear. However, their approach still depends on aligning independently trained monolingual embeddings.

Additionally, Lample et al. [2018b] found positive results learning word embeddings over concatenated corpora using regular monolingual algorithms. Wang et al. [2019] built upon this method by learning a linear mapping afterwards. However, both of these methods rely on the existence of identically spelled words, as they serve as anchor points during the learning, and thus cannot work on purely unsupervised settings.

Work by Lample and Conneau [2019] has shown promising results by extracting cross-lingual embeddings from the first layer of deep cross-lingual language models, that are jointly trained on multiple language corpora. However, they didn't evaluate these extracted embeddings in common evaluation tasks, making direct comparison difficult.

Chapter 3

Unsupervised Cross-lingual Embeddings Beyond Offline Mapping

In this chapter we will present the first method developed in this thesis, and the experiments carried out to analyze its performance compared to the state-of-the-art, and its properties.

3.1 Main method

We recall that the general mapping paradigm uses the following approach: first word embeddings for both languages are independently learned using monolingual algorithms, and then these embeddings are aligned into a shared space through a linear transformation. As we have seen in the introduction, this approach can be problematic when there is a high degree of mismatch between the monolingual embeddings. Instead, our method's general idea is as follows: we first learn the embeddings for one of the two languages using monolingual methods, and then learn the second embeddings in such a way that they are aligned with the first ones, which are kept fixed.

Specifically, we start with a set of word and context embeddings in the source and target languages—denoted as $W(src)$, $C(src)$, $W(trg)$ and $C(trg)$—trained with Skip-Gram with Negative Sampling over monolingual corpora, as described in Section 2.1. Given these embeddings, our method consists of three steps:

1. A **dictionary induction step**, where a dictionary is induced using an unsupervised mapping method

2. A **reassignment step**, which uses the induced dictionary to reassign target space

vectors to source space words, discarding the original source embeddings.

3. A **refinement step**, where the source language embeddings are retrained using SGNS, from the initial state given by the reassignment step, *while* keeping the context vectors thought to be reliable fixed in order to retain the alignment with the target language.

3.1.1 Dictionary induction

For the dictionary induction, we start by mapping $W(src)$ and $W(trg)$ into a cross-lingual space using any existing mapping method, obtaining $W^{(1)}(src)$ and $W^{(1)}(trg)$. Once mapped, these aligned embeddings are used to induce a source-target dictionary using CSLS retrieval, as described in the embedding evaluation Section 2.3. More concretely, the translation for the ith source word is denoted by T_i, and given by

$$T_i = \arg\max_j \text{CSLS}(W^{(1)}(src)_{i*}, W^{(1)}(trg)_{j*}).$$

3.1.2 Source space reassignment

In this step, we define a new source language embedding matrix $W^{(2)}(src)$ by assigning target word vectors from $W(trg)$ to source words according to the dictionary induced in the previous step. More concretely, the vector for the ith source word is defined as $W^{(2)}(src)_{i*} = W(trg)_{T_i*}$. We also create a context vector matrix $C^{(2)}(src)$ for the source language in a similar manner by translating the context vectors from $C(trg)$, such that $C^{(2)}(src)_{i*} = C(trg)_{T_i*}$.

3.1.3 Source space refinement

The embedding space produced by the previous step does not adequately preserve the structure of the original embeddings in the source language. In fact, the reassigned space has some pathological properties. For example, all the source words that share the same translation will get the same target vector assigned to them. Due to the hubness problem described in Section 2.3, there will be multiple target words that are the translations of many source words, and thus there will be a large number of identical vectors in the source space. It is effectively impossible for this to happen in regular word embeddings trained through algorithms such as Skip-gram, and it is problematic when translating in the target-source direction: there will be many ties when translating, which will have to be broken arbitrarily or through some heuristic.

With the goal of learning a more sensible representation, we run an additional refinement step where we retrain the reassigned source embeddings using SGNS. Specifically, we initialize the word and context matrices with the reassigned state, $W^{(2)}(src)$ and $C^{(2)}(src)$, and perform several iterations of SGNS over the source monolingual corpus as usual.

By initializing the source space with the target embeddings, we want the refinement procedure to produce embeddings that are aligned to the target space. However, there is no explicit term in the SGNS objective that prevents departing from this initialization, and thus simply initializing the vectors to this state might not be enough to achieve good alignment. So as to better retain the alignment to the target space, we freeze the context vectors that are expected to be the most reliable (referred to as **context freezing**). More concretely, we freeze the context vector of a source word i if it satisfies the following cyclic consistency condition:

$$i = \arg\max_k \cos(W^{(1)}(src)_{k*}, W^{(1)}(trg)_{T_i*})$$

The condition is satisfied when the nearest neighbor[1] of T_i (the translation of i in the induced dictionary) is again i. The frozen context embeddings act as anchor points to preserve the alignment with the target language, while the word embeddings and the rest of the context embeddings are free to change to learn a sensible representation.

3.2 Experimental design

In this section we will outline the experiments we designed and carried out in order to compare the performance of our novel method to the current state of the art, and to analyze its properties.

To learn the original monolingual embeddings, we use the SGNS version of the *word2vec* implementation, with the following parameters: 300-dimensional vectors, 10 negative samples, a sub-sampling threshold of 1e-5, and 10 epochs. We train the embeddings on Wikipedia corpora, following common practice in the literature. We explain the process of obtaining and processing these corpora in Appendix A.

For the initial dictionary induction step, we use the unsupervised version of VecMap [2][Artetxe et al., 2018a], as it is an state-of-the-art method and has been shown to perform

[1] We found nearest neighbor retrieval over cosine similarity to work better than CSLS in our preliminary experiments.

[2] `https://github.com/artetxem/vecmap`

well across multiple evaluation tasks [Glavaš et al., 2019]. A description of the method can be found in the background Section 2.2.1 .

For the refinement step we used an extended version of *word2vec* that allows for context freezing and custom embedding initialization. Instead of using a fixed number of iterations for the source language refinement, we choose the number of iterations so that the number of updates is similar to the 10 epochs done for the target language. Specifically, the number of iterations for the source language refinement is set to

$$N = 10\frac{\text{\#trg sents}}{\text{\#src sents}}.$$

Due to the time constraints, and following common practice in the literature, we decided to focus on Bilingual Lexicon Induction (BLI) for evaluation. This allows us to compare directly to other works in the literature. Specifically, we use the CSLS retrieval metric described in Section 2.3 to induce a dictionary, and measure the precision at one against a gold standard. For the gold standard we used the MUSE collection described in Section 2.3. Each test dictionary in the MUSE collection consists of 1500 entries spread into different word frequency bins.

We experiment with six language pairs, covering languages from multiple different families. We use English as the target language and Spanish, German, French, Finnish, Russian and Chinese as the source languages. This list covers most pairs frequently used in the literature, as well as more uncommon ones such as Finnish-English and Chinese-English. In our preliminary experiments we found the method to be sensitive to which language is chosen as the target, and the language with the biggest corpus usually worked best as the target language, which is why we always use English for the target. We hypothesize that this is because a larger corpus normally yields better word representations, and thus it is better to fix the embeddings for the larger corpus language and learn the other ones such that they are aligned to it instead of the other way around.

3.3 Results

In this section we will discuss the results of the experimentation. We start with the main results, followed by an analysis of the freezing strategy and the learning curve of our method. Finally, we put our numbers into context by comparing to those reported in other works.

	de-en		es-en		fr-en		fi-en		ru-en		zh-en		avg.	
	→	←	→	←	→	←	→	←	→	←	→	←	→	←
Initial mapping (baseline)	74.4	76.6	83.5	83.3	82.7	83.0	61.9	45.2	66.1	49.1	45.0	32.2	68.9	61.6
+ reassignment	74.4	44.7	83.5	60.7	82.7	64.2	61.9	26.2	66.1	26.9	45.0	17.9	68.9	40.1
+ refinement	**76.7**	**77.5**	**86.8**	**84.5**	**84.4**	**84.4**	**65.0**	**52.4**	**66.4**	**52.8**	**45.1**	**36.3**	**70.7**	**64.6**

Table 3.1: Bilingual lexicon induction results on the MUSE dataset (P@1).

3.3.1 Main Results

Table 3.1 shows the main results for this chapter.

We see that, in the source-target direction, the embeddings obtained after the reassignment step give the exact same accuracy as the initial mapping, which was to be expected as the initial mapping is used to do the reassignment. However, in the target-source direction we observe a large drop in accuracy. This also makes sense, considering the pathological properties of the reassigned space mentioned in Section 3.1.3. For example, many source words will get assigned the exact same target vector due to the hubness problem, resulting in a tie when translating in the target-source direction, which will be broken arbitrarily resulting in a drop in performance.

However, the refinement step recovers this loss, and the full system achieves the best results in all cases, with an average improvement of 2.4 percentage points over all language pairs and directions. These results show that our strategy of fixing one of the embeddings and learning the other so that it is aligned to it works, outperforming the mapping baseline that was used to initialize the process.

3.3.2 Analysis of the freezing strategy

Our reasoning for freezing only the context vectors in the refinement step was that it would help encourage alignment, while allowing enough flexibility for the word vectors to deviate from the initialization into a good representation. However, there are other freezing strategies we could have chosen. In order to better understand the role of freezing in our method, we also experimented with other refinement variants: freezing both the word and context vectors, freezing only the word vectors, and not freezing at all. The results obtained from these different strategies, in terms of the average P@1, are shown in table 3.2.

We can see that all alternative freezing strategies perform substantially worse, not

	xx-en	en-xx
Initial mapping (baseline)	68.9	61.6
+ proposed method	**70.7**	**64.6**
w/ no freezing	66.3	61.0
w/ word freezing	68.3	58.9
w/ context & word freezing	69.1	59.5

Table 3.2: Results in bilingual lexicon induction (avg P@1) with different freezing strategies.

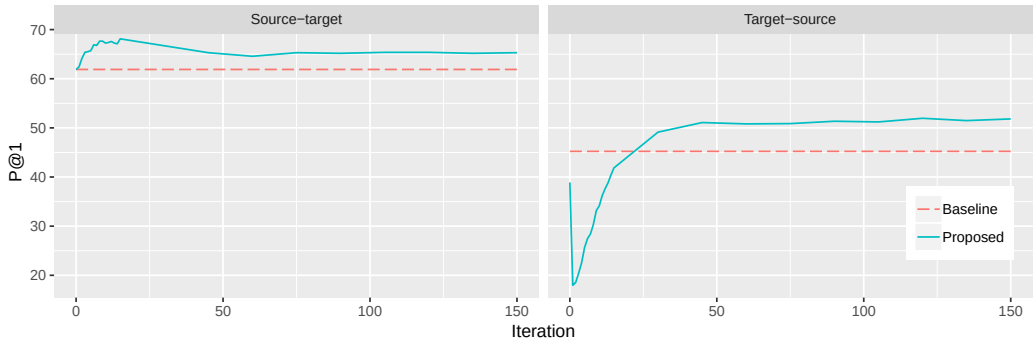


Figure 3.1: Finnish-English learning curves. Iteration 0 corresponds to the reassigned space.

even reaching the baseline, which confirms that our choice of freezing only the context vectors was a sensitive way to retain alignment while allowing flexibility in the word vectors. When no freezing is performed, there is no explicit incentive to keep the representations aligned to the target, leading to inferior results. On the other hand, when word vectors are also frozen, they cannot deviate from the initial state, which as previously mentioned has undesirable properties, also leading to a drop in performance. Context freezing doesn't suffer from either of these problems, and obtains the best results.

3.3.3 Learning curve

The quality of the embeddings for the Finnish-English pair throughout the training iterations, as measured by P@1 on the BLI task in both directions, is shown in figure 3.1.

We can see that each direction exhibits a different pattern. In the source-target

direction, performance starts at the initial mapping level (by design), rises quickly during the first iterations, and decreases slightly after the 15th iteration until it stabilizes well above the baseline, around the 50th iteration.

In contrast, for the target-source direction the initial performance is very low, due to the pathological properties of the reassigned embedding discussed in Section 3.1.3, and gradually increases throughout the training process until it stabilizes around the 50th iteration. We also observe a stark drop in performance from the 0th to the 1st iteration in the target-source direction, which might seem counter intuitive since performance increases steadily with the iterations otherwise. However, this drop can be explained by a particularity of the evaluation process and the reassigned embeddings. In the reassigned embeddings, many source words will have the exact same vectors and thus there will be many ties when translating in the target-source direction. In this situation, our evaluation script picks the word with the highest frequency rank (we didn't explicitly design for this, as the possibility of ties isn't usually a concern for cross-lingual embeddings). However, after the first iteration, the structure of the embeddings won't have changed much but all the source vectors will have been perturbed slightly, which effectively amounts to breaking these ties randomly. Since picking the most frequent candidate is a better heuristic than choosing a candidate at random, the P@1 obtained in the 0th iteration is quite higher. In order to test this theory we also tried to break the ties at random in the evaluation script, and the performance drop between the 0th and 1st iteration indeed disappeared.

Importantly, the BLI performance is stable with respect to the number of iterations and doesn't drop further after stabilizing, which allowed us to set a heuristic for the number of iterations to use with all languages (i.e. the number of iterations needed to have as many updates as in 10 iterations for the target language, as described in Section 3.2), instead of having to tune it as a hyper-parameter, which would be problematic in an unsupervised scenario.

3.3.4 Comparison to the state-of-the-art

In order to put our results into perspective, we also compare them to those reported in the state-of-the-art literature, shown in table 3.3. It is worth noting that these numbers aren't directly comparable, as many works in the literature align pre-trained *fastText* [Bojanowski et al., 2017] embeddings, while we train our own embeddings using *word2vec* and our own extension of it. However, so as to make the comparison as direct

	de-en		es-en		fr-en		ru-en		avg	
	→	←	→	←	→	←	→	←	→	←
Conneau et al. [2018]	72.2	74.0	83.3	81.7	82.1	82.3	59.1	44.0	74.2	70.5
Hoshen and Wolf [2018]	73.0	74.7	84.1	82.1	82.9	82.3	61.8	47.5	75.4	71.6
Grave et al. [2018]	73.3	75.4	84.1	82.8	82.9	82.6	59.1	43.7	74.8	71.1
Alvarez-Melis and Jaakkola [2018]	72.8	71.9	80.4	81.7	78.9	81.3	43.7	45.1	68.9	70.0
Yang et al. [2018]	70.3	71.5	79.3	79.9	78.9	78.4	-	-	-	-
Mukherjee et al. [2018]	-	-	79.2	84.5	-	-	-	-	-	-
Alvarez-Melis et al. [2018]	71.1	73.8	81.8	81.3	81.6	82.9	55.4	41.7	72.5	69.9
Xu et al. [2018]	67.0	69.3	77.8	79.5	75.5	77.9	-	-	-	-
Wang et al. [2019]	72.2	74.2	84.2	81.4	83.6	82.8	58.3	45.0	74.6	70.8
Our method	**76.7**	**77.5**	**86.8**	**84.5**	**84.4**	**84.4**	**66.4**	**52.8**	**78.6**	**74.8**

Table 3.3: Results of the proposed method in comparison to previous work (P@1). All systems are fully unsupervised and use SGNS embeddings trained on Wikipedia.

as possible, we used the same Wikipedia corpora and SGNS hyper-parameters when training the embeddings.

Our approach outperforms all other methods by a substantial margin, including that of Wang et al. [2019], who also try to address the limitations of mapping methods by combining them with joint training over concatenated monolingual corpora.

3.4 Conclusions

In this chapter we have introduced our novel approach to learning cross-lingual embeddings, and analyzed its performance and properties. Despite its simplicity, our approach outperformed all previous state-of-the-art methods, showing that this new paradigm for learning cross-lingual word embeddings beyond offline mapping is worth pursuing. The contents described in this chapter were submitted in the form of a short paper to the EMNLP 2020 conference and are currently under review.

In light of the experimentation and analysis, we identify two issues with our current method:

- It is dependent on a mapping method to obtain the initial dictionary.
- The source space obtained after the reassignment step has some undesirable properties as mentioned in Section 3.1.2, which can make it a non-ideal state to begin the refinement step from.

In the next chapter, we will discuss further improvements to the method that will allow us to reduce the dependency on a mapping step.

Chapter 4

Removing Dependency on Mapping Methods: Iterative Re-induction and Random Restarts

In this chapter we will describe further improvements made to our cross-lingual embedding method, based on the results of the previous chapter. We will also carry out an experimental analysis of the improved method to study its performance and properties.

4.1 Removing dependency on mapping methods

The biggest issue with our current method of Chapter 3 is its dependency on an existing mapping method: the reassignment step requires a good quality bilingual dictionary, and we currently use the unsupervised mapping algorithm VecMap to obtain it. This works well in practice, as our method substantially outperforms the mapping method that is used to initialize it. However, this has some shortcomings; for example, as shown by Søgaard et al. [2018], purely unsupervised mapping methods often completely fail or show very poor results under unfavorable conditions, and thus they cannot be used as an initialization in this setting. Additionally, we consider it interesting for our method to be self-contained, making it an entirely new approach to cross-lingual embedding algorithms, without depending on an external unsupervised algorithm at all.

The improvements made in this chapter revolve around three main ideas, designed to remove the dependency on a separate CLWE method: exploring alternative ways to obtain an initial dictionary that don't rely on a mapping method, iteratively re-inducing the dictionary used for context freezing, and using a random initialization instead of a

reassignment step.

4.1.1 Alternative initial dictionaries

As seen in Chapter 3, our method requires an initial dictionary to decide which target vectors to assign to which source words, and which words to freeze. However, we want to find a way for our method to work without access to an existing CLWE algorithm that can provide this dictionary. One way to remedy this is to obtain the initial dictionary from semi-supervised heuristics that rely on shared words between languages, such as using a dictionary consisting only of numerals (i.e. 1-1, 2-2, ...) or identically spelled words. As seen in works such as Artetxe et al. [2017], these initial dictionaries, although poor, have been enough to bootstrap self-learning methods in the unsupervised mapping scenario. Thus we hypothesize that such an initialization, coupled with the iterative re-induction introduced in the next section, could be enough to obtain good results without relying on mapping methods.

4.1.2 Iterative re-induction

As seen in the results of Chapter 3, our method is initialized by a given dictionary, but once converged the obtained embeddings can induce a dictionary of higher quality than the initial one. This naturally raises the question of whether it is possible to iterate this process. That is, if we can initialize our method with a dictionary D_1, and induce a dictionary D_2 from the final embeddings that is better than D_1, we could run our method again, but using D_2 as the initial dictionary, to obtain potentially better dictionaries D_3, D_4, and so on. This would be akin to the self-learning step using in many works in the unsupervised cross-lingual embedding literature, where an initial poor dictionary is iteratively improved upon by alternating mapping and induction steps [Artetxe et al., 2018a, Conneau et al., 2018]. Additionally, this dictionary re-induction step could be integrated into the learning process, where the current state of the embeddings is used to induce a dictionary, that is then used for the rest of the learning process.

4.1.3 Random initialization

We have seen in Section 3.1.2 that the reassignment step leads to an space with pathological properties in the target-source direction. As seen in the analysis of the learning curve in 3.3.3, this gives the method a very low point to start from in that direction.

Furthermore, the results indicate that context freezing is a key part of our method, since the reassignment initialization alone isn't enough to retain alignment and surpass the baseline. Thus it could be better if we *only* used context freezing, that is, if we only initialized the context vectors that are frozen with those from the target embedding, and used a random initialization for the rest, as in regular SGNS. If context freezing provides a strong enough incentive for alignment, this could help us get rid of some of the pathological properties of the reassignment step, while retaining the quality of our method. Additionally, this could also help deal with poorer initial dictionaries such as the heuristic based ones mentioned in Section 4.1.1, as only the vectors that are necessary for context freezing are initialized according to the dictionary.

4.2 New method

In this section we will describe in detail the improved method, developed around the ideas described in the previous section. We will first re-frame the method of Chapter 3 in a different framework, and then we outline the additions to this framework that compose the method.

The notation will vary slightly in this chapter. In Chapter 3 we assumed that we start with embeddings for both languages, $W(src), C(src), W(trg)$ and $C(trg)$, since they were used to induce the intial dictionary. In this chapter, we more generally suppose that we start only with the target embeddings, $W(trg)$ and $C(trg)$, a source-target dictionary in the form of a translation function T that maps each source word to a target word, and a set C of source words, which we call the freezing set, for which the context vectors will be frozen. This allows us to obtain the initial dictionary T and set C from sources other than an initial unsupervised mapping method. The goal is to learn embeddings $W(src)$ and $C(src)$ for the source language such that the word vectors $W(src)$ are aligned with the target space $W(trg)$.

4.2.1 Re-formulation of context freezing

To train the embeddings $W(src)$ and $C(src)$, we use a modified version of SGNS. As explained in 2.1, whenever a word-context pair (w_1, w_2) is processed in SGNS, the dot product between the context and word vectors $W(src)_{w_1} \cdot C(src)_{w_2}$ is used to calculate their similarity. In our method, if $w_2 \in C$, we use $C(trg)_{T(w_2)}$ instead of $C(src)_{w_2}$ for the context vector. That is, for words contained in C, the context vector from the target

embedding corresponding to its translation is used instead of the source context vector from $C(src)$. Additionally, the target embeddings $C(src)$ aren't trained, and are kept fixed throughout the learning process. Note that if the dictionary T is obtained from an unsupervised mapping method and the freezing set C is chosen using the cyclic consistency freezing criteria described in Section 3.1.3, this method is completely equivalent to the one of Chapter 3, with one exception: only the context vectors for words that are "frozen" (i.e. translated to the fixed target language) are initialized, and the rest retain the random initialization given by the Skip-gram algorithm.

This formulation of our approach is more flexible since it doesn't actually reassign and freeze any context vectors, instead translating them to the fixed target embedding according to the dictionary T and set C, which can be modified on the fly.

4.2.2 Iterative re-induction

This new formulation allows us to naturally incorporate the iterative re-induction idea: every k iterations, we use the current $W(src)$ and $C(trg)$ embeddings to redefine the dictionary T using CSLS retrieval:

$$T_i = \arg\max_j \text{CSLS}(W(src)_{i*}, W(trg)_{j*}),$$

and redefine the freezing set C so that a word $i \in C$ if and only if it satisfies the cyclic condition:

$$i = \arg\max_k \cos(W(src)_{k*}, W(trg)_{T_i*}).$$

Then the training process continues normally using the new T and C for another k iterations, until they are re-induced again.

4.2.3 Initial dictionary

In the previous description we have left the initial dictionary and freezing set as inputs to our method. We now describe three options to obtain these:

- Unsupervised mapping. One option is to train monolingual embeddings for the source target using regular SGNS, and to induce the dictionary T and freezing set C using an unsupervised mapping method and the cyclic consistency property as in Chapter 3.

- Numeral based. For all source words i such that i is a numeral, and i is also in the target vocabulary, we define $T_i = i$ and $i \in C$. For all the rest, we define T_i arbitrarily and $i \notin C$. Note that the translations for words that are not in C are irrelevant, since they are never used.

- Identical word based. For all source words i such that i is also in the target vocabulary, we define $T_i = i$ and $i \in C$. For all the rest, we define T_i arbitrarily and $i \notin C$. This is the same as the numeral based approach, but removing the restriction that i should be a numeral. This will provide a larger dictionary, but "false friends", that is, words that are spelled the same in two languages but have different meanings, could potentially be an issue.

4.2.4 Random restarts

Even though we re-induce the dictionary every k iterations, as the training process goes on the learning rate is reduced and the embeddings converge towards a local optimum, and a change in the dictionary T and freezing set C might not be very significant late in the training process. This can be an issue when the initial dictionary is poor, such as when using the heuristic based dictionary initializations, as the re-induced dictionary late in the training process might be much better than the initial one, but it won't have a big influence. Thus we introduce random restarts: we run the method as described for N iterations, keep the final dictionary T and freezing set C, and then restart the whole training process again, this time using T and C for the initial dictionary and freezing set. This can be repeated multiple times.

4.3 Experimental design

In this section we will describe the experiments we designed and executed to analyze the performance and properties of our new method, compared to the one of Chapter 3 and the literature.

As in Chapter 3, we use SGNS with the following parameters to learn the monolingual embeddings: 300-dimensional vectors, 10 negative samples, a sub-sampling threshold of 1e-5, and 10 epochs. We also use the same Wikipedia corpora.

To obtain the initial dictionary and freezing set, we consider the three options previously described: unsupervised mapping, for which we use the unsupervised mode

in VecMap, a dictionary derived from identically spelled words, and a numeral derived one.

For the refinement step, we again use an extended implementation of *word2vec* that allows for the changes described in Section 4.2. As in Chapter 3, we chose the number of iterations such that the number of updates is similar to the number of updates done in 10 epochs for the target language:

$$N = 10\frac{\text{\#trg sents}}{\text{\#src sents}}.$$

We also use the same BLI evaluation with CSLS retrieval and same language pairs as in Chapter 3: English as the target language, and Spanish, German, French, Finnish, Russian and Chinese as the source languages.

For the iterative re-induction, we again do not set a fixed number, and instead set this parameter so that the re-induction is done after a similar number of updates in every language. Specifically, for the Finnish-English pair we set it to $k = 3$, and for the rest of the languages we set it to

$$k = 3\frac{\text{\#src sents}}{\text{\#Finnish src sents}}.$$

For most language pairs k will be fractional, meaning that the re-induction can happen in the middle of an epoch. We chose to do the re-induction after (approximately) a fixed number of updates instead of a fixed number of iterations, as it aligned better with the similar choice we made for the number of iterations. Since the number of updates is approximately constant across languages, and the re-induction frequency is also set in reference to the number of updates, this also means that the total number of re-inductions done in a run remains constant at 50 (since the number of iterations N for the Finnish-English pair is 150, and we re-induce every 3 iterations).

As for the random restarts, when using the numeral or identical word based initialization we do two random restarts, that is, we run our method with N iterations three times in total, keeping the final dictionary and freezing set from the previous run each time. When using the unsupervised mapping initialization we do no random restarts, since the initial dictionary obtained from the mapping method is good enough to achieve a good solution without them. [1]

[1] Since in an unsupervised setting we have no validation set to tune the re-induction frequency k or the number of restarts, we validated these parameters on the Finnish-English pair, which is usually not used in the literature as seen in the comparison to the state-of-the-art of Section 3.3.4, so we could fairly compare to other works. This is also why the re-induction hyper-parameter k is set in relation to the Finnish corpus size.

	de-en		es-en		fr-en		fi-en		ru-en		zh-en		avg.	
	→	←	→	←	→	←	→	←	→	←	→	←	→	←
Initial mapping (baseline)	74.4	76.6	83.5	83.3	82.7	83.0	61.9	45.2	66.1	49.1	45.0	32.2	68.9	61.6
Reassign + context freezing (Chapter 3)	76.7	77.5	**86.8**	84.5	84.4	84.4	**65.0**	**52.4**	**66.4**	**52.8**	45.1	36.3	70.7	64.6
Full method w/ mapping init	76.6	**78.0**	86.6	84.0	84.9	84.6	**65.0**	52.0	65.6	51.4	**46.1**	**38.6**	**70.8**	**64.8**
Full method w/ identical init	76.4	77.8	86.5	84.1	**85.0**	84.8	63.6	51.4	65.9	51.6	45.0	37.7	70.4	64.6
Full method w/ numeral init	**77.0**	77.6	86.4	**85.0**	**85.0**	**84.9**	63.8	50.7	65.2	51.7	1.4	2.2	63.1	58.7

Table 4.1: Bilingual lexicon induction results on the MUSE dataset (P@1).

4.4 Results

In this section we will discuss the results of the experimentation. We first present the main results, followed by an ablation test on the different components of the new method. We also analyze the learning curve and compare it to that of Chapter 3, followed by an error analysis. Finally, we put our results into context by comparing them to those reported in the literature.

4.4.1 Main results

The main results for this chapter are shown in table 4.1. We observe that the new method performs similarly to our method from Chapter 3, obtaining slightly worse results for some pairs and slightly better ones for other, yielding a slight average P@1 increase of 0.15 percentage points on average for the unsupervised mapping initialization case. However, although the absolute change in accuracy is not significant, the key advancement is that our method can now be initialized with a very poor dictionary and still reach the same level of performance. In fact, the identical word and numeral based initializations have yielded approximately similar results to the mapping intialization across the board, even though they are much weaker. As we will see in the following sections, the iterative re-induction and random restart components play a key role in this.

There is one exception, where the numeral based initialization failed for the Chinese-English pair, achieving less than 3% P@1. We hypothesize that this is due to there being many fewer numerals shared between the vocabularies for the Chinese-English pair compared to the others. To verify this, we compiled the number of shared numerals between the vocabularies for different language pairs, shown in Table 4.2. Indeed, we observe that the number of shared numerals is 244 for the Chinese-English pair, much

	Number of shared numerals
de-en	1360
es-en	1617
fr-en	1573
fi-en	2353
ru-en	1070
zh-en	244

Table 4.2: Number of shared numerals between the vocabularies of different language pairs.

lower than the second lowest value of 1070 for the Russian-English pair.

4.4.2 Ablation test

In this section we will analyze the importance of the two main additions to our method: iterative re-induction and random restarts. To do this, we re-run the experiments with the exact same parameters, but removing each of these components. We first remove the random restarts, and then we remove both the random restarts and the iterative re-induction (we don't consider the case where we remove the re-induction but not the random restarts, since the random restarts depend on the last dictionary given by the iterative re-induction). The results of the ablation test are shown in table 4.3.

We can see that, when removing the random restarts, there is a small dip in performance for the identical word based initialization, and a big dip for the numeral based intialization. That is, when it is initialized with a numeral derived dictionary, the iterative re-induction process gets stuck in a bad solution late in the learning process, and the re-induction steps aren't enough to depart from this state. However, the random restarts help by resetting the learning rate and weights to the initial state, while keeping the best dictionary from the previous run. This can also be seen in the learning curve analysis of the next section.

When removing both the random restarts and the iterative re-induction, we are essentially left with the method from Chapter 3, with one difference: the reassignment step is skipped, so the word and context vectors that aren't frozen are initialized randomly instead of having target embedding vectors assigned to them. We can see that this yields a small dip in performance for the method when it is initialized with

	xx-en	en-xx
Initial mapping (baseline)	68.9	61.6
Reassign + context freezing (Chapter 3)	70.7	64.6
Full method		
Mapping init	**70.8**	**64.8**
Cognate init	70.4	64.6
Numeral init	63.1	58.7
w/ no restarts		
Mapping init	70.8	64.8
Cognate init	69.7	64.5
Numeral init	55.0	52.4
w/ no restarts & no re-induction		
Mapping init	70.1	64.2
Cognate init	54.3	53.3
Numeral init	2.4	1.9

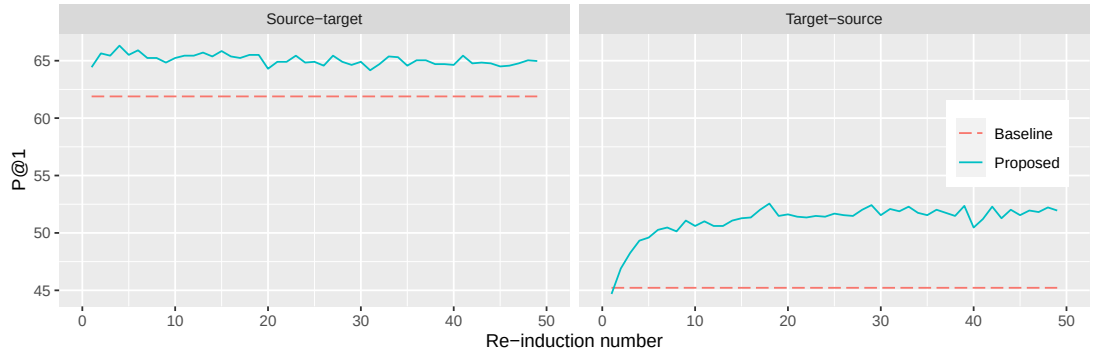
Table 4.3: Results in bilingual lexicon induction (avg P@1) with different freezing strategies.

the dictionary from the unsupervised mapping, and a very large drop for the identical word and numeral based initializations. This was to be expected, as the dictionaries obtained from these heuristics are usually poor, and thus they need to be improved in a self-learning process. This agrees with the pattern seen in the semi-supervised embedding mapping literature [Artetxe et al., 2017].

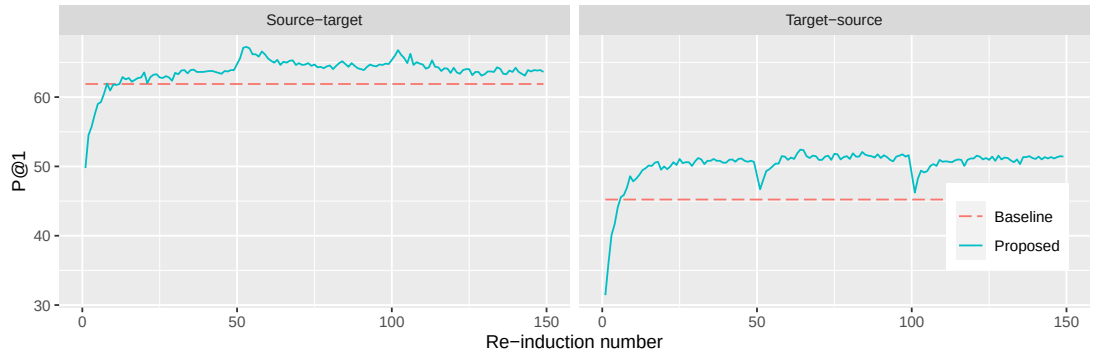
We conclude that random restarts and iterative re-induction are both helpful; the random restarts allow us to use much poorer dictionaries for the initialization step, and iterative re-induction helps recover the small dip caused by the random initialization, surpassing the performance of our previous method without utilizing the reassignment step that led to undesirable properties in the space as seen in Section 3.3.3.

4.4.3 Learning curve

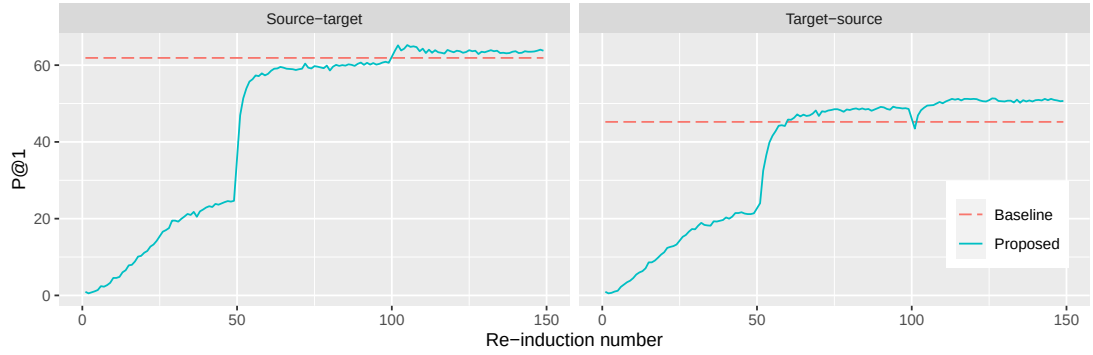
In this section we will analyze the performance of our method throughout the learning process. The BLI score of our method for each iteration for the English-Finnish pair is shown in figure 4.1. For the graphs in this section, the horizontal axis indicates the



(a) Initial dictionary obtained from unsupervised mapping



(b) Initial dictionary obtained from identical words



(c) Initial dictionary obtained from numerals

Figure 4.1: Finnish-English BLI P@1 in both directions for each iteration. Note the difference in vertical axes.

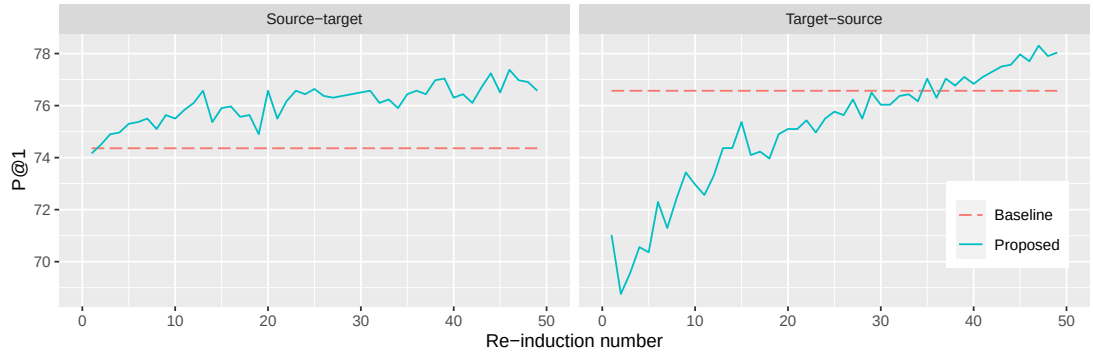
number of dictionary re-inductions done at that point, which as mentioned previously remains constant for every language pair: it is 50 when not doing random restarts, and 150 when doing two random restarts.

We will discuss the learning curve for the unsupervised mapping initialization first. We can see that, unlike in the learning curve of Chapter 3, we don't observe a drop in performance after an early peak in the source-target direction. Instead, both directions improve steadily until they stabilize well above the baseline, although the target-source direction still takes longer to do so. We also see that the performance in the target-source starts much higher and reaches the baseline much quicker than it did in the previous method. The big drop in performance of the previous method was caused by the reassignment step, which created undesirable artifacts when translating in the target-source direction. Thus we can see that removing this step has helped the method achieve good results much quicker in this direction.

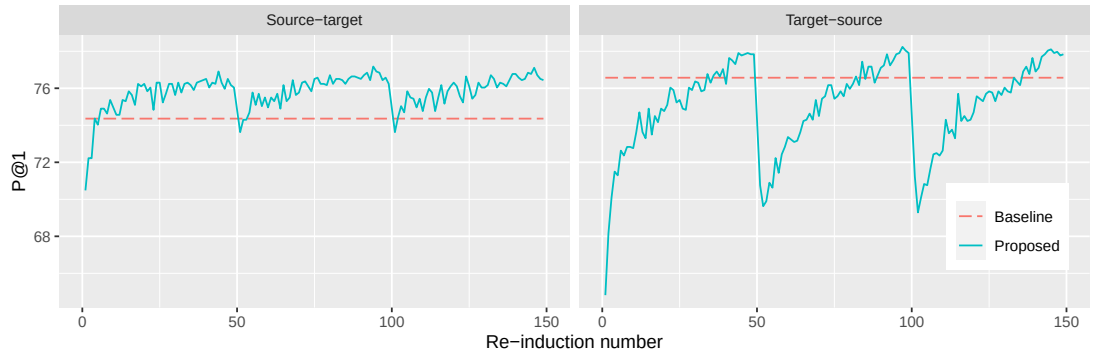
When it comes to the numeral based initialization, we can see that the random restarts play a key role in achieving a good solution, as discussed in the ablation section. For the numeral and identical word based intializations, the random restarts happen at iterations 50 and 100. While the improvements yielded by iterative re-induction slow down considerably as the training process goes on, restarting the learning process and weights while keeping the dictionary yields a big jump in performance, and two random restarts are enough to achieve a solution quality similar to that of the model initialized with a state-of-the-art mapping method.

For the identical word based intialization, we observe that even before the first restart the method achieves a much better solution than for the numeral based dictionary, which makes sense given that this dictionary is much bigger. However, the random restarts still help improve the embedding quality slightly.

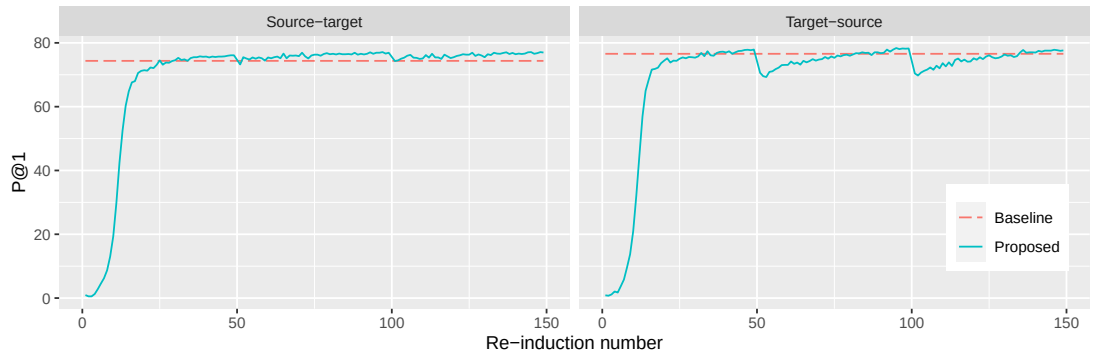
It is also worth noting that these patterns can be different for different language pairs. Figure 4.2 shows the learning curve for the German-English pair. In this case we observe that even for the numeral based initialization, the method is able to achieve a good solution even before the first random restart. For the numeral and identical word based initializations, after each random restart we observe a dip in BLI performance (which makes sense, since all the weights are re-initialized randomly), and then this gap is recovered throughout the SGNS training process until a quality similar to that of before the restart is achieved. Thus we observe that, even though the random restarts don't hurt the solution quality, in some cases they might be superfluous, and could be skipped to make the method run faster. However, since in an unsupervised scenario we



(a) Initial dictionary obtained from unsupervised mapping



(b) Initial dictionary obtained from identical words



(c) Initial dictionary obtained from numerals

Figure 4.2: German-English BLI P@1 in both directions for each iteration. Note the difference in vertical axes.

	de-en		es-en		fr-en		ru-en		avg	
	→	←	→	←	→	←	→	←	→	←
Conneau et al. [2018]	72.2	74.0	83.3	81.7	82.1	82.3	59.1	44.0	74.2	70.5
Hoshen and Wolf [2018]	73.0	74.7	84.1	82.1	82.9	82.3	61.8	47.5	75.4	71.6
Grave et al. [2018]	73.3	75.4	84.1	82.8	82.9	82.6	59.1	43.7	74.8	71.1
Alvarez-Melis and Jaakkola [2018]	72.8	71.9	80.4	81.7	78.9	81.3	43.7	45.1	68.9	70.0
Yang et al. [2018]	70.3	71.5	79.3	79.9	78.9	78.4	-	-	-	-
Mukherjee et al. [2018]	-	-	79.2	84.5	-	-	-	-	-	-
Alvarez-Melis et al. [2018]	71.1	73.8	81.8	81.3	81.6	82.9	55.4	41.7	72.5	69.9
Xu et al. [2018]	67.0	69.3	77.8	79.5	75.5	77.9	-	-	-	-
Wang et al. [2019]	72.2	74.2	84.2	81.4	83.6	82.8	58.3	45.0	74.6	70.8
Reassign + context freezing (Chapter 3)	76.7	77.5	**86.8**	84.5	84.4	**84.4**	**66.4**	**52.8**	**78.6**	74.8
Full method w/ mapping initialization	76.6	**78.0**	86.6	84.0	84.9	84.6	65.6	51.4	78.4	**75.0**
Full method w/ numeral initialization	**77.0**	77.6	86.4	**85.0**	**85.0**	**84.9**	65.2	51.7	78.4	74.8
Full method w/ identical word initialization	76.4	77.8	86.5	84.1	**85.0**	84.8	65.9	51.6	78.4	74.6

Table 4.4: Results of the proposed method in comparison to previous work (P@1). All systems are fully unsupervised and use SGNS embeddings trained on Wikipedia.

don't have access to a validation set, doing multiple random restarts is a good way to ensure convergence is achieved.

4.4.4 Comparison to the state-of-the-art

In this section we again compare our results to the numbers reported in the literature. As in Chapter 3, it is important to note that many works in the literature use pre-trained *fastText* embeddings, while we train our own using *word2vec*, so the numbers aren't directly comparable. However, we have used the same Wikipedia training corpora and SGNS hyper-parameters in order to make the comparison as direct as possible. The results are shown in table 4.4.

Again, we observe that all three versions of our method, corresponding to different initialization strategies, outperform all others by a significant margin.

4.4.5 Error analysis

In this section we will analyze the type of mistakes our final model makes, as compared to the unsupervised mapping VecMap baseline. Due to space constraints we will focus on the identical word based initialization.

	de-en		es-en		fr-en		fi-en		ru-en		zh-en		avg
	→	←	→	←	→	←	→	←	→	←	→	←	
Gained correct translations	64	67	70	45	61	55	145	187	75	116	110	184	98.3
Lost correct translations	37	48	26	32	25	29	101	94	78	88	112	84	62.9
Number of words translated identically													
Mapping baseline	423	445	325	384	447	492	310	253	0	38	0	133	
Full method w/ identical word init	466	542	376	438	487	557	393	374	0	63	0	184	
Gold standard dictionary	524	586	445	492	644	619	464	432	0	59	1	296	

Table 4.5: Number of gained and lost correct translations for each language pair, as well as identically translated words by each method. Each test dictionary has 1500 total entries. See text for further details.

In order to analyze whether both models tend to fail on the same words, we calculated how many correct translations are gained and lost by switching from the baseline mapping method to our new method. We count a dictionary entry as gained if the mapping method translated it incorrectly, but our method translated it correctly. Similarly, we count it as lost if our method gave a wrong translation and the mapping baseline gave a correct one. A manual analysis of the outputs of our method compared to the baseline did not recognize any pattern, except a large number of words that were translated identically (i.e. gave the word itself as the translation). In order to test this we counted the number of words translated identically by each method, as well as the number of entries in the test dictionary for which the word itself is a correct translation. The results are shown in Table 4.5

We observe that our method loses 62.9 correct translations on average, while gaining 98.3. This tells us that the set of words that our model translates correctly isn't a clean superset of the words the mapping method translates correctly, and thus that they make different kinds of mistakes. This also suggests that an ensemble of both methods might perform well, which we would like to study in future work.

As for the identically translated words, we see that our model consistently translates more words identically. This is quite interesting, since there is no vocabulary sharing between languages in our model, and nothing to explicitly incentivize the same word to be chosen as a translation. We also observe that the gold standard dictionaries from the MUSE collection usually have quite a few entries where the word itself is a valid translation. This seems unusual, specially for distant language pairs such as Finnish-English, and could be explained by the fact that the MUSE set of dictionaries

was automatically generated using internal translation tools [Conneau et al., 2018]. In the future we would like to evaluate our method on a different set of cleaner dictionaries, to analyze whether the pattern remains.

4.5 Conclusions

In this chapter we have built on the method described in Chapter 3, by introducing several changes aiming to address two deficiencies identified in the conclusions of the previous chapter: the pathological properties of the space after the reassignment step, and the dependency on an existing unsupervised mapping method.

By removing the reassignment step and instead randomly initializing the weights, we fixed the translation artifacts in the target-source direction for the initial state. As seen in the learning curve Section 4.4.3, this helps achieve a good performance much quicker in this direction.

However, as seen in the ablation study of Section 4.4.2, the random initialization leads to a small dip in the resulting performance. Nevertheless, the iterative re-induction step, where we use the current embeddings to update the dictionary and freezing set after a certain amount of updates, helps bridge this gap. Our new method surpasses the previous state-of-the-art in all cases and can now work without depending on a separate mapping method, achieving an average gain in BLI P@1 of 2.5 percentage points over the unsupervised mapping baseline, when run with the unsupervised mapping based initialization.

Although the iterative re-induction process gives good results when initialized with the dictionary given by unsupervised mapping, it wasn't good enough to obtain a good solution when starting with poorer dictionaries such as the numeral or identical word based ones. Adding random restarts, where the weights and learning rates are reset while keeping the last dictionary from the previous run, helped overcome this issue, and our final method achieves good performance even when initialized with a poor numeral based dictionary. This removes the dependence on a mapping method, but it isn't strictly unsupervised, as it makes certain assumptions about the languages involved (i.e. there have to be enough identically spelled words or shared numerals to obtain a good enough initial dictionary).

All in all, the method presented in this chapter outperforms current state-of-the-art mapping based methods by a significant margin, proving that our paradigm of learning embeddings directly in a shared space without any supervision is worth pursuing.

Chapter 5

Conclusions

In this thesis we have introduced a new way to learn unsupervised cross-lingual word embeddings directly in a shared space. This approach was motivated by the limitations of the currently dominant unsupervised mapping paradigm, where monolingual embeddings are learned independently and later aligned through a linear transformation in a post-processing step. This process depends on the isometry assumption [Miceli Barone, 2016], which states that the independently trained monolingual embeddings will be approximately similar in structure, as it would otherwise be impossible to learn a linear mapping that aligns them. However, Søgaard et al. [2018] showed that this assumption is far from true under certain conditions. On the other hand, ”joint” methods trained on parallel corpora do not suffer from this structural divergence, and they outperform mapping methods when applicable [Ormazabal et al., 2019]. Thus it follows naturally that non-mapping based unsupervised methods are an avenue worth pursuing.

In Chapter 3, we introduced a novel method following this approach. Instead of learning the embeddings for each language independently, we learn them in two steps: first we learn the embeddings for the target language using a monolingual algorithm, and then we learn the representations for the source language in such a way that they are aligned with the target embeddings, which are kept fixed. This alignment is achieved through two additions to the regular embedding learning algorithm. First, we use a dictionary to ”translate” the target embeddings to the source language and use this as the initial state for learning, which we call the reassignment step. Second, we keep the context vectors for words that have a high chance of being correctly translated frozen, to make sure that alignment is retained while allowing enough flexibility for the word vectors to learn a good representation. Our new method outperformed all other unsupervised mapping methods by a significant margin in every language pair, showing

that this new paradigm is a good alternative to mapping methods, and opening a new research line. The work described in this chapter is currently under review as a short paper in the EMNLP 2020 conference.

Although the method introduced in Chapter 3 showed very promising results, we identified two main issues with it. On one hand, the reassignment step caused many vectors in the source language embeddings to be identical, which lead to some translation artifacts in the target-source direction. On the other hand, and more importantly, the method was still dependent on an unsupervised mapping algorithm, as it was used to generate the initial dictionary. In Chapter 4, we introduced several improvements to our method to address these issues. First, we removed the reassignment step, and re-formulated the context freezing in terms of translating some of the context vectors to the opposite language on-the-fly, which effectively amounts to only initializing the vectors that are frozen with those from the target embeddings. Second, we iteratively re-induce the dictionary every k iterations, and repeat the whole training process multiple times, restarting the embeddings to a random state while keeping the last dictionary from the previous run each time. These additions made it possible for the method to work with very weak initializations, such as a dictionary only consisting of numerals, while achieving similar or better performance. Thus our improved method is no longer dependent on a separate mapping method.

All in all, the methods developed in this thesis serve not only as a plug-in improvement to existing systems that rely on unsupervised cross-lingual embeddings, such as unsupervised machine translation systems [Artetxe et al., 2018b, Lample et al., 2018a], but as the foundation of a new way to learn cross-lingual embeddings.

5.1 Future work

Despite its simplicity, our approach obtains substantial improvements over the previous state-of-the-art. However, there are many potential areas for improvement that we could investigate. While we removed the dependency on a mapping method by initializing our algorithm with heuristic-based dictionaries, this approach is not purely unsupervised, as it makes some assumptions about the languages involved (i.e. that they use shared numerals, or that there will be enough identically spelled words for the initialization to work). In fact, we have seen that these heuristic initializations can fail, such as for the numeral based initialization in the Chinese-English case, where there weren't many shared numerals. In the future, we would like to further reduce this need for supervision

and achieve a completely unsupervised method under our paradigm, similar to what has been achieved in the mapping scenario [Conneau et al., 2018, Artetxe et al., 2018a].

Additionally, although our approach was motivated by the shortcomings of existing mapping methods, we have carried out our experimentation in a setting where current mapping methods are successful (i.e. using comparable Wikipedia corpora). This allowed us to directly compare our results to previous work, but we would like to analyze the performance of our method in an scenario were existing unsupervised mapping algorithms fail, or exhibit very poor performance. For example, it would be interesting to see how our method performs under strong domain mismatches in the corpora.

Finally, we would like to evaluate our method on tasks other than bilingual lexicon induction. Although we focused on it due to time constraints and for ease of comparison, the BLI task is just one of many ways to evaluate cross-lingual word embeddings, and it has been shown that methods that are tailored to dictionary induction tasks don't always perform well on downstream tasks [Glavaš et al., 2019]. Thus we believe it would be interesting to evaluate our model on different tasks, such as cross-lingual transfer.

Bibliography

Aitor Ormazabal. Cross-lingual word embedding methods and their limitations . Personal communication., 2020. Informatics Research Review.

David Alvarez-Melis and Tommi Jaakkola. Gromov-wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1881–1890, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D18-1214.

David Alvarez-Melis, Stefanie Jegelka, and Tommi S Jaakkola. Towards optimal transport with global invariances. *arXiv preprint arXiv:1806.09277*, 2018.

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, Vancouver, Canada, July 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/P17-1042.

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. Association for Computational Linguistics, 2018a. URL http://aclweb.org/anthology/P18-1073.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. In *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*, April 2018b. URL https://openreview.net/pdf?id=Sy2ogebAW.

Giusepppe Attardi. Wikiextractor. https://github.com/attardi/wikiextractor, 2015.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. doi: 10.1162/tacl_a_00051. URL https://www.aclweb.org/anthology/Q17-1010.

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1082. URL https://www.aclweb.org/anthology/D14-1082.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. In *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*, April 2018. URL https://openreview.net/pdf?id=H196sainb.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D18-1045.

J. R. Firth. Applications of general linguistics. *Transactions of the Philological Society*, 56(1):1–14, 1957. doi: 10.1111/j.1467-968X.1957.tb00568.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-968X.1957.tb00568.x.

Goran Glavaš, Robert Litschko, Sebastian Ruder, and Ivan Vulić. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In *Proceedings of ACL*, 2019.

Edouard Grave, Armand Joulin, and Quentin Berthet. Unsupervised alignment of embeddings with wasserstein procrustes. *arXiv preprint arXiv:1805.11222*, 2018.

Yedid Hoshen and Lior Wolf. Non-adversarial unsupervised word translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 469–478, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D18-1043.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proc. of EMNLP 2018*, pages 2979–2984. Association for Computational LinguisticsACL, 2018.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL https://www.aclweb.org/anthology/D14-1181.

Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. In *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*, April 2018a. URL https://openreview.net/pdf?id=rkYTTf-AZ.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium, October-November 2018b. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D18-1549.

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. Association for Computational Linguistics, 2015. doi: 10.3115/v1/P15-1027. URL http://aclweb.org/anthology/P15-1027.

Thang Luong, Hieu Pham, and Christopher D. Manning. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159. Association for Computational Linguistics, 2015. doi: 10.3115/v1/W15-1521. URL http://aclweb.org/anthology/W15-1521.

Antonio Valerio Miceli Barone. Towards cross-lingual distributed representations without parallel text trained with adversarial autoencoders. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 121–126. Association for Computational Linguistics, 2016. doi: 10.18653/v1/W16-1614. URL http://aclweb.org/anthology/W16-1614.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, 2013b. URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositional pdf.

Tanmoy Mukherjee, Makoto Yamada, and Timothy Hospedales. Learning unsupervised word translations without adversaries. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 627–632, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D18-1063.

Ndapa Nakashole. NORMA: Neighborhood sensitive maps for multilingual word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 512–522, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1047. URL https://www.aclweb.org/anthology/D18-1047.

Aitor Ormazabal, Mikel Artetxe, Gorka Labaka, Aitor Soroa, and Eneko Agirre. Analyzing the limitations of cross-lingual word embedding mappings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4990–4995, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1492. URL https://www.aclweb.org/anthology/P19-1492.

Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R Gormley, and Graham Neubig. BLISS in non-isometric embedding spaces, 2019. URL https://openreview.net/forum?id=Bkg93jC5YX.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://www.aclweb.org/anthology/D14-1162.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2018. URL https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(Sep):2487–2531, 2010.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. On the limitations of unsupervised bilingual dictionary induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/P18-1072.

Zirui Wang, Jiateng Xie, Ruochen Xu, Yiming Yang, Graham Neubig, and Jaime Carbonell. Cross-lingual alignment vs joint training: A comparative study and a simple unified framework. *arXiv preprint arXiv:1910.04708*, 2019.

Ruochen Xu, Yiming Yang, Naoki Otani, and Yuexin Wu. Unsupervised cross-lingual transfer of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2465–2474, Brussels,

Belgium, October-November 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D18-1268.

Pengcheng Yang, Fuli Luo, Shuangzhi Wu, Jingjing Xu, Dongdong Zhang, and Xu Sun. Learning unsupervised word mapping by maximizing mean discrepancy. *arXiv preprint arXiv:1811.00275*, 2018.

Appendix A

Obtaining the Wikipedia Corpora

Wikipedia dumps are freely available to download for every language, but they come in XML format with embedded metadata, while we need plain text to train embeddings. In this appendix we describe the process we followed to obtain and clean the Wikipedia corpora.

First, we download the XML dumps from `https://dumps.wikimedia.org/`. Then, we use the WikiExtractor [1] [Attardi, 2015] script to extract the plain text from these dumps, running the following command:

```
python WikiExtractor.py dump_name.xml -o extracted
```

Finally, we use scripts from the MOSES machine translation library [2] to pre-process the extracted text, by lowercasing, tokenizing, and normalizing punctuation. Specifically, we run the following command:

```
$MOSES/scripts/tokenizer/normalize-punctuation.perl -l $LANG | \
$MOSES/scripts/tokenizer/remove-non-printing-char.perl | \
$MOSES/scripts/tokenizer/tokenizer.perl -q -a -l $LANG -no-escape
-threads $THREADS | \
$MOSES/scripts/tokenizer/lowercase.perl
```

The output of this command is the final pre-processed corpus that we use in our experiments.

[1] `https://github.com/attardi/wikiextractor`
[2] `https://github.com/moses-smt/mosesdecoder`