A Simple and Effective Approach to Robust Unsupervised Bilingual Dictionary Induction

Yanyang Li[1*], **Yingfeng Luo**[1*], **Ye Lin**[1], **Quan Du**[1],
Huizhen Wang[1], **Shujian Huang**[3,4], **Tong Xiao**[1,2†] and **Jingbo Zhu**[1,2]
[1]Natural Language Processing Lab., Northeastern University, Shenyang, China
[2]NiuTrans Research, Shenyang, China
[3]National Key Laboratory for Novel Software Technology, Nanjing, China
[4]Nanjing University, Nanjing, China
blamedrlee@outlook.com, 1971646@stu.neu.edu.cn,
{linye2015, duquanneu}@outlook.com, huangsj@nju.edu.cn,
{wanghuizhen, xiaotong, zhujingbo}@mail.neu.edu.cn

Abstract

Unsupervised Bilingual Dictionary Induction methods based on the initialization and the self-learning have achieved great success in similar language pairs, e.g., English-Spanish. But they still fail and have an accuracy of 0% in many distant language pairs, e.g., English-Japanese. In this work, we show that this failure results from the gap between the actual initialization performance and the minimum initialization performance for the self-learning to succeed. We propose *Iterative Dimension Reduction* to bridge this gap. Our experiments show that this simple method does not hamper the performance of similar language pairs and achieves an accuracy of 13.64∼55.53% between English and four distant languages, i.e., Chinese, Japanese, Vietnamese and Thai.

1 Introduction

Unsupervised Bilingual Dictionary Induction (UBDI) is a task that aims to find the word translations given the monolingual word embeddings of two languages. Recent UBDI methods have shown promising results on similar language pairs such as English-Spanish (Artetxe et al., 2017; Lample et al., 2018; Zhou et al., 2019; Hartmann et al., 2019; Ren et al., 2020). These methods are mostly based on the initialization and the self-learning. The initialization first constructs a dictionary from the word embeddings, then the self-learning starts with this dictionary and alternates between refining the source-target word embedding mapping and inducing a new dictionary with this mapping.

Despite the success of UBDI, recent work has questioned the robustness of UBDI methods on distant language pairs (Søgaard et al., 2018; Vulić et al., 2019; Glavas et al., 2019), e.g., English-Japanese. They show that even for the most robust system VecMap (Artetxe et al., 2018), it still fails and has an accuracy of 0% on 87 out of 210 distant language pairs (Vulić et al., 2019). To be consistent with Artetxe et al. (2018), we define a system 'succeeds' when it has an accuracy above 5% and 'fails' otherwise.

Previous work has investigated how different properties of languages have an impact on UBDI performance (Søgaard et al., 2018). In this paper, we take a step further to inspect which part of VecMap breaks down. With a novel similarity metric to evaluate the initialization performance, we observe a gap between the actual initialization performance and the minimum initialization performance for the self-learning to succeed in distant language pairs.

We find that the dimension reduction approach is very effective in bridging this gap. Therefore, we propose *Iterative Dimension Reduction* (IDR) to improve the robustness of VecMap and avoid performance loss due to dimension reduction. IDR first reduces the dimension of word embeddings and performs unsupervised learning on them. Then it initializes the self-learning on larger dimension embeddings using this learned system. This simple dimension reduction removes unimportant or noisy features, making the algorithm easier to find a proper solution to distant language pairs.

[*] Authors contributed equally.

[†] Corresponding author.

Proceedings of the 28th International Conference on Computational Linguistics, pages 5990–6001
Barcelona, Spain (Online), December 8-13, 2020

Entry	En-Zh	Zh-En	En-Ja	Ja-En	En-Vi	Vi-En	En-Th	Th-En
VecMap Accuracy [%]	0.07	0	1.03	32.67	0.73	0.73	0	0.07
Maximum Accuracy [%]	37.07	35.20	49.01	33.36	47.13	57.80	24.20	17.76
Minimum Dictionary Size	150	74	74	97	137	110	147	154

Table 1: VecMap accuracy, the maximum accuracy of using a seed dictionary to initialize the self-learning and the minimum seed dictionary size to obtain that accuracy (we start with 10 pairs to estimate this size and add 10 pairs if the maximum accuracy is not achieved; Results are averaged over 3 runs).

We evaluate our approach on four similar European language pairs, including English-{Spanish, French, Italian, German} (En-{Es, Fr, It, De}), and four distant language pairs, including English-{Chinese, Japanese, Vietnamese, Thai} (En-{Zh, Ja, Vi, Th}). Our method not only has a close performance to the VecMap baseline in similar language pairs but also succeeds in all distant language pairs. In four distant language pairs, our method has an accuracy of 13.64∼55.53%, whereas the VecMap baseline has an accuracy of 0% in most cases, as shown in the first row of Table 1.

2 The VecMap Method

VecMap (Artetxe et al., 2018) learns weights W_X and W_Y for the source and target word embeddings X and Y and maps them to the same space for inducing the dictionary. It consists of two components:

- *Initialization*. The initialization first computes $M_X = XX^T$ and $M_Y = YY^T$. Then each row of M_X and M_Y is sorted and an initial dictionary D is induced by searching for nearest neighbors between the rows of $\sqrt{M_X}$ and $\sqrt{M_Y}$.

- *Self-learning*. With the initial dictionary D, the self-learning iterates the following two steps:
  - It finds W_X and W_Y that maximize $\sum_i \sum_j D_{ij}((X_{i*}W_X) \cdot (Y_{j*}W_Y))$ for the current dictionary D, where $D_{ij} = 1$ if the j-th target word is the translation of the i-th source word and 0 otherwise. An optimal solution is given by $W_X = U$ and $W_Y = V$, where $USV^T = X^TDY$ is the singular value decomposition of X^TDY;
  - A new dictionary D is induced by using the CSLS retrieval (Lample et al., 2018) to extract nearest neighbors in the similarity matrix $P = XW_XW_Y^TY^T$. To avoid being trapped in a poor local optimum and encourage the exploration of possible word translations, similarity scores in P are kept with a probability p and set to 0 otherwise.

There are some pre-processing and post-processing steps that are crucial to VecMap:

- *Normalization and mean centering* is applied before the initialization and the self-learning. It will normalize all vectors in X and Y to have a unit Euclidean norm. Then these vectors are mean centered dimension-wise and length-normalized again.

- *Whitening* (Bell and Sejnowski, 1997) is applied before the last iteration of the self-learning. It transforms X and Y such that each dimension has unit variance and that the dimensions are uncorrelated.

- *Symmetric re-weighting* is applied to the mapped embeddings XW_X and YW_Y at the last iteration. It further improves XW_X and YW_Y by weighting dimensions by their cross-correlation $\sqrt{S}$, where S is a diagonal matrix with singular values on its diagonal entries.

- *Dewhitening* is the reverse of whitening and applied after symmetric re-weighting if whitening is applied. It restores the variance information.

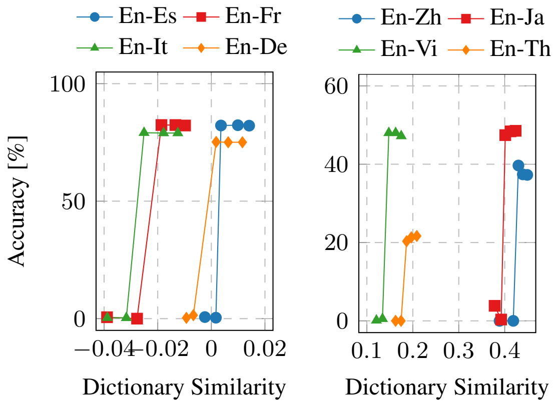

Figure 1: The accuracy obtained by the self-learning starting from a given dictionary (Accuracy) vs. the dictionary similarity of that starting dictionary (Dictionary Similarity).

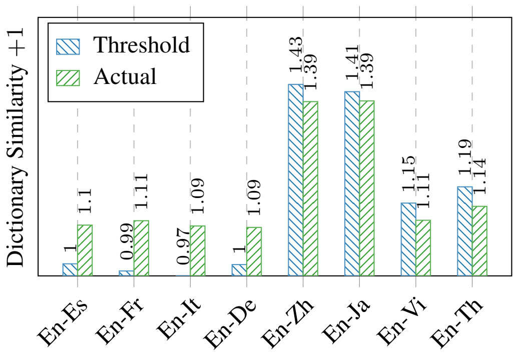

Figure 2: The minimum initial dictionary similarity to succeed (Threshold) vs. the actual initial dictionary similarity (Actual).

3 When does Unsupervised Learning Fail?

Since VecMap is pipelined by the initialization and the self-learning, we can assume the failure of unsupervised learning comes from either or both of these two components. Two hypotheses arise:

1. The self-learning cannot succeed even if the initialization is perfect;

2. The initialization is too bad to kick-off the self-learning even if the self-learning is able to succeed.

It is easy to verify the first hypothesis: we start the self-learning with a human-annotated seed dictionary. This way assumes a perfect initialization and thus eliminates the impact of the initialization. Table 1 shows that a small seed dictionary is enough for the self-learning to have a good result. This observation reveals that the self-learning is able to succeed. This left us the second hypothesis, that unsupervised learning fails at the initialization.

Two natural questions come from the second hypothesis:

1. How to quantify the initialization performance, i.e., the quality of the dictionary generated by the initialization (initial dictionary in short)?

2. How well the initial dictionary need to be so that the self-learning can succeed?

One might expect the accuracy is a good proxy of the quality of a dictionary. But the accuracy only takes the correct translations into account. The intuition is that though the system fails to find the correct translation, it can still be useful if its translations are close to the correct answers. Thus we evaluate the average cosine similarity between the word embeddings of the system translations and the correct answers, dubbed *dictionary similarity*. It will score high if the translations are close to the correct answers.

Figure 1 shows how the self-learning performs when starting from dictionaries with different dictionary similarities. These dictionaries are constructed by randomly replacing the translations in the initial dictionary. We can see that the self-learning only succeeds when the dictionary similarities of these starting dictionaries are above some thresholds. These thresholds represent the minimum similarity that the initial dictionary should have for the self-learning to succeed.

We test the dictionary similarity of the initial dictionary. As shown in Figure 2, the actual initial dictionary similarities of the initialization are above the thresholds in similar language pairs, but it is the opposite in distant language pairs. The gap between the actual initialization similarity and the minimum similarity for the self-learning to succeed implies the failure of VecMap in distant language pairs.

Metric	En	Es	Fr	It	De	Zh	Ja	Vi	Th
Explained variance [%]	6.930	5.636	4.331	4.508	4.446	49.17	96.71	8.632	11.19
Similarity (before dropmax)	0.155	0.158	0.156	0.158	0.163	0.812	0.991	0.179	0.186
Similarity (after dropmax)	0.062	0.029	0.042	0.023	0.025	0.051	0.033	0.012	0.012

Table 2: The percentage of variance explained by the highest eigenvalue and the average cosine similarity between any two embeddings before and after applying the dropmax trick.

4 Proposed Method

As the gap is determined by embeddings and the algorithm, one can improve the algorithm to bridge this gap. In this work, we start with a different angle by simplifying embeddings to make the algorithm easier to succeed. Concretely, we run the dimension reduction on the word embeddings. In principle, the dimension reduction algorithm will drop features that are less important in explaining the data. It can be considered as a way to remove the noise and clean up the data. Here we choose Principal Component Analysis (PCA) (Pearson, 1901) for study.

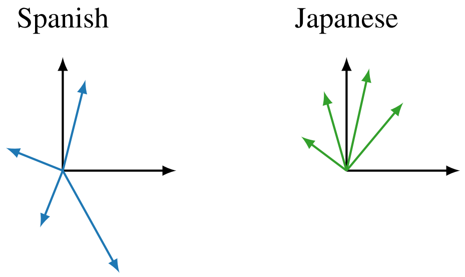

Figure 3: An example of embeddings with eigenvectors from PCA (Black arrows are eigenvectors/axes, coloured arrows are projected embedding vectors).

PCA first computes the covariance matrix of the features in the embeddings. This matrix represents the correlation among features. Then PCA performs the eigenvalue decomposition on this covariance matrix to obtain a set of eigenvectors. Finally, PCA uses eigenvectors with the highest n eigenvalues to project the raw embeddings to space with a lower dimension n.

However, this simple application of PCA only works in a few languages. We find that the highest eigenvalue is much larger than the others in most of the failed languages, e.g., it explains 96% variance in Japanese while 5.6% in Spanish as shown in the first row of Table 2 (the larger the highest eigenvalue is, the more variance it explains). This implies that the embeddings will be stretched the most on the direction of the eigenvector with the highest eigenvalue, resulting in most embeddings pointed to a close direction and being clustered together. Figure 3 shows a simple example of this. The average cosine similarity between every two embeddings in Table 2 also justifies our hypothesis that the more variance

Algorithm 1 Iterative Dimension Reduction

```
procedure IDR($E, n$)                       ▷ $E$ is the raw embeddings, $n$ is the initial dimension
    $D \leftarrow \varnothing$                                    ▷ Set the dictionary to empty
    while $n \leq 300$ do                        ▷ 300 is the dimension of the raw embeddings
        Reduce $E$ to $\bar{E}$ with dimension $\min(n, 300)$ using PCA and dropmax
        if $D = \varnothing$ then
            Run the initialization and the self-learning on $\bar{E}$
        else
            Run the self-learning on $\bar{E}$ with $D$ as the initial dictionary
        end if
        Translate 4K most frequent words and store the results in $D$
        $n \leftarrow n \times 2$
    end while
    return $W_X$ and $W_Y$
end procedure
```

System	En-Es	Es-En	En-Fr	Fr-En	En-It	It-En	En-De	De-En
MUSE (Lample et al., 2018)	83.20	83.66	82.66	82.39	78.20	77.90	75.10	72.93
VecMap (Artetxe et al., 2018)	82.33	84.60	82.47	83.60	79.13	79.80	75.33	74.27
C-MUSE (Hartmann et al., 2019)	82.33	84.73	82.40	83.73	79.13	79.67	75.27	74.20
POSTPROC (Vulic et al., 2020)	82.73	85.47	82.73	84.00	79.13	80.60	76.00	75.33
Proposed method (dim 50)	40.33	37.40	53.53	48.07	48.13	45.00	31.60	30.33
Proposed method (dim 100)	63.47	61.80	72.73	71.13	68.40	66.73	63.67	61.27
Proposed method (dim 200)	80.33	80.27	80.40	79.67	76.47	76.67	71.27	71.20
Proposed method (dim 300)	82.40	84.60	82.60	83.67	78.93	79.67	75.33	74.33

Table 3: The accuracy of different UBDI systems on similar language pairs (the proposed method starts with dimension 50 and doubles the dimension in each step until it reaches 300. We show the performance of each step).

System	En-Zh	Zh-En	En-Ja	Ja-En	En-Vi	Vi-En	En-Th	Th-En
MUSE (Lample et al., 2018)	34.93	32.33	0.06	5.09	0	0	0	0
VecMap (Artetxe et al., 2018)	0.07	0	1.03	32.67	0.73	0.73	0	0.07
C-MUSE (Hartmann et al., 2019)	39.80	34.53	0.27	32.18	0	56.8	0	0
POSTPROC (Vulic et al., 2020)	0.00	44.07	0.00	34.60	0.13	0.07	0.07	0.00
Proposed method (dim 50)	0.07	0	10.08	6.89	0.13	0.20	0.40	0
Proposed method (dim 100)	0.2	24.47	27.55	19.99	0.87	0.40	10.40	5.87
Proposed method (dim 200)	34.13	35.93	39.07	28.67	2.60	2.13	19.47	11.75
Proposed method (dim 300)	37.33	35.27	48.87	33.08	47.6	55.53	21.60	13.64

Table 4: The accuracy of different UBDI systems on distant language pairs (the proposed method starts with dimension 50 and doubles the dimension in each step until it reaches 300. We show the performance of each step).

it explains, the closer the embeddings are to each other (higher similarity). Such a phenomenon makes embeddings indistinguishable. The 'hubness' problem (Radovanovic et al., 2010) will occur, where one embedding is the nearest neighbor of many embeddings, even if they have different semantics.

The simplest solution is that when projecting the embeddings with a dimension m to n, we drop not only eigenvectors with the lowest $m - n + 1$ eigenvalues but also the one with the highest eigenvalue. We refer to this as the *dropmax* trick. This makes embeddings distinguishable (lower similarity) as shown in the last row of Table 2 and helps VecMap to succeed in the remaining distant language pairs. Since the dimension reduction incurs the loss of information and hinders the further improvement of our system, we propose *Iterative Dimension Reduction* (IDR), as shown in Algorithm 1. The algorithm first runs VecMap on embeddings with the smallest dimension n (default set to 50). Then it uses the trained system to translate the K most frequent words (default set to 4K). The resulting dictionary will serve as the initial dictionary for the self-learning in the next step, where it runs on embeddings with a larger dimension (default set to $2\times$ larger than in the previous step).

5 Experiments

5.1 Setup

We compare our method with four popular UBDI systems: MUSE[1] (Lample et al., 2018), VecMap[2] (Artetxe et al., 2018), C-MUSE (Hartmann et al., 2019) and POSTPROC (Vulic et al., 2020). We

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

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

System	En-Zh	Zh-En	En-Ja	Ja-En	En-Vi	Vi-En	En-Th	Th-En
Proposed method	37.33	35.27	48.87	33.08	47.6	55.53	21.60	13.64
- IDR	40.20	41.07	47.64	34.18	0.13	0.20	0.07	0
- Dropmax	0	0	0	0	0.33	0.27	20.73	14.25

Table 5: Ablation study of the proposed method on distant language pairs.

reproduce the C-MUSE and POSTPROC system using Python. All these systems are run with the default hyper-parameters settings. Our method is based on the open-sourced VecMap implementation.

We evaluate the baseline and our method on 4 similar language pairs, En-{Es, Fr, It, De}, and 4 distant language pairs, En-{Zh, Ja, Vi, Th}. We use the pretrained 300-dimensional fastText embeddings (Bojanowski et al., 2017)[3]. The evaluation dictionaries are from Lample et al. (2018). We trim all vocabularies to the 20K most frequent words for training. Specifically, VecMap retains the top-4K words for the initialization, while others use the whole vocabulary. All experiments are done on a single Nvidia GTX 1080Ti. We run each experiment 3 times but with different random seeds, then pick the one with the highest cosine similarity of induced nearest neighbors as the final result. This unsupervised model selection criterion has shown to correlate well with UBDI performance (Hartmann et al., 2019).

5.2 Results

Table 3 shows the results of various systems on similar language pairs, En-{Es, Fr, It, De} and their reverse {Es, Fr, It, De}-En. We can see that all baseline systems perform well on these language pairs. VecMap and C-MUSE outperform MUSE in most cases. This is because both systems employ the dropout trick in their self-learning processes, which has proven to be effective in jumping out of the local optimum (Artetxe et al., 2018; Hartmann et al., 2019). However, all these baseline systems perform poor on distant language pairs, En-{Zh, Ja, Vi, Th} and their reverse {Zh, Ja, Vi, Th}-En, as shown in Table 4. C-MUSE is better than the others by obtaining positive results on En-Zh, Zh-En, Ja-En and Vi-En tasks, but still fails on other tasks, i.e., having an accuracy below 5%.

Our method is based on VecMap, thus it has a good performance in similar language pairs, as shown in the last row of Table 3. On the other hand, our method is robust to distant language pairs as shown in Table 4. In all four distant language pairs and the two directions, our method obtains much better results than the baselines. For example, our method has an accuracy of 21.6% in En-Th and 13.64% in Th-En, where none of the baseline systems has an accuracy above 1%.

We also observe that the performance of our method in low-dimensional space is much worse than the one in high-dimensional space. For instance, our method has an accuracy of 21.6% in En-Th when the dimension is 300, while only 10.4% when the dimension is 100. This observation justifies our previous claim in Section 4 that dimension reduction incurs the loss of information and thus hinders the further improvement of our method. But directly run on high-dimensional embeddings does not succeed, as none of the baselines consistently has an accuracy above 0% in raw 300-dimensional embeddings. Therefore, it is necessary to run on the low-dimensional space first as a warm start of the high-dimensional counterpart to obtain better performance.

6 Analysis

6.1 Ablation Study

Table 5 shows the results of using IDR and dropmax solely on distant language pairs. We can see that IDR is crucial to En-Th and Th-En. In Section 6.2, we show that dimension reduction helps to obtain isomorphic embeddings. These embeddings match the isomorphic assumption made by VecMap.

On the other hand, the dropmax trick is crucial to En-{Zh, Ja} and their reverse. This fact relates well with the observation in Table 2, where these two languages suffer from the hubness problem due to the

[3] https://fasttext.cc/docs/en/pretrained-vectors.html

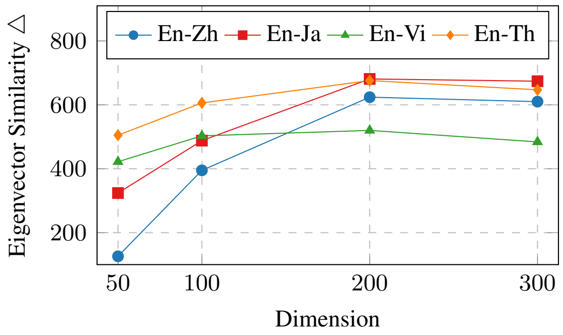

Figure 4: The eigenvector similarity vs. different dimensions on distant language pairs.

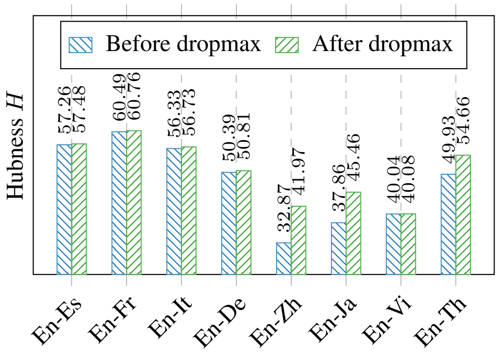

Figure 5: The hubness level before and after applying the dropmax trick.

highest eigenvalue. The dropmax trick avoids this issue by removing this highest eigenvalue, as shown in Section 6.3. We also see that both IDR and dropmax are crucial to En-Vi and Vi-En, which implies that the hubness and isomorphism are their central problems.

6.2 Isomorphism

Many UBDI methods, including VecMap, make the isomorphic assumption, that the underlying nearest neighbor graphs of two language embedding spaces are connected in the same way. Søgaard et al. (2018) propose the eigenvector similarity to measure how well this assumption is held. Here we are interested in how the isomorphism of the underlying graphs change when the dimension is different. We first normalize, center and normalize the embeddings as in the pre-processing step, calculate the nearest neighbor graphs of the 10K most frequent words in each language, and compute their Laplacian matrices L_1 and L_2. We then find the smallest k_1 such that the sum of the largest k_1 eigenvalues of L_1 is at least 90% of the sum of all its eigenvalues, and analogously for k_2 and L_2. Finally we set $k = \min(k_1, k_2)$, and define the eigenvector similarity of the two graphs as the sum of the squared differences between the k largest eigenvalues λ of L_1 and L_2, $\triangle = \sum_{i=1}^{k}(\lambda_{1i} - \lambda_{2i})^2$. The higher $\triangle$ is, the less similar the graphs are.

As shown in Figure 4, the eigenvector similarity drops significantly when the dimension is reduced. This implies that the underlying nearest neighbor graphs of two languages become similar in low-dimensional space. This helps the algorithm to succeed in low-dimensional space as the assumption it makes is held. This phenomenon might be the result that many language pairs share some principle axes of variation, especially the ones with high eigenvalues (Hoshen and Wolf, 2018).

6.3 Hubness

Cross-lingual word embeddings are known to suffer from the hubness problem (Lample et al., 2018), where a few points (known as *hubs*) are the nearest neighbors of many other points in high-dimensional spaces. As suggested in Section 4, distant language pairs might suffer more from this problem and the dropmax trick helps to alleviate this problem. Thus we would like to know to what extent the dropmax trick helps in the hubness problem. Here we adopt the hubness metric proposed by Ormazabal et al. (2019) for evaluation. This metric measures the percentage of target words H that are the nearest neighbor of all the source words. For instance, a hubness value of $H = 60\%$ would indicate that 60% of the target words are the nearest neighbors of all the source words. This way, lower values of H are indicative of a higher level of hubness.

Figure 5 is the hubness level of different languages before and after applying the dropmax trick. We can see that the dropmax trick generally alleviates the hubness problem, even when the hubness level is low. For example, En-It has $H = 56.39\%$ before applying the dropmax trick. After that, it has $H = 56.73\%$, a 0.34% improvement on the hubness level. For those language pairs with a high hubness level such as En-Zh and En-Ja, the improvement is obvious, e.g., more than 9% improvement on En-Zh.

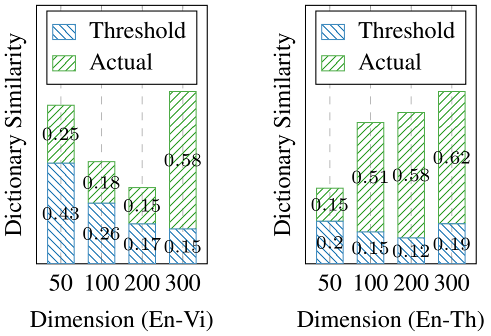

Figure 6: The minimum initial dictionary similarity to succeed (Threshold) and the actual initial dictionary similarity (Actual) vs. dimensions.

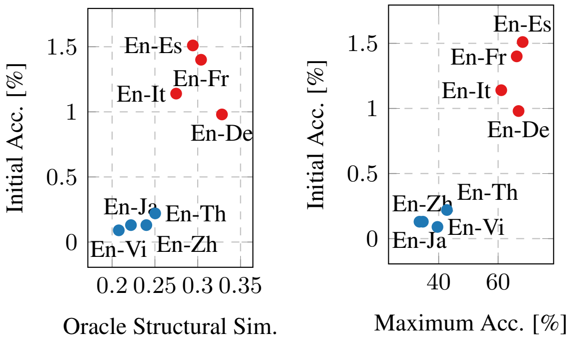

Figure 7: The initial accuracy (Initial Acc.) vs. the oracle structural similarity (Oracle Structural Sim.) and the maximum accuracy (Maximum Acc.).

6.4 Dictionary Similarity

As suggested in Section 4, dimension reduction helps to bridge the performance gap and we examine it here. This gap can be measured by the dictionary similarity proposed in Section 3. Here we choose En-$\{$Vi, Th$\}$ for study, since dimension reduction is crucial to their successes as shown in Table 5.

In Figure 6, we can see that the gap between the initialization and the self-learning varies in different dimensions. In the dimension that VecMap succeeds, the actual initial dictionary similarity is much higher than the minimum initial dictionary similarity to succeed, e.g., 0.58 vs. 0.15 in 300 dimensions for En-Vi and 0.51 vs. 0.15 in 100 dimensions for En-Th. This means that VecMap in the previous dimension reduction step generates a good initial dictionary. When looking at the results in the previous step, we find that the actual initial dictionary similarity is close to but not surpass the minimum initial dictionary similarity to succeed, e.g., 0.15 vs. 0.17 in 200 dimensions for En-Vi and 0.15 vs. 0.2 in 50 dimensions for En-Th. This implies that though VecMap does not succeed in that dimension, closing this gap already allows it to translate well on the frequent words, i.e., words to construct the initial dictionary. This enables unsupervised learning in the next dimension reduction step.

6.5 Understanding the Initialization

In Figure 2, the initialization is poor on distant language pairs as indicated by the gap. We investigate which factor might have an impact on the accuracy of the initial dictionary (initial accuracy in short). It allows us to identify the obstacles in the initialization of distant language pairs. Here we measure the initial accuracy instead of the proposed dictionary similarity, as its value depends on embeddings and thus can not be compared across languages.

Here we study two factors: the oracle structural similarity and the maximum accuracy. Structural similarity is the cosine similarity between rows of $\sqrt{M_X}$ and $\sqrt{M_Y}$, where a row in $\sqrt{M_X}$ associates to a source word and represents the similarities between this source word and other source words, and analogously for $\sqrt{M_Y}$. This similarity is used in the initialization to select word translations to construct the initial dictionary. The oracle structural similarity is the average of the structural similarity of all possible and correct word translations in the initialization. This oracle structural similarity measures how strong the assumption made in the initialization is, which assumes aligned source and target words should have high structural similarity. The higher the oracle structural similarity is, the easier the initialization finds correct translations. As shown in the left part of Figure 7, the oracle structural similarity is positively related to the initial accuracy. Distant language pairs have a low similarity, which means that the assumption made in the initialization is unlikely to hold.

The maximum accuracy is the accuracy that is obtained by a perfect initialization strategy. It is lower than 100% as the vocabulary in the initialization might not contain the translations of some source and target words. We see that in the right part of Figure 7, the maximum accuracy is also positively related to the initial accuracy. Distant language pairs have a low maximum accuracy, which means that the initialization is less likely to find the correct translation for a source word.

6.6 Error Analysis

In Table 4, we can see that even for our best system it still has low accuracy in distant language pairs. To identify the main source of errors, we perform an error analysis of the system output in a sized 5K En-Zh dictionary from Lample et al. (2018). We randomly sample 200 error examples and let a human expert to classify these examples into four main categories: the answer and the translation are correct (CC), the answer is correct and the translation is wrong (CW), the answer is wrong and the translation is correct (WC), the answer and the translation are wrong (WW).

In our analysis, there are 25.5% errors are CC. This is due to the polysemy of words and the dictionary does not cover all possible translations. A few cases are WC (0.5%) and WW (3%). This means that there are some minor issues on the dictionary quality. For the main category CW (71%), there are 17% resulted from proper nouns, which have shown to be meaningless in the evaluation (Kementchedjhieva et al., 2019). 18.5% have a close meaning to the answer. 10.5% are the untranslated error, where the translation is identical to the source word. The remaining 25% are true errors, e.g., antonym.

7 Related Work

In recent years a number of methods have been proposed to learn bilingual dictionary from monolingual word embeddings. Early work (Mikolov et al., 2013) relies on a seed dictionary to learn the source-target word embedding mapping. Xing et al. (2015) enforce the word embeddings to be of unit length and the orthogonal constraint on the linear mapping. Faruqui and Dyer (2014) on the other hand use Canonical Correlation Analysis (CCA) to project both source and target embeddings to a common low-dimensional space. Artetxe et al. (2016) show that the above methods are variants of the same objective. Smith et al. (2017) further show that this objective is closely related to the orthogonal Procrustes problem. Artetxe et al. (2017) obtain competitive results using the self-learning with a seed dictionary of only 25 word pairs.
Adversarial methods. Zhang et al. (2017a) attempt the unsupervised bilingual dictionary induction task using the adversarial network. They use a generator to transform the source word embeddings to the target word embeddings and a discriminator to classify whether the given embedding is sampled from the true target word embeddings or generated by the generator. The generator is trained to fool the discriminator and the discriminator is trained to identify the generated word embeddings. In the end, the generator will be used to induce the bilingual dictionary. Their following work (Zhang et al., 2017b) minimizes Earth-Mover's distance between the transformed source and target embeddings distribution. Lample et al. (2018) improve the results by treating the dictionary produced by the adversarial network as the seed dictionary of the self-learning. To mitigate the hubness problem (Radovanovic et al., 2010), they propose an effective nearest neighbors retrieval method CSLS for dictionary induction. Xu et al. (2018) minimize Sinkhorn distance instead and introduce the circle consistency such that a source word embedding can be translated back after translating it to a target word. Mohiuddin and Joty (2019) extract latent codes from word embeddings and align words according to their latent codes.
Non-adversarial methods. There is another line of research that focuses on a non-adversarial approach. Artetxe et al. (2018) propose a heuristic to induce an initial dictionary by exploiting the structural similarity of embeddings. They also propose the stochastic dictionary induction method, which significantly improves the robustness as well as the performance of self-learning. Hoshen and Wolf (2018) assume that many language pairs share some principle axes of variation. Therefore they first use PCA to project the word embeddings to a lower-dimensional space. Then they apply a variant of the Iterative Closest Point method to find the source and target word embeddings mapping. Zhou et al. (2019) use normalizing flows to match the distribution of source and target word embeddings. But they rely on a numeral seed dictionary and the additional word frequency information. More recently, Hartmann et al. (2019) find that more robust results can be obtained by using the adversarial method to produce the initial dictionary for the advanced self-learning (with the stochastic dictionary induction). Artetxe et al. (2019) first generate a pseudo parallel corpus by an unsupervised machine translation system. They then extract a bilingual dictionary from the word alignment learned on that corpus. This simple process shows much better results than previous methods. Vulic et al. (2020) introduce a simple post-processing step to improve UBDI performance on distant language pairs.

8 Conclusion

In this work, we pinpoint in which part the representative UBDI system, VecMap, fails on distant language pairs. We identify a gap between the initialization performance and the minimum initialization performance for the self-learning to succeed, which is responsible for its failure. We propose Iterative Dimension Reduction to bridge this gap. Our method obtains substantial gains in distant language pairs without scarifying the performance of similar language pairs. It has shown to robust to the four distant language pairs we experiment with.

Acknowledgement

This work was supported in part by the National Science Foundation of China (Nos. 61876035 and 61732005), the National Key R&D Program of China (No. 2019QY1801). The authors would also like to thank anonymous reviewers for their valuable comments.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2289–2294. The Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 451–462. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 789–798. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2019. Bilingual lexicon induction through unsupervised machine translation. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5002–5007. Association for Computational Linguistics.

Anthony J Bell and Terrence J Sejnowski. 1997. The "independent components" of natural scenes are edge filters. *Vision Research*, 37(23):3327–3338.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, 5:135–146.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In Gosse Bouma and Yannick Parmentier, editors, *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden*, pages 462–471. The Association for Computer Linguistics.

Goran Glavas, Robert Litschko, Sebastian Ruder, and Ivan Vulic. 2019. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 710–721. Association for Computational Linguistics.

Mareike Hartmann, Yova Kementchedjhieva, and Anders Søgaard. 2019. Comparing unsupervised word translation methods step by step. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 6031–6041.

Yedid Hoshen and Lior Wolf. 2018. Non-adversarial unsupervised word translation. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 469–478. Association for Computational Linguistics.

Yova Kementchedjhieva, Mareike Hartmann, and Anders Søgaard. 2019. Lost in evaluation: Misleading benchmarks for bilingual dictionary induction. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3334–3339. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Tasnim Mohiuddin and Shafiq R. Joty. 2019. Revisiting adversarial autoencoder for unsupervised word translation with cycle consistency and improved training. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3857–3867. Association for Computational Linguistics.

Aitor Ormazabal, Mikel Artetxe, Gorka Labaka, Aitor Soroa, and Eneko Agirre. 2019. Analyzing the limitations of cross-lingual word embedding mappings. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4990–4995. Association for Computational Linguistics.

Karl Pearson. 1901. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Milos Radovanovic, Alexandros Nanopoulos, and Mirjana Ivanovic. 2010. Hubs in space: Popular nearest neighbors in high-dimensional data. *J. Mach. Learn. Res.*, 11:2487–2531.

Shuo Ren, Shujie Liu, Ming Zhou, and Shuai Ma. 2020. A graph-based coarse-to-fine method for unsupervised bilingual lexicon induction. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3476–3485. Association for Computational Linguistics.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Anders Søgaard, Sebastian Ruder, and Ivan Vulic. 2018. On the limitations of unsupervised bilingual dictionary induction. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 778–788. Association for Computational Linguistics.

Ivan Vulić, Goran Glavaš, Roi Reichart, and Anna Korhonen. 2019. Do we really need fully unsupervised cross-lingual embeddings? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4406–4417, Hong Kong, China, November. Association for Computational Linguistics.

Ivan Vulic, Anna Korhonen, and Goran Glavas. 2020. Improving bilingual lexicon induction with unsupervised post-processing of monolingual word vector spaces. In Spandana Gella, Johannes Welbl, Marek Rei, Fabio Petroni, Patrick S. H. Lewis, Emma Strubell, Min Joon Seo, and Hannaneh Hajishirzi, editors, *Proceedings of the 5th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2020, Online, July 9, 2020*, pages 45–54. Association for Computational Linguistics.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar, editors, *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 1006–1011. The Association for Computational Linguistics.

Ruochen Xu, Yiming Yang, Naoki Otani, and Yuexin Wu. 2018. Unsupervised cross-lingual transfer of word embedding spaces. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2465–2474. Association for Computational Linguistics.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017a. Adversarial training for unsupervised bilingual lexicon induction. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1959–1970. Association for Computational Linguistics.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017b. Earth mover's distance minimization for unsupervised bilingual lexicon induction. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1934–1945. Association for Computational Linguistics.

Chunting Zhou, Xuezhe Ma, Di Wang, and Graham Neubig. 2019. Density matching for bilingual word embedding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1588–1598. Association for Computational Linguistics.