LNMAP: Departures from Isomorphic Assumption in Bilingual Lexicon Induction Through Non-Linear Mapping in Latent Space

Tasnim Mohiuddin[¶], **M Saiful Bari**[¶], and **Shafiq Joty**[¶†]
[¶]Nanyang Technological University, Singapore
[†]Salesforce Research
{mohi0004, bari0001, srjoty}@ntu.edu.sg

Abstract

Most of the successful and predominant methods for Bilingual Lexicon Induction (BLI) are mapping-based, where a linear mapping function is learned with the assumption that the word embedding spaces of different languages exhibit similar geometric structures (*i.e.,* approximately *isomorphic*). However, several recent studies have criticized this simplified assumption showing that it does not hold in general even for closely related languages. In this work, we propose a novel semi-supervised method to learn cross-lingual word embeddings for BLI. Our model is independent of the isomorphic assumption and uses non-linear mapping in the latent space of two independently pre-trained autoencoders. Through extensive experiments on fifteen (15) different language pairs (in both directions) comprising resource-rich and low-resource languages from two different datasets, we demonstrate that our method outperforms existing models by a good margin. Ablation studies show the importance of different model components and the necessity of non-linear mapping.

1 Introduction

In recent years, a plethora of methods have been proposed to learn cross-lingual word embeddings (or CLWE for short) from monolingual word embeddings. Here words with similar meanings in different languages are represented by similar vectors, regardless of their actual language. CLWE enable us to compare the meaning of words across languages, which is key to most multi-lingual applications such as bilingual lexicon induction (Heyman et al., 2017), machine translation (Lample et al., 2018; Artetxe et al., 2018c), or multi-lingual information retrieval (Vulić and Moens, 2015). They also play a crucial role in cross-lingual knowledge transfer between languages (*e.g.,* from resource-rich to low-resource languages) by providing a common representation space (Ruder et al., 2019).

Mikolov et al. (2013a), in their pioneering work, learn a *linear* mapping function to transform the source embedding space to the target language by minimizing the squared Euclidean distance between the translation pairs of a seed dictionary. They assume that the similarity of geometric arrangements in the embedding spaces is the key reason for their method to succeed as they found linear mapping superior to non-linear mappings with multi-layer neural networks. Subsequent studies propose to improve the model by normalizing the embeddings, imposing an orthogonality constraint on the linear mapper, modifying the objective function, and reducing the seed dictionary size (Artetxe et al., 2016, 2017, 2018a; Smith et al., 2017).

A more recent line of research attempts to eliminate the seed dictionary totally and learn the mapping in a purely unsupervised way (Barone, 2016; Zhang et al., 2017; Conneau et al., 2018; Artetxe et al., 2018b; Xu et al., 2018; Hoshen and Wolf, 2018; Alvarez-Melis and Jaakkola, 2018; Mohiuddin and Joty, 2019, 2020). While not requiring any cross-lingual supervision makes these methods attractive, Vulić et al. (2019) recently show that even the most robust unsupervised method (Artetxe et al., 2018b) fails for a large number of language pairs. They suggest to rethink the main motivations behind fully unsupervised methods showing that with a small seed dictionary (500-1K pairs) their semi-supervised method always outperforms the unsupervised method and does not fail for any language pair. Other concurrent work (Ormazabal et al., 2019; Doval et al., 2019) also advocates for weak supervision in CLWE methods.

Almost all mapping-based CLWE methods, supervised and unsupervised alike, solve the *Procrustes* problem in the final step or during self-learning (Ruder et al., 2019). This restricts the transformation to be orthogonal linear mappings.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 2712–2723,
November 16–20, 2020.

However, learning an orthogonal linear mapping inherently assumes that the embedding spaces of different languages exhibit similar geometric structures (*i.e.,* approximately *isomorphic*). Several recent studies have questioned this strong assumption and empirically showed that the isomorphic assumption does not hold in general even for two closely related languages like English and German (Søgaard et al., 2018; Patra et al., 2019).

In this work, we propose LNMAP (**L**atent space **N**on-linear **Map**ping), a novel semi-supervised approach that uses *non-linear* mapping in the latent space to learn CLWE. It uses minimal supervision from a seed dictionary, while leveraging semantic information from the monolingual word embeddings. As shown in Figure 1, LNMAP comprises two *autoencoders*, one for each language. The auto-encoders are first trained independently in a self-supervised way to induce the latent code space of the respective languages. Then, we use a small seed dictionary to learn the non-linear mappings between the two code spaces. To guide our mapping in the latent space, we include two additional constraints: back-translation and original embedding reconstruction. Crucially, our method does not enforce any strong prior constraints like the orthogonality (or isomorphic), rather it gives the model the flexibility to induce the required latent structures such that it is easier for the non-linear mappers to align them in the code space.

In order to demonstrate the effectiveness and robustness of LNMAP, we conduct extensive experiments on bilingual lexicon induction (BLI) with fifteen (15) different language pairs (in both directions) comprising high- and low-resource languages from two different datasets for different sizes of the seed dictionary. Our results show significant improvements for LNMAP over the state-of-the-art in most of the tested scenarios. It is particularly very effective for low-resource languages; for example, using 1K seed dictionary, LNMAP yields about 18% absolute improvements on average over a state-of-the-art supervised method (Joulin et al., 2018). It also outperforms the most robust unsupervised system of Artetxe et al. (2018b) in most of the translation tasks. Interestingly, for resource-rich language pairs, linear autoencoder performs better than non-linear ones. Our ablation study reveals the collaborative nature of LNMAP's different components and efficacy of its non-linear mappings in the code space. We open-source our framework at https://ntunlpsg.github.io/project/lnmap/.

2 Background

Limitations of Isomorphic Assumption. Almost all CLWE methods inherently assume that embedding spaces of different languages are approximately *isomorphic* (i.e., similar in geometric structure). However, recently researchers have questioned this simplified assumption and attributed the performance degradation of existing CLWE methods to the strong mismatches in embedding spaces caused by the linguistic and domain divergences (Søgaard et al., 2019; Ormazabal et al., 2019). Søgaard et al. (2018) empirically show that even closely related languages are far from being isomorphic. Nakashole and Flauger (2018) argue that mapping between embedding spaces of different languages can be approximately linear only at small local regions, but must be non-linear globally. Patra et al. (2019) also recently show that etymologically distant language pairs cannot be aligned properly using orthogonal transformations.

Towards Semi-supervised Methods. A number of recent studies have questioned the robustness of existing unsupervised CLWE methods (Ruder et al., 2019). Vulić et al. (2019) show that even the most robust unsupervised method (Artetxe et al., 2018b) fails for a large number of language pairs; it gives zero (or near zero) BLI performance for 87 out of 210 language pairs. With a seed dictionary of only 500 - 1000 word pairs, their supervised method outperforms unsupervised methods by a wide margin in most language pairs. Other recent work also suggested using semi-supervised methods (Patra et al., 2019; Ormazabal et al., 2019).

Mapping in Latent Space. Mohiuddin and Joty (2019) propose adversarial autoencoder for *unsupervised* word translation. They use *linear* autoencoders in their model, and the mappers are also linear. They emphasize the benefit of using latent space over the original embedding space. Although their method is more robust than other existing adversarial models, still it suffers from training instability for distant language pairs.

Our Contributions. Our proposed LNMAP is independent of the isomorphic assumption. It uses weak supervision from a small seed dictionary, while leveraging rich structural information from monolingual embeddings. Unlike Mohiuddin and Joty (2019), the autoencoders in LNMAP are

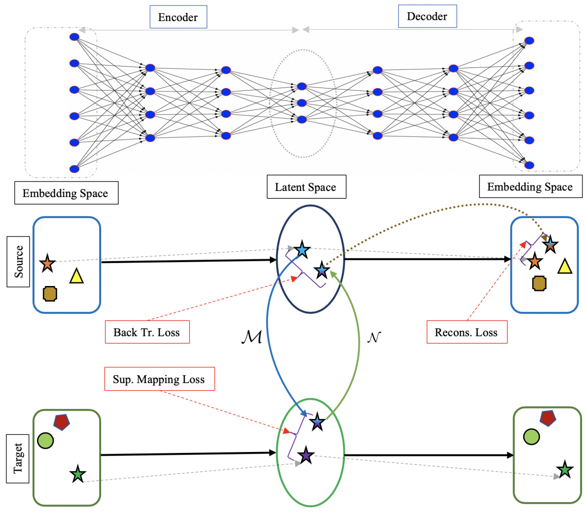

Figure 1: LNMap: Our proposed semi-supervised framework. Identical shapes with different colors denote the similar meaning words in different spaces (e.g., source/target embedding space or latent space).

not limited to only linearity. More importantly, it uses *non-linear* mappers. These two factors contribute to its robust performance even for very low-resource languages (§5). To the best of our knowledge, we are the first to showcase such robust and improved performance with non-linear methods.[1]

3 LNMap Semi-supervised Framework

Let $\mathcal{V}_{\ell_x} = \{v_{x_1}, ..., v_{x_{n_x}}\}$ and $\mathcal{V}_{\ell_y} = \{v_{y_1}, ..., v_{y_{n_y}}\}$ be two sets of vocabulary consisting of n_x and n_y words for a source (ℓ_x) and a target (ℓ_y) language, respectively. Each word v_{x_i} (resp. v_{y_j}) has an embedding $x_i \in \mathbb{R}^d$ (resp. $y_j \in \mathbb{R}^d$), trained with any word embedding models, *e.g.*, FastText (Bojanowski et al., 2017). Let $\mathcal{E}_{\ell_x} \in \mathbb{R}^{n_x \times d}$ and $\mathcal{E}_{\ell_y} \in \mathbb{R}^{n_y \times d}$ be the word embedding matrices for the source and target languages, respectively. We are also given with a seed dictionary $\mathcal{D} = \{(x_1, y_1), ..., (x_k, y_k)\}$ with k word pairs. Our objective is to learn a transformation function $\mathcal{M}$ such that for any $v_{x_i} \in \mathcal{V}_{\ell_x}$, $\mathcal{M}(x_i)$ corresponds to its translation y_j, where $v_{y_j} \in \mathcal{V}_{\ell_y}$. Our approach LNMap (Figure 1) follows two sequential steps:

(i) Unsupervised latent space induction using monolingual autoencoders (§3.1), and

(ii) Supervised non-linear transformation learning with back-translation and source embedding reconstruction constraints (§3.2).

3.1 Unsupervised Latent Space Induction

We use two autoencoders, one for each language. Each autoencoder comprises an encoder E_{ℓ_x} (resp. E_{ℓ_y}) and a decoder D_{ℓ_x} (resp. D_{ℓ_y}). Unless otherwise stated, the autoencoders are *non-linear*, where each of the encoder and decoder is a three-layer feed-forward neural network with two non-linear hidden layers. More formally, the encoding-decoding operations of the source autoencoder ($\texttt{autoenc}_{\ell_x}$) are defined as:

$$h_1^{E_{\ell_x}} = \phi(\theta_1^{E_{\ell_x}} x_i) \quad (1) \qquad h_1^{D_{\ell_x}} = \phi(\theta_3^{D_{\ell_x}} z_{x_i}) \quad (4)$$

$$h_2^{E_{\ell_x}} = \phi(\theta_2^{E_{\ell_x}} h_1^{E_{\ell_x}}) (2) \qquad h_2^{D_{\ell_x}} = \phi(\theta_2^{D_{\ell_x}} h_1^{D_{\ell_x}}) \quad (5)$$

$$z_{x_i} = \theta_3^{E_{\ell_x}} h_2^{\ell_x} \quad (3) \qquad \hat{x}_i = \phi(\theta_1^{D_{\ell_x}} h_2^{D_{\ell_x}}) \quad (6)$$

where $\theta_i^{E_{\ell_x}} \in \mathbb{R}^{c_i \times d_i}$ and $\theta_i^{D_{\ell_x}} \in \mathbb{R}^{d_i \times c_i}$ are the parameters of the layers in the encoder and decoder respectively, and ϕ is a non-linear activation function; we use Parametric Rectified Linear Unit (PReLU) in all the hidden layers and tanh in the final layer of the decoder (Eq. 6). We use linear activations in the output layer of the encoder (Eq. 3). We train $\texttt{autoenc}_{\ell_x}$ with l_2 reconstruction loss as:

$$\mathcal{L}_{\text{autoenc}_{\ell_x}}(\Theta_{E_{\ell_x}}, \Theta_{D_{\ell_x}}) = \frac{1}{n_x} \sum_{i=1}^{n_x} \|x_i - \hat{x}_i\|^2 \quad (7)$$

where $\Theta_{E_{\ell_x}} = \{\theta_1^{E_{\ell_x}}, \theta_2^{E_{\ell_x}}, \theta_3^{E_{\ell_x}}\}$ and $\Theta_{D_{\ell_x}} = \{\theta_1^{D_{\ell_x}}, \theta_2^{D_{\ell_x}}, \theta_3^{D_{\ell_x}}\}$ are the parameters of the encoder and the decoder of $\texttt{autoenc}_{\ell_x}$.

The encoder, decoder and the reconstruction loss for the target autoencoder ($\texttt{autoenc}_{\ell_y}$) are similarly defined.

3.2 Supervised Non-linear Transformation

Let $q(z_x|x)$ and $q(z_y|y)$ be the distributions of latent codes in $\texttt{autoenc}_{\ell_x}$ and $\texttt{autoenc}_{\ell_y}$, respectively. We have two non-linear mappers: $\mathcal{M}$ that translates a source code into a target code, and $\mathcal{N}$ that translates a target code into a source code (Figure 1). Both mappers are implemented as a feed-forward neural network with a single hidden layer and tanh activations, and they are trained using the provided seed dictionary $\mathcal{D}$.

Non-linear Mapping Loss. Let $\Theta_{\mathcal{M}}$ and $\Theta_{\mathcal{N}}$ denote the parameters of the two mappers $\mathcal{M}$ and $\mathcal{N}$, respectively. While mapping from $q(z_x|x)$ to $q(z_y|y)$, we jointly train the mapper $\mathcal{M}$ and the source encoder E_{ℓ_x} with the following l_2 loss.

[1] Our experiments with (unsupervised) adversarial training showed very unstable results with the non-linear mappers.

$$\mathcal{L}_{\text{MAP}}(\Theta_{\mathcal{M}}, \Theta_{E_{\ell_x}}) = \frac{1}{k}\sum_{i=1}^{k} \|z_{y_i} - \mathcal{M}(z_{x_i})\|^2 \quad (8)$$

The mapping loss for $\mathcal{N}$ and E_{ℓ_y} is similarity defined. To learn a better transformation function, we enforce two additional constraints to our objective – back-translation and reconstruction.

Back-Translation Loss. To ensure that a source code $z_{x_i} \in q(z_x|x)$ translated to the target language latent space $q(z_y|y)$, and then translated back to the original latent space remain unchanged, we enforce the back-translation constraint, that is, $z_{x_i} \rightarrow \mathcal{M}(z_{x_i}) \rightarrow \mathcal{N}(\mathcal{M}(z_{x_i})) \approx z_{x_i}$. The back-translation (BT) loss from $q(z_y|y)$ to $q(z_x|x)$ is

$$\mathcal{L}_{\text{BT}}(\Theta_{\mathcal{M}}, \Theta_{\mathcal{N}}) = \frac{1}{k}\sum_{i=1}^{k} \|z_{x_i} - \mathcal{N}(\mathcal{M}(z_{x_i}))\|^2 \quad (9)$$

The BT loss in the other direction $(z_{y_j} \rightarrow \mathcal{N}(z_{y_j}) \rightarrow \mathcal{M}(\mathcal{N}(z_{y_j})) \approx z_{y_j})$ is similarly defined.

Reconstruction Loss. In addition to back-translation, we include another constraint to guide the mapping further. In particular, we ask the decoder D_{ℓ_x} of $\texttt{autoenc}_{\ell_x}$ to reconstruct the original embedding x_i from the back-translated code $\mathcal{N}(\mathcal{M}(z_{x_i}))$. We compute this original embedding reconstruction loss for $\texttt{autoenc}_{\ell_x}$ as:

$$\mathcal{L}_{\text{REC}}(\theta_{E_{\ell_x}}, \theta_{D_{\ell_x}}, \Theta_{\mathcal{M}}, \Theta_{\mathcal{N}}) = \frac{1}{k}\sum_{i=1}^{k} \|x_i - D_{\ell_x}(\mathcal{N}(\mathcal{M}z_{x_i})))\|^2 \quad (10)$$

The reconstruction loss for $\texttt{autoenc}_{\ell_y}$ is defined similarly. Both back-translation and reconstruction lead to more *stable training* in our experiments. In our ablation study (§5.4), we empirically show the efficacy of the addition of these two constraints.

Total Loss. The total loss for mapping a batch of word embeddings from source to target is:

$$\mathcal{L}_{\ell_x \rightarrow \ell_y} = \mathcal{L}_{\text{MAP}} + \lambda_1 \mathcal{L}_{\text{BT}} + \lambda_2 \mathcal{L}_{\text{REC}} \quad (11)$$

where λ_1 and λ_2 control the relative importance of the loss components. Similarly we define the total loss for mapping in the opposite direction $\mathcal{L}_{\ell_y \rightarrow \ell_x}$.

Remark. Note that our approach is fundamentally different from existing methods in two ways. First, most of the existing methods directly map the distribution of the source embeddings $p(x)$ to the distribution of the target embeddings $p(y)$. Second, they learn a linear mapping function assuming that the two languages' embedding spaces are nearly isomorphic, which does not hold in general (Søgaard et al., 2018; Patra et al., 2019).

Mapping the representations in the code space using non-linear transformations gives our model the flexibility to induce the required semantic structures in its latent space that could potentially yield more accurate cross-lingual mappings (§5).

3.3 Training Procedure

We present the training method of LNMAP in Algorithm 1. In the first step, we pre-train $\texttt{autoenc}_{\ell_x}$ and $\texttt{autoenc}_{\ell_y}$ separately on the respective monolingual word embeddings. In this unsupervised step, we use the first 200K embeddings. This pre-training induce word semantics (and relations) in the code space (Mohiuddin and Joty, 2019).

The next step is the self-training process, where we train the mappers along with the autoencoders using the seed dictionary in an iterative manner. We keep a copy of the original dictionary $\mathcal{D}$; let us call it $\mathcal{D}_{\text{orig}}$. We first update the mapper $\mathcal{M}$ and the source encoder E_{ℓ_x} on the mapping loss (Eq. 8). The mappers (both $\mathcal{M}$ and $\mathcal{N}$) then go through two more updates, one for back-translation (Eq. 9) and the other for reconstruction of the source embedding (Eq. 10). The entire source autoencoder $\texttt{autoenc}_{\ell_x}$ (both E_{ℓ_x} and D_{ℓ_x}) in this stage gets updated only on the reconstruction loss.

After each iteration of training (step *i.* in Alg. 1), we induce a new dictionary $\mathcal{D}_{\text{new}}$ using the learned encoders and mappers. To find the nearest target word (y_j) of a source word (x_i) in the target latent space, we use the Cross-domain Similarity Local Scaling (CSLS) measure which works better than simple cosine similarity in mitigating the *hubness* problem (Conneau et al., 2018). It penalizes the words that are close to many other words in the target latent space. To induce the dictionary, we compute CSLS for K most frequent source and target words and select the translation pairs that are nearest neighbors of each other according to CSLS.

For the next iteration of training, we construct the dictionary $\mathcal{D}$ by merging $\mathcal{D}_{\text{orig}}$ with the l most similar (based on CSLS) word pairs from $\mathcal{D}_{\text{new}}$.

Algorithm 1: Training LNMAP

```
Input :Word embedding matrices: $\mathcal{E}_{\ell_x}$, $\mathcal{E}_{\ell_y}$, seed dictionary: $\mathcal{D}$, and increment count $C$
// Unsup.  latent space induction
1. Train autoenc$_{\ell_x}$ and autoenc$_{\ell_y}$ separately for some epochs on monolingual word
   embeddings
// Sup.  non-linear transformation
2. $iter = 0$; $\mathcal{D}_{\text{orig}} = \mathcal{D}$
3. do
   $iter = iter + 1$
   i. for $n\_epochs$ do
       ($a$) Sample a mini-batch from $\mathcal{D}$
       ($b$) Update mapper $\mathcal{M}$ and $E_{\ell_x}$ on the non-linear mapping loss
       ($c$) Update mappers $\mathcal{M}$ and $\mathcal{N}$ on the back-translation loss
       ($d$) Update mappers ($\mathcal{M}, \mathcal{N}$) and autoenc$_{\ell_x}$ on the reconstruction loss
     end
   ii. Induce a new dictionary $\mathcal{D}_{\text{new}}$ of size: $iter \times C$
   iii. Create a new dictionary, $\mathcal{D} = \mathcal{D}_{\text{orig}} \bigcup \mathcal{D}_{\text{new}}$
while not converge;
```

We set l as $l = iter \times C$, where $iter$ is the current iteration number and C is a hyperparameter. This means we incrementally update the dictionary size. This is because the induced dictionary at the initial iterations is likely to be noisy. As the training progresses, the model becomes more mature, and the induced dictionary pairs become better. For convergence, we use the criterion: if the difference between the average similarity scores of two successive iteration steps is less than a threshold (we use $1e^{-6}$), then stop the training process.

4 Experimental Settings

We evaluate our approach on bilingual lexicon induction, also known as *word translation*.

4.1 Datasets

To demonstrate the effectiveness of our method, we evaluate our models against baselines on two popularly used datasets: MUSE (Conneau et al., 2018) and VecMap (Dinu et al., 2015).

The MUSE dataset consists of FastText monolingual embeddings of 300 dimensions (Bojanowski et al., 2017) trained on Wikipedia monolingual corpus and gold dictionaries for 110 language pairs.[2] To show the generality of different methods, we consider 15 different language pairs with $15 \times 2 = 30$ different translation tasks encompassing resource-rich and low-resource languages from different language families. In particular, we evaluate on English (En) from/to Spanish (Es), German (De), Italian (It), Russian (Ru), Arabic (Ar), Malay (Ms), Finnish (Fi), Estonian (Et), Turkish (Tr), Greek (El), Persian (Fa), Hebrew (He), Tamil (Ta), Bengali (Bn), and Hindi (Hi). We differentiate between high- and low-resource languages by the availability of NLP-resources in general.

The VecMap dataset (Dinu et al., 2015; Artetxe et al., 2018a) is a more challenging dataset and contains monolingual embeddings for English, Spanish, German, Italian, and Finnish.[3] According to Artetxe et al. (2018b), existing unsupervised methods often fail to produce meaningful results on this dataset. English, Italian, and German embeddings were trained on WacKy crawling corpora using CBOW (Mikolov et al., 2013b), while Spanish and Finnish embeddings were trained on WMT News Crawl and Common Crawl, respectively.

4.2 Baseline Methods

We compare our proposed LNMAP with several existing methods comprising supervised, semi-supervised, and unsupervised models. For each baseline model, we conduct experiments with the publicly available code. In the following, we give a brief description of the baseline models.

Supervised & Semi-supervised Methods.

(*a*) Artetxe et al. (2017) propose a *self-learning framework* that performs two steps iteratively until

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

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

convergence. In the first step, they use the dictionary (starting with the seed dictionary) to learn a linear mapping, which is then used in the second step to induce a new dictionary.

(*b*) Artetxe et al. (2018a) propose a *multi-step framework* that generalizes previous studies. Their framework consists of several steps: whitening, orthogonal mapping, re-weighting, de-whitening, and dimensionality reduction.

(*c*) Conneau et al. (2018) compare their unsupervised model with a supervised baseline that learns an orthogonal mapping between the embedding spaces by iterative Procrustes refinement. They also propose CSLS for nearest neighbour search.

(*d*) Joulin et al. (2018) show that minimizing a convex relaxation of the CSLS loss significantly improves the quality of bilingual word vector alignment. Their method achieves state-of-the-art results for many languages (Patra et al., 2019).

(*e*) Jawanpuria et al. (2019) propose a geometric approach where they decouple CLWE learning into two steps: (i) learning rotations for language-specific embeddings to align them to a common space, and (ii) learning a similarity metric in the common space to model similarities between the embeddings of the two languages.

(*f*) Patra et al. (2019) propose a semi-supervised technique that relaxes the isomorphic assumption while leveraging both seed dictionary pairs and a larger set of unaligned word embeddings.

Unsupervised Methods.

(*a*) Conneau et al. (2018) are the first to show impressive results for unsupervised word translation by pairing adversarial training with effective refinement methods. Given two monolingual word embeddings, their adversarial training plays a *two-player game*, where a linear mapper (generator) plays against a discriminator. They also impose the orthogonality constraint on the mapper. After adversarial training, they use the iterative Procrustes solution similar to their supervised approach.

(*b*) Artetxe et al. (2018b) learn an initial dictionary by exploiting the structural similarity of the embeddings in an unsupervised way. They propose a robust self-learning to improve it iteratively. This model is by far the most robust and best performing unsupervised model (Vulić et al., 2019).

(*c*) Mohiuddin and Joty (2019) use adversarial autoencoder for unsupervised word translation. They use linear autoencoders in their model, and the mappers are also linear.

4.3 Model Variants and Settings

We experiment with two variants of our model: the default LNMAP that uses non-linear autoencoders and LNMAP (LIN. AE) that uses linear autoencoders. In both the variants, the mappers are non-linear. We train our models using stochastic gradient descent (SGD) with a batch size of 128, a learning rate of $1e^{-4}$, and a step learning rate decay schedule. During the dictionary induction process in each iteration, we consider $K = 15000$ most frequent words from the source and target languages. For dictionary update, we set $C = 2000$.

5 Results and Analysis

We present our results on low-resource and resource-rich languages from MUSE dataset in Tables 1 and 2, respectively, and the results on VecMap dataset in Table 3. We present the results in *precision@1*, which means how many times one of the correct translations of a source word is predicted as the top choice. For each of the cases, we show results on seed dictionary of three different sizes including 1-to-1 and 1-to-many mappings; **"1K Unique"** and **"5K Unique"** contain 1-to-1 mappings of 1000 and 5000 source-target pairs respectively, while **"5K All"** contains 1-to-many mappings of all 5000 source and target words, that is, for each source word there can be multiple target words. Through experiments and analysis, our goal is to assess the following questions.

(i) Does LNMAP improve over the best existing methods in terms of mapping accuracy on low-resource languages (§5.1)?

(ii) How well does LNMAP perform on resource-rich languages (§5.2)?

(iii) What is the effect of non-linearity in the autoencoders? (§5.3)

(iv) Which components of LNMAP attribute to improvements (§5.4)?

5.1 Performance on Low-resource Languages

Most of the unsupervised models fail in the majority of the low-resource languages (Vulić et al., 2019). On the other hand, the performance of supervised models on low-resource languages was not satisfactory, especially with small seed dictionary.

	En-Ms		**En-Fi**		**En-Et**		**En-Tr**		**En-El**		**En-Fa**		**En-He**		**En-Ta**		**En-Bn**		**En-Hi**		**Avg.**
	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←	
GH Distance	0.49		0.54		0.68		0.41		0.46		0.39		0.45		0.47		0.49		0.56		
Unsupervised Baselines																					
Artetxe et al. (2018b)	49.0	49.7	49.8	63.5	33.7	51.2	52.7	63.5	47.6	63.4	33.4	40.7	43.8	57.5	0.0	0.0	18.4	23.9	39.7	48.0	41.5
Conneau et al. (2018)	46.2	0.0	38.4	0.0	19.4	0.0	46.4	0.0	39.5	0.0	30.5	0.0	36.8	53.1	0.0	0.0	0.0	0.0	0.0	0.0	15.5
Mohiuddin and Joty (2019)	54.1	51.7	44.8	62.5	31.8	48.8	51.3	61.7	47.9	63.5	36.7	44.5	44.0	57.1	0.0	0.0	0.0	0.0	0.0	0.0	35.0
Supervision With **“1K Unique”** Seed Dictionary																					
Sup./Semi-sup. Baselines																					
Artetxe et al. (2017)	36.5	41.0	40.8	56.0	21.3	39.0	39.5	56.5	34.5	56.2	24.1	35.7	30.2	51.7	5.4	12.7	6.2	19.9	22.6	38.8	33.5
Artetxe et al. (2018a)	35.3	34.0	30.8	40.8	21.6	32.6	33.7	43.3	32.0	46.4	22.8	27.6	32.27	39.1	7.3	11.9	11.3	15.7	26.2	30.7	28.8
Conneau et al. (2018)	46.2	44.7	46.0	58.4	29.3	40.0	44.8	58.5	42.1	56.5	31.6	38.4	38.3	52.4	11.7	16.0	14.3	19.7	32.5	42.3	38.2
Joulin et al. (2018)	31.4	30.7	30.4	41.4	20.1	26.0	30.7	36.5	28.8	43.6	18.7	23.1	33.5	34.3	6.0	10.1	7.6	11.3	20.7	25.7	25.6
Jawanpuria et al. (2019)	40.0	39.6	37.5	50.7	24.9	38.4	39.7	49.7	36.6	52.9	26.1	33.0	35.1	44.5	10.0	15.9	12.0	19.7	30.5	37.1	33.7
Patra et al. (2019)	40.4	41.4	44.3	59.8	21.0	40.4	41.4	58.8	37.1	58.9	26.5	39.6	38.4	54.1	6.4	15.1	6.1	18.1	24.9	35.4	35.4
LNMap	**50.6**	**49.5**	**52.5**	**62.1**	**38.2**	**49.4**	**52.6**	**62.1**	**48.2**	**58.9**	**35.5**	**40.9**	**46.6**	**52.8**	**17.6**	**21.2**	**18.4**	**27.2**	**37.1**	**47.4**	**43.4**
LNMap (Lin. AE)	49.8	48.7	48.5	61.2	36.5	49.1	49.3	61.9	47.2	58.3	34.7	40.1	43.0	52.3	14.5	20.3	16.5	26.1	35.6	46.6	42.1
Supervision With **“5K Unique”** Seed Dictionary																					
Sup./Semi-sup. Baselines																					
Artetxe et al. (2017)	36.5	42.0	40.8	57.0	22.4	39.6	39.6	56.7	37.2	56.4	26.0	35.3	31.6	51.9	6.2	13.4	8.2	21.3	23.2	38.3	34.2
Artetxe et al. (2018a)	54.6	52.5	48.8	65.2	38.2	54.8	52.0	65.1	47.5	64.6	38.4	42.4	47.4	**57.4**	18.4	25.8	21.9	31.8	**40.3**	49.5	45.8
Conneau et al. (2018)	46.4	45.7	46.0	59.2	31.0	41.7	45.9	60.1	43.1	56.8	31.6	37.7	38.4	53.4	14.3	19.1	15.0	22.6	32.9	42.8	39.2
Joulin et al. (2018)	50.0	49.3	**53.0**	66.1	39.8	52.0	**54.0**	61.7	47.6	63.4	**39.6**	42.2	**53.0**	56.3	16.0	24.2	21.3	27.0	38.3	47.5	45.2
Jawanpuria et al. (2019)	51.0	49.8	47.4	65.1	36.0	49.8	49.3	63.9	46.6	62.3	36.6	40.8	44.1	56.1	16.1	23.2	18.6	25.9	37.5	45.9	43.3
Patra et al. (2019)	46.0	46.7	48.6	60.9	33.1	47.2	48.3	61.0	44.2	60..9	34.4	40.7	43.5	56.5	15.3	22.0	15.2	25.0	34.7	43.5	41.4
LNMap	**51.3**	**54.2**	52.7	**67.9**	**40.2**	**56.4**	53.1	**65.5**	**48.2**	**64.8**	36.2	**44.4**	47.5	56.6	**19.7**	**31.5**	**22.0**	**36.2**	38.5	**52.2**	**46.9**
LNMap (Lin. AE)	50.1	53.9	51.3	67.0	38.6	55.6	51.1	64.9	47.7	63.6	35.6	44.0	44.2	55.9	18.6	27.3	19.6	31.6	36.5	51.3	45.4
Supervision With **“5K All”** (“5K Unique” Source Words) Seed Dictionary																					
Sup./Semi-sup. Baselines																					
Artetxe et al. (2017)	37.0	41.6	40.8	57.0	22.7	39.5	38.8	56.9	37.5	57.2	25.4	36.3	32.2	52.1	5.9	14.1	7.7	21.7	22.4	38.3	34.3
Artetxe et al. (2018a)	55.2	51.7	48.9	64.6	37.4	54.0	52.2	63.7	48.2	65.0	39.0	42.6	47.6	58.0	**19.6**	25.2	21.1	30.6	**40.4**	50.0	45.8
Conneau et al. (2018)	46.3	44.8	46.4	59.0	30.9	42.0	45.8	59.0	44.4	57.4	31.8	38.8	39.0	53.4	15.1	18.4	15.5	22.4	32.9	44.4	39.4
Joulin et al. (2018)	**51.4**	49.1	**55.6**	65.8	40.0	50.2	**53.8**	61.7	**49.1**	62.8	**40.5**	42.4	**52.2**	57.9	17.7	24.0	20.2	26.9	38.2	47.1	45.3
Jawanpuria et al. (2019)	**51.4**	47.7	46.7	63.4	33.7	48.7	48.6	61.9	46.3	61.8	38.0	40.9	43.1	56.7	16.5	23.1	19.3	25.6	37.7	44.1	42.8
Patra et al. (2019)	48.4	43.8	53.2	63.8	36.3	48.3	51.8	59.6	48.2	61.8	38.4	39.3	51.6	55.2	16.5	22.7	17.5	26.7	36.2	45.4	43.3
LNMap	50.3	**54.1**	53.1	**70.5**	**41.2**	**57.5**	52.5	**65.3**	**49.1**	**66.6**	36.8	**43.7**	47.6	**59.2**	18.9	**32.1**	**21.4**	**35.2**	37.6	**51.6**	**47.2**
LNMap (Lin. AE)	50.0	53.2	51.2	67.5	39.9	54.5	50.9	64.2	48.6	66.1	36.4	42.9	44.6	59.0	18.0	28.7	20.1	30.8	37.1	50.5	46.7

Table 1: Word translation accuracy (P@1) on **low-resource** languages on **MUSE dataset** using fastText.

Hence, we first compare LNMap's performance on these languages. From Table 1, we see that on average LNMap outperforms every baseline by a good margin (1.1% - 5.2% from the best baselines).

For "1K Unique" dictionary, LNMap exhibits impressive performance. In all the 20 translation tasks, it outperforms all the (semi-)supervised baselines by a wide margin. If we compare with Joulin et al. (2018), a state-of-the-art supervised model, LNMap's average improvement is ∼18%, which is remarkable. Compared to other baselines, the average margin of improvement is also quite high – $9.9\%, 14.6\%, 5.2\%, 9.7\%$, and 8.0% gains over Artetxe et al. (2017), Artetxe et al. (2018a), Conneau et al. (2018), Jawanpuria et al. (2019), and Patra et al. (2019), respectively. We see that among the supervised baselines, Conneau et al. (2018)'s model performs better than others.

If we increase the dictionary size, we can still see the dominance of LNMap over the baselines. For "5K Unique" seed dictionary, it performs better than the baselines on 14/20 translation tasks, while for "5K All" seed dictionary, the best performance by LNMap is on 13/20 translation tasks.

One interesting thing to observe is that, under resource-constrained setup LNMap's performance is impressive, making it suitable for very low-resource languages like En-Ta, En-Bn, and En-Hi.

Now if we look at the performance of unsupervised baselines on low-resource languages, we see that Conneau et al. (2018)'s model fails to converge on the majority of the translation tasks (12/20), while the model of Mohiuddin and Joty (2019) fails to converge on En↔Ta, En↔Bn, and En↔Hi. Although the most robust unsupervised method of Artetxe et al. (2018b) performs better than the other unsupervised approaches, it still fails to converge on En↔Ta tasks. If we compare its performance with LNMap, we see that our model outperforms the best unsupervised model of Artetxe et al. (2018b) on 18/20 low-resource translation tasks.

5.2 Results on Resource-rich Languages

Table 2 shows the results for 5 resource-rich language pairs (10 translation tasks) from the MUSE dataset. We notice that our model achieves the highest accuracy in all the tasks for "1K Unique", 4 tasks for "5K Unique", 3 for "5K All".

We show the results on the VecMap dataset in Table 3, where there are 3 resource-rich language

	En-Es		En-De		En-It		En-Ar		En-Ru		Avg.
	→	←	→	←	→	←	→	←	→	←	
GH Distance	0.21		0.31		0.19		0.46		0.46		
Unsupervised Baselines											
Artetxe et al. (2018b)	82.2	84.4	74.9	74.1	78.9	79.5	33.2	52.8	48.93	65.0	67.4
Conneau et al. (2018)	81.8	83.7	74.2	72.6	78.3	78.1	29.3	47.6	41.9	59.0	64.7
Mohiuddin and Joty (2019)	82.7	84.7	75.4	74.3	79.0	79.6	36.3	52.6	46.9	64.7	67.6
Supervision With **“1K Unique”** Seed Dictionary											
Sup./Semi-sup. Baselines											
Artetxe et al. (2017)	81.0	83.6	73.8	72.4	76.6	77.8	24.9	44.9	46.3	61.7	64.3
Artetxe et al. (2018a)	73.8	76.6	62.5	57.6	67.9	70.0	25.8	37.3	40.2	49.5	56.2
Conneau et al. (2018)	81.2	82.8	73.6	73.0	77.6	76.6	34.7	46.4	48.5	60.6	65.5
Joulin et al. (2018)	70.8	74.1	59.0	54.0	62.7	67.2	22.4	32.2	39.6	45.4	52.8
Jawanpuria et al. (2019)	75.1	77.3	66.0	62.6	69.3	71.6	28.4	40.6	41.7	53.9	58.6
Patra et al. (2019)	81.9	83.8	74.6	73.1	78.0	78.1	29.8	50.9	46.3	63.6	66.0
LNMap	80.1	80.2	73.3	71.8	77.1	75.2	**40.5**	52.2	49.9	62.1	66.2
LNMap (Lin. AE)	**83.2**	**85.5**	**76.2**	**74.9**	**79.2**	**79.6**	37.7	**54.0**	**52.6**	**66.2**	**68.8**
Supervision With **“5K Unique”** Seed Dictionary											
Sup./Semi-sup. Baselines											
Artetxe et al. (2017)	81.3	83.3	72.8	72.6	76.3	77.6	24.1	45.3	47.5	60.3	64.1
Artetxe et al. (2018a)	80.8	84.5	73.3	74.3	77.4	79.7	42.0	54.7	51.5	68.2	68.7
Conneau et al. (2018)	81.6	83.5	74.1	72.7	77.8	77.2	34.3	48.5	49.0	60.7	66.0
Joulin et al. (2018)	**83.4**	85.4	**77.0**	**76.4**	78.7	**81.6**	**41.3**	54.0	**58.1**	67.4	**70.4**
Jawanpuria et al. (2019)	81.3	**86.3**	74.5	75.9	78.6	81.3	38.7	53.4	52.3	67.6	68.9
Patra et al. (2019)	82.2	84.6	75.6	73.7	77.8	78.6	35.0	51.9	52.2	65.2	69.5
LNMap	80.9	80.8	74.9	72.3	77.1	76.5	40.7	56.6	52.2	64.8	67.7
LNMap (Lin. AE)	**83.4**	85.7	75.5	75.4	**79.0**	81.1	39.5	**56.8**	53.8	**68.4**	69.9
Supervision With **“5K All”**(5K Unique Source Words) Seed Dictionary											
Sup./Semi-sup. Baselines											
Artetxe et al. (2017)	81.2	83.5	72.8	72.5	76.0	77.5	24.4	45.3	47.3	61.2	64.2
Artetxe et al. (2018a)	80.5	83.8	73.5	73.5	77.1	79.2	41.2	55.5	50.5	67.3	68.2
Conneau et al. (2018)	81.6	83.2	73.7	72.6	77.3	77.0	34.1	49.4	49.8	60.7	66.0
Joulin et al. (2018)	**84.4**	**86.4**	**79.0**	76.0	79.0	81.4	**42.2**	55.5	**57.4**	67.0	**70.9**
Jawanpuria et al. (2019)	81.4	85.5	74.7	**76.7**	77.8	80.9	38.1	53.3	51.1	67.6	68.7
Patra et al. (2019)	84.0	**86.4**	78.7	76.4	**79.3**	**82.4**	41.1	53.9	57.2	64.8	70.4
LNMap	80.5	82.2	73.9	72.7	76.7	78.3	41.5	57.1	53.5	67.1	68.4
LNMap (Lin. AE)	82.9	**86.4**	75.5	75.9	78.1	81.4	39.3	**57.3**	52.3	**67.8**	69.6

Table 2: Word translation accuracy (P@1) on **resource-rich** languages on **MUSE dataset** using fastText.

pairs, and one low-resource pair (En-Fi) with a total of 8 translation tasks. Overall, we have similar observations as in MUSE – our model outperforms other models on 7 tasks for “1K Unique”, 4 tasks for “5K Unique”, and 4 for “5K All”.

5.3 Effect of Non-linearity in Autoencoders

The comparative results between our model variants in Tables 1 - 3 reveal that LNMap (with non-linear autoencoders) works better for low-resource languages, whereas LNMap (Lin. AE) works better for resource-rich languages. This can be explained by the geometric similarity between the embedding spaces of the two languages.

In particular, we measure the geometric similarity of the language pairs using the **Gromov-Hausdorff (GH)** distance (Patra et al., 2019), which is recently proposed to quantitatively estimate isometry between two embedding spaces.[4] From the measurements (Tables 1-2), we see that etymologically close language pairs have lower GH distance compared to etymologically distant and low-resource language pairs.[5] Low-resource language pairs' high GH distance measure implies that English and those languages embedding spaces are far from isomorphism. Hence, we need strong non-linearity for those distant languages.

5.4 Dissecting LNMap

We further analyze our model by dissecting it and measuring the contribution of its different components. Specifically, our goal is to assess the contribution of back-translation, reconstruction, non-linearity in the mapper, and non-linearity in the autoencoder. We present the ablation results in Table 4 on 8 translation tasks from 4 language pairs consisting of 2 resource-rich and 2 low-resource languages. We use MUSE dataset for this purpose.

[4] https://github.com/joelmoniz/BLISS

[5] We could not compute GH distances for the VecMap dataset; the metric gives ‘inf’ in the BLISS framework.

	En-Es		En-It		En-De		En-Fi		Avg.
	→	←	→	←	→	←	→	←	
Unsupervised Baselines									
Artetxe et al. (2018b)	36.9	31.6	47.9	42.3	48.3	44.1	32.9	33.5	39.7
Conneau et al. (2018)	34.7	0.0	44.9	38.7	0.0	0.0	0.0	0.0	14.8
Mohiuddin and Joty (2019)	37.4	31.9	47.6	42.5	0.0	0.0	0.0	0.0	19.9
Supervision With **“1K Unique”** Seed Dictionary									
Sup./Semi-sup. Baselines									
Artetxe et al. (2017)	33.3	27.7	43.9	38.1	46.8	40.8	30.4	26.0	35.9
Artetxe et al. (2018a)	29.0	20.0	38.6	29.2	36.3	26.0	25.8	15.0	27.5
Conneau et al. (2018)	35.7	30.8	45.4	38.3	46.9	42.3	29.1	27.2	37.0
Joulin et al. (2018)	24.2	17.9	33.9	25.1	31.6	25.5	21.9	14.5	24.4
Jawanpuria et al. (2019)	31.5	23.2	39.2	32.4	39.1	30.9	26.8	21.4	30.6
Patra et al. (2019)	31.4	30.5	30.9	38.8	**47.9**	43.7	30.5	31.6	35.7
LNMAP	32.9	28.6	44.2	39.1	43.0	39.2	26.6	25.4	34.9
LNMAP (LIN. AE)	**36.5**	**33.6**	**46.0**	**40.1**	46.4	**44.8**	**31.7**	**37.1**	**39.5**
Supervision With **“5K Unique”** Seed Dictionary									
Sup./Semi-sup. Baselines									
Artetxe et al. (2017)	33.3	27.6	43.9	38.4	46.0	41.1	30.9	25.7	35.9
Artetxe et al. (2018a)	**37.6**	34.0	45.7	**41.6**	47.2	45.0	34.0	**38.8**	40.2
Conneau et al. (2018)	36.0	31.1	46.0	38.8	47.6	43.2	31.1	28.2	37.8
Joulin et al. (2018)	34.2	31.1	43.1	37.2	44.5	41.9	30.9	34.7	37.2
Jawanpuria et al. (2019)	36.9	33.3	**47.1**	39.9	47.7	44.6	**35.1**	38.0	40.2
Patra et al. (2019)	34.3	31.6	41.1	39.3	47.5	43.6	30.7	33.4	37.7
LNMAP	33.4	27.3	44.1	38.9	42.5	39.4	29.7	28.6	35.5
LNMAP (LIN. AE)	37.1	**34.1**	46.2	40.3	**47.7**	**45.6**	33.3	**38.8**	**40.3**
Supervision With **“5K All”** (5K Unique Source Words) Seed Dictionary									
Sup./Semi-sup. Baselines									
Artetxe et al. (2017)	32.7	28.1	43.8	38.0	47.4	40.8	30.8	26.2	36.0
Artetxe et al. (2018a)	38.2	33.4	47.3	**41.6**	47.2	44.8	**34.9**	38.6	**40.8**
Conneau et al. (2018)	36.1	31.2	45.7	38.5	47.2	42.8	31.2	28.3	37.7
Joulin et al. (2018)	35.5	31.2	44.6	37.6	46.6	41.7	32.1	34.4	38.0
Jawanpuria et al. (2019)	37.5	33.1	**47.6**	40.1	**48.8**	45.1	34.6	37.7	40.6
Patra et al. (2019)	34.5	32.1	46.2	39.5	48.1	44.1	31.0	33.6	39.4
LNMAP	33.7	27.9	43.7	38.9	43.6	39.2	29.9	31.5	36.1
LNMAP (LIN. AE)	**37.8**	**34.6**	46.7	40.2	47.7	**45.2**	34.1	**38.9**	40.6

Table 3: Word translation accuracy (P@1) on **VecMap dataset** using CBOW embeddings.

	Resource-rich				Low-Resource			
	En-Es		En-It		En-Ta		En-Bn	
	→	←	→	←	→	←	→	←
LNMAP	80.1	80.2	77.1	75.3	17.6	21.2	18.4	27.2
⊖ Recon. loss	79.6	75.4	75.7	69.4	14.8	14.9	16.2	20.7
⊖ Back-tran. loss	79.8	79.1	76.6	74.4	16.7	20.3	16.5	26.7
⊕ Linear mapper	78.8	78.9	76.3	74.7	16.6	20.2	18.0	26.3
⊕ Procrustes sol.	75.9	73.9	72.0	72.2	11.1	12.1	12.2	14.8
⊕ Linear autoenc.	83.2	85.5	79.2	79.6	14.5	20.3	16.5	26.1

Table 4: Ablation study of LNMAP with “1K Unique” dictionary. ⊖ indicates the component is removed from the full model, and ‘⊕’ indicates the component is added by replacing the corresponding component.

All the experiments for the ablation study are done using “1K Unique” seed dictionary.

⊖ **Reconstruction loss:** For removing the reconstruction loss from the full model, on average high-resource language pairs lose accuracy by 0.9% and 5.3% for from and to English, respectively. The losses are even higher for low-resource language pairs, on average 2.5% and 6.4% in accuracy.

⊖ **Back-translation (BT) loss:** Removing the BT loss also has a negative impact, but not as high as the reconstruction. This is because the reconstruction loss (Eq. 10) also covers the BT signal.

⊕ **Linear mapper:** If we replace the non-linear mapper with a linear one in the full model, we see that the effect is not that severe. The reason can be explained by the fact that the autoencoders are still non-linear, and the non-linear signal passes through back-translation and reconstruction.

⊕ **Procrustes solution:** To assess the proper effect of the non-linear mapper, we need to replace it with a linear mapper through which no non-linear signal passes by during training. This can be achieved by replacing the non-linear mapper with the Procrustes solution. The results show an adverse effect on removing non-linearity in the mapper in all the language pairs. However, low-resource pairs' performance drops quite significantly.

⊕ **Linear autoencoder:** For high-resource language pairs, linear autoencoder works better than the non-linear one. However, it is the opposite for the low-resource pairs, where the performance drops significantly for the linear autoencoder.

6 Conclusions

We have presented a novel semi-supervised framework LNMAP to learn the cross-lingual mapping between two monolingual word embeddings. Apart from exploiting weak supervision from a small (1K) seed dictionary, our LNMAP leverages the information from monolingual word embeddings. In contrast to the existing methods that directly map word embeddings using the isomorphic assumption, our framework is independent of any such strong prior assumptions. LNMAP first learns to transform the embeddings into a latent space and then uses a non-linear transformation to learn the mapping. To guide the non-linear mapping further, we include constraints for back-translation and original embedding reconstruction.

Extensive experiments with fifteen different language pairs comprising high- and low-resource languages show the efficacy of non-linear transformations, especially for low-resource and distant languages. Comparison with existing supervised, semi-supervised, and unsupervised baselines show that LNMAP learns a better mapping. With an in-depth ablation study, we show that different components of LNMAP works in a collaborative nature.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. Shafiq Joty would like to thank the funding support from NRF (NRF2016IDM-TRANS001-062), Singapore.

References

David Alvarez-Melis and Tommi Jaakkola. 2018. Gromov-wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1881–1890. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294, Austin, Texas. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. 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. Association for Computational Linguistics.

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

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018b. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *ACL*.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018c. Unsupervised neural machine translation. In *Proceedings of the Sixth International Conference on Learning Representations*.

Antonio Valerio Miceli Barone. 2016. 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.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations (ICLR)*.

Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. 2015. Improving zero-shot learning by mitigating the hubness problem. In *ICLR, Workshop track*.

Yerai Doval, José Camacho-Collados, Luis Espinosa Anke, and Steven Schockaert. 2019. On the robustness of unsupervised and semi-supervised cross-lingual word embedding learning. *ArXiv*, abs/1908.07742.

Geert Heyman, Ivan Vulić, and Marie-Francine Moens. 2017. Bilingual lexicon induction by learning to combine word-level and character-level representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1085–1095, Valencia, Spain. Association for Computational Linguistics.

Yedid Hoshen and Lior Wolf. 2018. Non-adversarial unsupervised word translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 469–478. Association for Computational Linguistics.

Pratik Jawanpuria, Arjun Balgovind, Anoop Kunchukuttan, and Bamdev Mishra. 2019. Learning multilingual word embeddings in latent metric space: a geometric approach. *Transaction of the Association for Computational Linguistics (TACL)*, 7:107–120.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. 2018. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2984, Brussels, Belgium. Association for Computational Linguistics.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium. Association for Computational Linguistics.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Tasnim Mohiuddin and Shafiq Joty. 2019. Revisiting adversarial autoencoder for unsupervised word translation with cycle consistency and improved training. 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 3857–3867, Minneapolis, Minnesota. Association for Computational Linguistics.

Tasnim Mohiuddin and Shafiq Joty. 2020. Unsupervised word translation with adversarial autoencoder. *Computational Linguistics*, 46(2):257–288.

Ndapa Nakashole and Raphael Flauger. 2018. Characterizing departures from linearity in word translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 221–227, Melbourne, Australia. 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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4990–4995, Florence, Italy. Association for Computational Linguistics.

Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R. Gormley, and Graham Neubig. 2019. Bilingual lexicon induction with semi-supervision in non-isometric embedding spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 184–193, Florence, Italy. Association for Computational Linguistics.

Sebastian Ruder, Anders Søgaard, and Ivan Vulić. 2019. Unsupervised cross-lingual representation learning. In *Proceedings of ACL 2019, Tutorial Abstracts*, pages 31–38.

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 *International Conference on Learning Representations (ICLR)*.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. 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.

Anders Søgaard, Ivan Vulić, Sebastian Ruder, and Manaal Faruqui. 2019. *Cross-Lingual Word Embeddings*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool.

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. Association for Computational Linguistics.

Ivan Vulić and Marie-Francine Moens. 2015. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 363–372, New York, NY, USA. ACM.

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

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970. Association for Computational Linguistics.

A Appendix

A.1 Reproducibility Settings

- Computing infrastructure - Linux machine with a single GTX 1080 Ti GPU
- PyTorch version 1.2.0
- CUDA version 10.0
- cuDNN version 7.6.0
- Average runtime - 15-20 minutes

A.2 Optimal Hyperparameters

Hyperparameter	Value
<u>Encoder</u>	
#layers	3
input dim	300
hidden dim	350-400
output dim	350-400
hidden non-linearity	`PReLU`
output non-linearity	`linear`
<u>Decoder</u>	
#layers	3
input dim	350-400
hidden dim	350-400
output dim	300
hidden non-linearity	`PReLU`
output non-linearity	`tanh`

Table 5: Optimal hyper-parameter settings for autoencoder.

Hyperparameter	**Value**
type	`linear/non-linear`
#layers	2
input dim	350-400
hidden dim	400
output dim	350-400
hidden non-linearity	`tanh`
output non-linearity	`linear`

Table 6: Optimal hyper-parameter settings for mapper.

Hyperparameter	**Value**
normalization	`renorm,center,renorm`
#iterations	`dynamic`
sup. dict size	1K-5K
batch size	128
autoenc. epochs	25
mapper epochs	100
nearest-neighbor	`CSLS`
autoenc. optimizer	`SGD`
autoenc. learning-rate	0.0001
mapper optimizer	`SGD`
mapper learning-rate	0.0001
mapping-loss weight	1.0
cycle-loss weight	1.0
recons.-loss weight	1.0

Table 7: Optimal hyper-parameter settings for LN-MAP training.