Revisiting Adversarial Autoencoder for Unsupervised Word Translation with Cycle Consistency and Improved Training

Tasnim Mohiuddin[⋆] and **Shafiq Joty**[⋆†]
[⋆]Nanyang Technological University, Singapore
[†]Salesforce Research Asia, Singapore
{mohi0004,srjoty}@ntu.edu.sg

Abstract

Adversarial training has shown impressive success in learning bilingual dictionary without any parallel data by mapping monolingual embeddings to a shared space. However, recent work has shown superior performance for non-adversarial methods in more challenging language pairs. In this work, we revisit adversarial autoencoder for unsupervised word translation and propose two novel extensions to it that yield more stable training and improved results. Our method includes regularization terms to enforce cycle consistency and input reconstruction, and puts the target encoders as an adversary against the corresponding discriminator. Extensive experimentations with European, non-European and low-resource languages show that our method is more robust and achieves better performance than recently proposed adversarial and non-adversarial approaches.

1 Introduction

Learning cross-lingual word embeddings has been shown to be an effective way to transfer knowledge from one language to another for many key linguistic tasks including machine translation, named entity recognition, part-of-speech tagging, and parsing (Ruder et al., 2017). While earlier efforts solved the associated word alignment problem using large parallel corpora (Luong et al., 2015), broader applicability demands methods to relax this requirement since acquiring a large corpus of parallel data is not feasible in most scenarios. Recent methods instead use embeddings learned from monolingual data, and learn a linear mapping from one language to another with the underlying assumption that two embedding spaces exhibit similar geometric structures (*i.e.,* approximately *isomorphic*). This allows the model to learn effective cross-lingual representations without expensive supervision (Artetxe et al., 2017).

Given monolingual word embeddings of two languages, Mikolov et al. (2013a) show that a linear mapping can be learned from a seed dictionary of 5000 word pairs by minimizing the sum of squared Euclidean distances between the mapped vectors and the target vectors. Subsequent works (Xing et al., 2015; Artetxe et al., 2016, 2017; Smith et al., 2017) propose to improve the model by normalizing the embeddings, imposing an orthogonality constraint on the mapper, and modifying the objective function. While these methods assume some supervision in the form of a seed dictionary, recently fully unsupervised methods have shown competitive results. Zhang et al. (2017a,b) first reported encouraging results with *adversarial training*. Conneau et al. (2018) improved this approach with post-mapping refinements, showing impressive results for several language pairs. Their learned mapping was then successfully used to train a fully unsupervised neural machine translation system (Lample et al., 2018a,b).

Although successful, adversarial training has been criticized for not being stable and failing to converge, inspiring researchers to propose *non-adversarial* methods more recently (Xu et al., 2018a; Hoshen and Wolf, 2018; Alvarez-Melis and Jaakkola, 2018; Artetxe et al., 2018b). In particular, Artetxe et al. (2018b) show that the adversarial methods of Conneau et al. (2018) and Zhang et al. (2017a,b) fail for many language pairs.

In this paper, we revisit adversarial training and propose a number of key improvements that yield more robust training and improved mappings. Our main idea is to learn the cross-lingual mapping in a projected latent space and add more constraints to guide the unsupervised mapping in this space. We accomplish this by proposing a novel *adversarial autoencoder* framework (Makhzani et al., 2015), where adversarial mapping is done at the (latent) code space as opposed to the original embedding space (Figure 1). This gives the model the flexibility to automatically induce the required geometric structures in its latent code space that could potentially yield better mappings. Søgaard et al. (2018) recently find that the *isomorphic* assumption made by most existing methods does not hold in general even for two closely related languages like English and German. In their words *"approaches based on this assumption have important limitations"*. By mapping the latent vectors through adversarial training, our approach therefore departs from the isomorphic assumption.

In our adversarial training, not only the mapper but also the target encoder is trained to fool the discriminator. This forces the discriminator to improve its discrimination skills, which in turn pushes the mapper to generate indistinguishable translation. To guide the mapping, we include two additional constraints. Our first constraint enforces *cycle consistency* so that code vectors after being translated from one language to another, and then translated back to their source space remain close to the original vectors. The second constraint ensures reconstruction of the original input word embeddings from the back-translated codes. This grounding step forces the model to retain word semantics during the mapping process.

We conduct a series of experiments with six different language pairs (in both directions) comprising European, non-European, and low-resource languages from two different datasets. Our results show that our model is more robust and yields significant gains over Conneau et al. (2018) for all translation tasks in all evaluation measures. Our method also gives better initial mapping compared to other existing methods (Artetxe et al., 2018b). We also perform an extensive ablation study to understand the contribution of different components of our model. The study reveals that cycle consistency contributes the most, while adversarial training of the target encoder and post-cycle reconstruction also have significant effect. We have released our source code at `https://ntunlpsg.github.io/project/unsup-word-translation/`

The remainder of this paper is organized as follows. After discussing related work in Section 2, we present our unsupervised word translation approach with adversarial autoencoder in Section 3. We describe our experimental setup in Section 4, and present our results with in-depth analysis in Section 5. Finally, we summarize our findings with possible future directions in Section 6.

2 Related Work

In recent years a number of methods have been proposed to learn bilingual dictionary from monolingual word embeddings.[1] Many of these methods use an initial seed dictionary. Mikolov et al. (2013a) show that a linear transformation can be learned from a seed dictionary of 5000 pairs by minimizing the squared Euclidean distance. In their view, the key reason behind the good performance of their model is the similarity of geometric arrangements in vector spaces of the embeddings of different languages. For translating a new source word, they map the corresponding word embedding to the target space using the learned mapping and find the nearest target word. In their approach, they found that simple linear mapping works better than non-linear mappings with multilayer neural networks.

Xing et al. (2015) enforce the word vectors to be of unit length during the learning of the embeddings and modify the objective function for learning the mapping to maximize the cosine similarity instead of using Euclidean distance. To preserve length normalization after mapping, they enforce the *orthogonality* constraint on the mapper. Instead of learning a mapping from the source to the target embedding space, Faruqui and Dyer (2014) use a technique based on Canonical Correlation Analysis (CCA) to project both source and target embeddings to a common low-dimensional space, where the correlation of the word pairs in the seed dictionary is maximized. Artetxe et al. (2016) show that the above methods are variants of the same core optimization objective and propose a closed form solution for the mapper under orthogonality constraint. Smith et al. (2017) find that this solution is closely related to the orthogonal *Procrustes* solution. In their follow-up work, Artetxe et al. (2017) obtain competitive results using a seed dictionary of only 25 word pairs. They propose a *self-learning framework* that performs two steps iteratively until convergence. In the first step, they use the dictionary (starting with the seed) to learn a linear mapping, which is then used in the second step to induce a new dictionary.

A more recent line of research attempts to eliminate the seed dictionary totally and learn the map-

[1]see (Ruder et al., 2017) for a nice survey

ping in a purely unsupervised way. This was first proposed by Miceli Barone (2016), who initially used an adversarial network similar to Conneau et al. (2018), and found that the mapper (which is also the encoder) translates everything to a single embedding, known commonly as the *mode collapse* issue (Goodfellow, 2017). To preserve diversity in mapping, he used a decoder to reconstruct the source embedding from the *mapped embedding*, extending the framework to an adversarial autoencoder. His preliminary qualitative analysis shows encouraging results but not competitive with methods using bilingual seeds. He suspected issues with training and with the isomorphic assumption. In our work, we successfully address these issues with an improved model that also relaxes the isomorphic assumption. Our model uses two separate autoencoders, one for each language, which allows us to put more constraints to guide the mapping. We also distinguish the role of an encoder from the role of a mapper. The encoder projects embeddings to latent code vectors, which are then translated by the mapper.

Zhang et al. (2017a) improved adversarial training with orthogonal parameterization and cycle consistency. To aid training, they incorporate additional techniques like noise injection which works as a regularizer. For selecting the best model, they rely on sharp drops of the discriminator accuracy. In their follow-up work (Zhang et al., 2017b), they minimize Earth-Mover's distance between the distribution of the transformed source embeddings and the distribution of the target embeddings. Conneau et al. (2018) show impressive results with adversarial training and refinement with the Procrustes solution. Instead of using the adversarial loss, Xu et al. (2018a) use Sinkhorn distance and adopt cycle consistency inspired by the CycleGAN (Zhu et al., 2017). We also incorporate cycle consistency along with the adversarial loss. However, while all these methods learn the mapping in the original embedding space, our approach learns it in the latent code space considering both the mapper and the target encoder as adversary. In addition, we use a post-cycle reconstruction to guide the mapping.

A number of non-adversarial methods have also been proposed recently. Artetxe et al. (2018b) learn an initial dictionary by exploiting the structural similarity of the embeddings and use a robust self-learning algorithm to improve it iteratively. Hoshen and Wolf (2018) align the second moment of word distributions of the two languages using principal component analysis (PCA) and then refine the alignment iteratively using a variation of the Iterative Closest Point (ICP) method used in computer vision. Alvarez-Melis and Jaakkola (2018) cast the problem as an optimal transport problem and exploit the Gromov-Wasserstein distance which measures how similarities between pairs of words relate across languages.

3 Approach

Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ and $\mathcal{Y} = \{y_1, \ldots, y_m\}$ be two sets consisting of n and m word embeddings of d-dimensions for a source and a target language, respectively. We assume that $\mathcal{X}$ and $\mathcal{Y}$ are trained independently from monolingual corpora. Our aim is to learn a mapping $f(x)$ in an unsupervised way (*i.e.,* no bilingual dictionary given) such that for every x_i, $f(x)$ corresponds to its translation in $\mathcal{Y}$. Our overall approach follows the same sequence of steps as Conneau et al. (2018):

(i) Induction of seed dictionary through adversarial training.
(ii) Iterative refinement of the initial mapping through the Procrustes solution.
(iii) Apply CSLS for nearest neighbor search.

We propose a novel adversarial autoencoder model to learn the initial mapping for inducing a seed dictionary in step (i), and we adopt existing refinement methods for steps (ii) and (iii).

3.1 Adversarial Autoencoder for Initial Dictionary Induction

Our proposed model (Figure 1) has two **autoencoders**, one for each language. Each autoencoder comprises an encoder $E_{\mathcal{X}}$ (res. $E_{\mathcal{Y}}$) and a decoder $D_{\mathcal{X}}$ (res. $D_{\mathcal{Y}}$). The encoders transform an input x (res. y) into a latent code z_x (res. z_y) from which the decoders try to reconstruct the original input. We use a linear encoder and l_2 **reconstruction loss**

$$z_{x_i} = \theta_{E_{\mathcal{X}}} x_i; \qquad \hat{x}_i = \theta_{D_{\mathcal{X}}} z_{x_i} \quad (1)$$

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

where $\theta_{E_{\mathcal{X}}} \in \mathbb{R}^{c \times d}$ and $\theta_{D_{\mathcal{X}}} \in \mathbb{R}^{d \times c}$ are the parameters of the encoder and the decoder for d-dimensional word embedding and c-dimensional

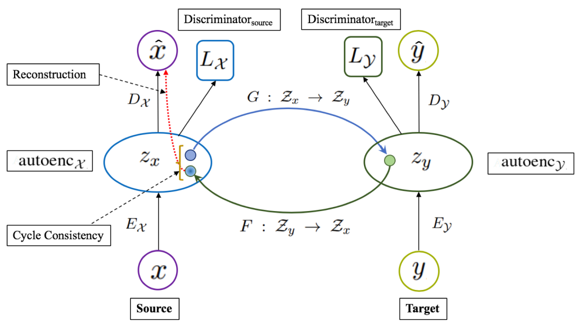

Figure 1: Our proposed adversarial autoencoder framework for unsupervised word translation.

code vector.[2] The encoder, decoder and the reconstruction loss for the other autoencoder (autoenc$_{\mathcal{Y}}$) is similarly defined.

Let $q(z_x|x)$ and $q(z_y|y)$ be the encoding distributions of the two autoencoders. We use *adversarial training* to find a mapping between $q(z_x|x)$ and $q(z_y|y)$. This is in contrast with most existing methods (*e.g.,* Conneau et al. (2018); Artetxe et al. (2017)) that directly map the distribution of the source word embeddings $p(x)$ to the distribution of the target $p(y)$. As Søgaard et al. (2018) pointed out, the *isomorphism* does not hold in general between the word embedding spaces of two languages. Mapping the latent codes gives our model more flexibility to induce the required semantic structures in its code space that could potentially yield more accurate mappings.

As shown in Figure 1, we include two linear **mappings** $G : \mathcal{Z}_x \rightarrow \mathcal{Z}_y$ and $F : \mathcal{Z}_y \rightarrow \mathcal{Z}_x$ to project the code vectors (samples from $q(.|.)$) from one language to the other. In addition, we have two language **discriminators**, $L_{\mathcal{X}}$ and $L_{\mathcal{Y}}$. The discriminators are trained to discriminate between the mapped codes and the encoded codes, while the mappers and encoders are jointly trained to fool their respective discriminator. This results in a **three-player** game, where the discriminator tries to identify the origin of a code, and the mapper and the encoder act together to prevent the discriminator to succeed by making the mapped vector and the encoded vector as similar as possible.

Discriminator Loss Let $\theta_{L_{\mathcal{X}}}$ and $\theta_{L_{\mathcal{Y}}}$ denote the parameters of the two discriminators, and W_G and W_F are the mapping weight matrices. The loss for the source discriminator $L_{\mathcal{X}}$ can be written as

[2]We also experimented with a non-linear encoder, but it did not work well.

$$\mathcal{L}_{L_{\mathcal{X}}}(\theta_{L_{\mathcal{X}}}|W_F, \theta_{E_{\mathcal{X}}}) = -\frac{1}{m}\sum_{j=1}^{m} \log P_{L_{\mathcal{X}}}(\text{src} = 0|F(z_{y_j})) - \frac{1}{n}\sum_{i=1}^{n} \log P_{L_{\mathcal{X}}}(\text{src} = 1|z_{x_i}) \quad (3)$$

where $P_{L_{\mathcal{X}}}(\text{src}|z)$ is the probability according to $L_{\mathcal{X}}$ to distinguish whether z is coming from the source encoder (src $= 1$) or from the target-to-source mapper F (src $= 0$). The discrimination loss $\mathcal{L}_{L_{\mathcal{Y}}}(\theta_{L_{\mathcal{Y}}}|W_G, \theta_{E_{\mathcal{Y}}})$ is similarly defined for the target discriminator $L_{\mathcal{Y}}$ using G and $E_{\mathcal{Y}}$.

Our discriminators have the same architecture as Conneau et al. (2018). It is a feed-forward network with two hidden layers of size 2048 and Leaky-ReLU activations. We apply dropout with a rate of 0.1 on the input to the discriminators. Instead of using 1 and 0, we also apply a smoothing coefficient ($s = 0.2$) in the discriminator loss.

Adversarial Loss The mappers and encoders are trained jointly with the following adversarial loss to fool their respective discriminators.

$$\mathcal{L}_{\text{adv}}(W_F, \theta_{E_{\mathcal{X}}}|\theta_{L_{\mathcal{X}}}) = -\frac{1}{m}\sum_{i=1}^{m} \log P_{L_{\mathcal{X}}}(\text{src} = 1|F(z_{y_j})) - \frac{1}{n}\sum_{i=1}^{n} \log P_{L_{\mathcal{X}}}(\text{src} = 0|z_{x_i}) \quad (4)$$

The adversarial loss for mapper G and encoder $E_{\mathcal{Y}}$ is similarly defined. Note that we consider both the mapper and the target encoder as generators. This is in contrast to existing adversarial methods, which do not use any autoencoder in the target side. The mapper and the target encoder team up to fool the discriminator. This forces the discriminator to improve its skill and vice versa for the generators, forcing them to produce indistinguishable codes through better mapping.

Cycle Consistency and Reconstruction The adversarial method introduced above maps a "bag" of source embeddings to a "bag" of target embeddings, and in theory, the mapper can match the target language distribution. However, mapping at the bag level is often insufficient to learn the individual word level mappings. In fact, there exist infinite number of possible mappings that can match the same target distribution. Thus to learn better mappings, we need to enforce more constraints to our objective.

The first form of constraints we consider is **cycle consistency** to ensure that a source code z_x

translated to the target language code space, and translated back to the original space remains unchanged, *i.e.,* $z_x \rightarrow G(z_x) \rightarrow F(G(z_x)) \approx z_x$. Formally, the cycle consistency loss in one direction:

$$\mathcal{L}_{\text{cyc}}(W_G, W_F) = \frac{1}{n}\sum_{i=1}^{n} \|z_{x_i} - F(G(z_{x_i}))\| \tag{5}$$

The loss in the other direction ($z_y \rightarrow F(z_y) \rightarrow G(F(z_y)) \approx z_y$) is similarly defined. In addition to cycle consistency, we include another constraint to guide the mapping further. In particular, we ask the decoder of the respective autoencoder to reconstruct the original input from the back-translated code. We compute this **post-cycle reconstruction loss** for the source autoencoder as follows:

$$\mathcal{L}_{\text{rec}}(\theta_{E_\mathcal{X}}, \theta_{D_\mathcal{X}}, W_G, W_F) = \frac{1}{n}\sum_{i=1}^{n} \|x_i - D_\mathcal{X}(F(G(z_{x_i})))\|^2 \tag{6}$$

The reconstruction loss at the target autoencoder is defined similarly. Apart from improved mapping, both cycle consistency and reconstruction lead to more stable training in our experiments. Specifically, they help our training to converge and get around the *mode collapse* issue (Goodfellow, 2017). Since the model now has to translate the mapped code back to the source code and reconstruct the original word embedding, the generators cannot get away by mapping all source codes to a single target code.

Total Loss The total loss for mapping a batch from source to target is

$$\mathcal{L}_{\text{src}\rightarrow\text{tar}} = \mathcal{L}_{\text{adv}} + \lambda_1 \mathcal{L}_{\text{cyc}} + \lambda_2 \mathcal{L}_{\text{rec}} \tag{7}$$

where λ_1 and λ_2 control the relative importance of the three loss components. Similarly we define the total loss for mapping in the opposite direction $\mathcal{L}_{\text{tar}\rightarrow\text{src}}$. The complete objective of our model is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{src}\rightarrow\text{tar}} + \mathcal{L}_{\text{tar}\rightarrow\text{src}} \tag{8}$$

3.2 Training and Dictionary Construction

We present the training procedure of our model and the overall word translation process in Algorithm 1. We first pre-train the autoencoders separately on monolingual embeddings (Step 1). This pre-training is required to induce word semantics (and relations) in the latent code space.

We start adversarial training (Step 2) by updating the discriminators for $n\_critics$ (5) times, each time with a random batch. Then we update the generators (the mapper and target encoder) on the adversarial loss. The mappers then go through two more updates, one for cycle consistency and another for post-cycle reconstruction. The autoencoders (encoder-decoder) in this stage get updated only on the post-cycle reconstruction loss. We also apply the orthogonalization update to the mappers following Conneau et al. (2018) with $\beta = 0.01$.

Our training setting is similar to Conneau et al. (2018), and we apply the same pre- and post-processing steps. We use stochastic gradient descent (SGD) with a batch size of 32, a learning rate of 0.1, and a decay of 0.98.

For selecting the best model, we use the *unsupervised validation criterion* proposed by Conneau et al. (2018), which correlates highly with the mapping quality. In this criterion, $10,000$ most

Algorithm 1: Unsupervised word translation with cycle-consistent adversarial autoencoder

```
Input : Two sets of word embeddings: $\mathcal{X}$ and $\mathcal{Y}$
// Initial autoencoder training
1. Train autoenc$_\mathcal{X}$ and autoenc$_\mathcal{Y}$ separately for some
   epochs on monolingual embeddings (Eq. 2);
// Adversarial training
2. for n_epochs do
    for n_iterations do
        // Critic update
        for n_critics do
            (i) Sample a batch from $\mathcal{X}$ and $\mathcal{Y}$
            (ii) Update discriminators ($L_\mathcal{X}$, $L_\mathcal{Y}$) (Eq. 3)
        end
        (a) Sample a batch from $\mathcal{X}$ as source and $\mathcal{Y}$ as
            target
        (b) Update mapper $G$ and encoder $E_\mathcal{Y}$ on
            adversarial loss to fool $L_\mathcal{Y}$ (Eq. 4)
        (c) Update mappers $G$ and $F$ on cycle
            consistency loss (Eq. 5)
        (d) Update mappers ($G$, $F$) and autoenc$_\mathcal{X}$ on
            post-cycle reconstruction loss (Eq. 6)
        // Orthogonalize the mapper
        (e) Update weight matrices of mapper $G$ and $F$
            using:
                $W_G \leftarrow (1+\beta)W_G - \beta(W_G W_G^T)W_G$
                $W_F \leftarrow (1+\beta)W_F - \beta(W_F W_F^T)W_F$
        (f) Sample a batch from $\mathcal{Y}$ as source and $\mathcal{X}$ as
            target and update accordingly (symmetric to
            (b) -(e) steps).
    end
    Use validation criterion to save the best model.
end
// Iterative Procrustes/fine-tuning
3. Load the best model.
for n_iterations do
    (a) Build a synthetic dictionary (using source
        encoder, source-to-target mapper, and CSLS)
    (b) Apply the Procrustes solution on the dictionary.
end
// Test
4. Test the model on gold bilingual dictionary.
```

frequent source words along with their nearest neighbors in the target space are considered. The average cosine similarity between these pseudo translations is considered as the validation metric.

The initial bilingual dictionary induced by adversarial training (or any other unsupervised method) is generally of lower quality than what could be achieved by a supervised method. Conneau et al. (2018) and Artetxe et al. (2018b) propose fine-tuning methods to refine the initial mappings. Similar to Conneau et al. (2018)), we fine-tune our initial mappings (G and F) by iteratively solving the *Procrustes* problem and applying a dictionary induction step. This method uses singular value decomposition or SVD of $\mathcal{Z}_y^T \mathcal{Z}_x$ to find the optimal mappings G (similarly SVD($\mathcal{Z}_x^T \mathcal{Z}_y$) for F) given the approximate alignment of words from the previous step. For generating synthetic dictionary in each iteration, we only consider the translation pairs that are mutual nearest neighbors. In our fine-tuning, we run five iterations of this process. For finding the nearest neighbors, we use the Cross-domain Similarity Local Scaling (CSLS) which works better in mitigating the hubness problem (Conneau et al., 2018).

4 Experimental Settings

Following the tradition, we evaluate our model on **word translation** (*a.k.a.* **bilingual lexicon induction**) task, which measures the accuracy of the predicted dictionary to a gold standard dictionary.

4.1 Datasets

We evaluate our model on two different datasets. The first one is from Conneau et al. (2018), which consists of FastText monolingual embeddings of (d =) 300 dimensions (Bojanowski et al., 2017) trained on Wikipedia monolingual corpus and gold dictionaries for 110 language pairs.[3] To show the generality of different methods, we consider **European**, **non-European** and **low-resource** languages. In particular, we evaluate on English (En) from/to Spanish (Es), German (De), Italian (It), Arabic (Ar), Malay (Ms), and Hebrew (He).

We also evaluate on the more challenging dataset of Dinu et al. (2015) and its subsequent extension by Artetxe et al. (2018a). We will refer to this dataset as **Dinu-Artexe** dataset. From this dataset, we choose to experiment on English from/to Italian and Spanish. English and Italian embeddings were trained on WacKy corpora using CBOW (Mikolov et al., 2013b), while the Spanish embeddings were trained on WMT News Crawl. The CBOW vectors are also of 300 dimensions.

4.2 Baselines

We compare our method with the **unsupervised** models of Conneau et al. (2018), Artetxe et al. (2018b), Alvarez-Melis and Jaakkola (2018), Xu et al. (2018a), and Hoshen and Wolf (2018).

To evaluate how our unsupervised method compares with methods that rely on a bilingual seed dictionary, we follow Conneau et al. (2018), and compute a **supervised** baseline that uses the Procrustes solution directly on the seed dictionary (5000 pairs) to learn the mapping function, and then uses CSLS to do the nearest neighbor search. We also compare with the supervised approaches of Artetxe et al. (2017, 2018a), which to our knowledge are the state-of-the-art supervised systems. For some of the baselines, results are reported from their papers, while for the rest we report results by running the publicly available codes on our machine.

For training our model on European languages, the weight for cycle consistency (λ_1) in Eq. 7 was always set to 5, and the weight for post-cycle reconstruction (λ_2) was set to 1. For non-European languages, we use different values of λ_1 and λ_2 for different language pairs. [4] The dimension of the code vectors in our model was set to 350.

5 Results

We present our results on European languages on the datasets of Conneau et al. (2018) and Dinu et al. (2015) in Tables 1 and 3, while the results on non-European languages are shown in Table 2. Through experiments, our goal is to assess:

(i) Does the unsupervised mapping method based on our proposed adversarial autoencoder model improve over the best existing adversarial method of Conneau et al. (2018) in terms of mapping accuracy and convergence (Section 5.1)?

(ii) How does our unsupervised mapping method compare with other unsupervised and supervised approaches (Section 5.2)?

[3] `https://github.com/facebookresearch/MUSE`

[4] We did not tune the λ values much, rather used our initial observation. Tuning λ values might yield even better results.

	En-Es		En-De		En-It	
	→	←	→	←	→	←
Supervised (Procrustes-CSLS)	82.4	83.9	75.3	72.7	78.1	78.1
Unsupervised Baselines						
Artetxe et al. (2018b)	82.2	84.4	74.9	74.1	78.8	79.5
Alvarez and Jaakkola (2018)	81.7	80.4	71.9	72.8	78.9	75.2
Xu et al. (2018b)	79.5	77.8	69.3	67.0	73.5	72.6
Hoshen and Wolf (2018)	82.1	84.1	74.7	73.0	77.9	77.5
Conneau et al. (2018) (paper)	81.7	83.3	74.0	72.2	-	-
Conneau et al. (2018) (code)	82.3	83.7	74.2	72.6	78.3	78.1
Our Unsupervised Approach						
Adversarial autoencoder +						
Conneau et al. (2018) Refinement	82.6	84.4	**75.5**	73.9	78.8	78.5
Artetxe et al. (2018b) Refinement	**82.7**	**84.7**	75.4	**74.3**	**79.0**	**79.6**

Table 1: Word translation accuracy (P@1) on European languages on the dataset of Conneau et al. (2018) using fastText embeddings (trained on Wikipedia). '-' indicates the authors did not report the number.

	En-Ar		En-Ms		En-He	
	→	←	→	←	→	←
Supervised Baselines						
Artetxe et al. (2017)	24.8	43.3	38.8	41.6	32.7	51.1
Artetxe et al. (2018a)	36.2	52.9	51.2	47.7	43.6	56.8
Supervised (Procrus-CSLS)	34.5	49.7	47.3	46.6	39.2	54.1
Unsupervised Baselines						
Hoshen and Wolf (2018)	34.4	49.3	**	**	36.5	52.3
Artetxe et al. (2018b)	36.1	48.7	54.0	**55.4**	43.8	**57.5**
Conneau et al. (2018) (code)	29.3	47.6	46.2	**	36.8	53.1
Our Unsupervised Approach						
Adversarial autoencoder +						
Conneau et al. (2018) Refinement	33.6	49.7	49.5	44.3	40.0	54.9
Artetxe et al. (2018b) Refinement	**36.3**	**52.6**	**54.1**	51.7	**44.0**	57.1

Table 2: Word translation accuracy (P@1) on non-European and low-resource languages on the dataset of Conneau et al. (2018) using fastText embeddings. ** indicates the model failed to converge.

(iii) Which components of our adversarial autoencoder model attribute to improvements (Section 5.3)?

5.1 Comparison with Conneau et al. (2018)

Since our approach follows the same steps as Conneau et al. (2018), we first compare our proposed model with their model on European (Table 1), non-European and low-resource languages (Table 2) on their dataset. In the tables, we present the numbers that they reported in their paper (Conneau et al. (2018) (paper)) as well as the results that we get by running their code on our machine (Conneau et al. (2018) (code)). For a fair comparison with respect to the quality of the learned mappings (or induced seed dictionary), here we only consider the results of our approach that use the refinement procedure of Conneau et al. (2018).

In Table 1, we see that our **Adversarial autoencoder + Conneau et al. (2018) Refinement** outperforms Conneau et al. (2018) in all the six translation tasks involving European language pairs, yielding gains in the range 0.3 - 1.3%. Our method is also superior to theirs for the non-European and low-resource language pairs in Table 2. Here our method gives more gains ranging from 1.8 to 4.3%. Note specifically that Malay (Ms) is a low-resource language, and the FastText contains word vectors for only 155K Malay words. We found their model to be very fragile for En from/to Ms, and does not converge at all for Ms→En. We ran their code 10 times for Ms→En but failed every time. Compared to that, our method is more robust and converged most of the time we ran.

If we compare our method with Conneau et al. (2018) on the dataset of (Dinu et al., 2015; Artetxe et al., 2017) in Table 3, we see here also our method performs better than their method in all the four translation tasks involving European language pairs. In this dataset, our method shows more robustness compared to their method. For example, their method had difficulties in converging for En from/to Es translations; for En→Es, it converges only 2 times out of 10 attempts, while for Es→En it did not converge a single time in 10 attempts. Compared to that, our method was more robust, converging 4 times out of 10 attempts.

In Section 5.3, we compare our model with Conneau et al. (2018) more rigorously by evaluating them with and without fine-tuning and measuring their performance on P@1, P@5, and P@10.

5.2 Comparison with Other Methods

In this section, we compare our model with other state-of-the-art methods that do not follow the same procedure as us and Conneau et al. (2018). For example, Artetxe et al. (2018b) do the initial mapping in the similarity space, then they apply a different self-learning method to fine-tune the embeddings, and perform a final refinement with symmetric re-weighting. Instead of mapping from source to target, they map both source and target embeddings to a common space.

Let us first consider the results for European language pairs on the dataset of Conneau et al. (2018) in Table 1. Our **Adversarial autoencoder + Conneau et al. (2018) Refinement** performs better than most of the other methods on this dataset, achieving the highest accuracy for 4 out of 6 translation tasks. For De→En, our result is very close to the best system of Artetxe et al. (2018b) with only 0.2% difference.

	En-It		En-Es	
	→	←	→	←
Supervised Baselines				
Artetxe et al. (2017)	39.7	33.8	32.4	27.2
Artetxe et al. (2018a)	45.3	38.5	37.2	29.6
Procrustes-CSLS	44.9	38.5	33.8	29.3
Unsupervised Baselines				
Artetxe et al. (2018b)	**47.9**	42.3	**37.5**	31.2
Conneau et al. (2018) (paper)	45.1	38.3	-	-
Conneau et al. (2018) (code)	44.9	38.7	34.7	**
Our Unsupervised Approach				
Adversarial autoencoder +				
Conneau et al. (2018) Refinement	45.3	39.4	35.2	29.9
Artetxe et al. (2018b) Refinement	47.6	**42.5**	37.4	**31.9**

Table 3: Word translation accuracy (P@1) on the English-Italian and English-Spanish language pairs of Dinu-Artetxe dataset (Dinu et al., 2015; Artetxe et al., 2017). All methods use CBOW embeddings. ** indicates the model failed to converge; '-' indicates the authors did not report the number.

	En-It		En-Es	
	→	←	→	←
Dinu-Artetxe Dataset	**	**	**	**
Conneau Dataset	01.2	01.6	04.7	05.1

Table 4: Conneau et al. (2018) refinement applied to the initial mappings of Artetxe et al. (2018b). ** indicates the model failed to converge.

On the dataset of Dinu et al. (2015); Artetxe et al. (2017) in Table 3, our **Adversarial autoencoder + Conneau et al. (2018) Refinement** performs better than other methods except Artetxe et al. (2018b). On average our method lags behind by about 2%. However, as mentioned, they follow a different refinement and mapping methods. For non-European and low-resource language pairs in Table 2, our **Adversarial autoencoder + Conneau et al. (2018) Refinement** exhibits better performance than others in one translation task, where the model of Artetxe et al. (2018b) performs better in the rest. One important thing to notice here is that other unsupervised models (apart from ours and Artetxe et al. (2018b)) fail to converge in one or more language pairs.

We notice that the method of Artetxe et al. (2018b) gives better results than other baselines, even in some translation tasks they achieve the highest accuracy. To understand whether the improvements of their method are due to a better initial mapping or better post-processing, we conducted two additional experiments. In our first experiment, we use their method to induce the initial seed dictionary and then apply iterative Procrustes solution (same refinement procedure of Conneau et al. (2018)) for refinement. Table 4 shows the results. Surprisingly, on both datasets their initial mappings fail to produce any reasonable results. So we suspect that the main gain in (Artetxe et al., 2018b) comes from their fine-tuning method, which they call *robust self learning*. In our second experiment, we use the initial dictionary induced by our adversarial training and then apply their refinement procedure. Here for most of the translation tasks, we achieve better results; see the model **Adversarial autoencoder + Artetxe et al. (2018b) Refinement** in Tables 1 - 3. These two experiments demonstrate that the quality of the initial dictionary induced by our model is far better than that of Artetxe et al. (2018b).

5.3 Model Dissection

We further analyze our model by dissecting it and measuring the contribution of each novel component that is proposed in this work. We achieve this by *incrementally* removing a new component from the model and evaluating it on different translation tasks. In order to better understand the contribution of each component, we evaluate each model by measuring its **P@1**, **P@5**, and **P@10 with fine-tuning** and **without fine-tuning**. In case of **without fine-tuning**, the models apply the CSLS neighbor search directly on the mappings learned from the adversarial training, *i.e.,* no Procrustes solution based refinement is done after the adversarial training. This setup allows us to compare our model directly with the adversarial model of Conneau et al. (2018), putting the effect of fine-tuning aside.

Table 5 presents the ablation results for En-Es, En-De, and En-It in both directions. The first row (**Conneau-18**) presents the results of Conneau et al. (2018) that uses adversarial training to map the *word embeddings*. The next row shows the results of **our full** model. The subsequent rows incrementally detach one component from our model. For example, **- Enc. adv** denotes the variant of our model where the target encoder is not trained on the adversarial loss ($\theta_{E_{\mathcal{X}}}$ in Eq. 4); **- - Recon** excludes the post-cycle reconstruction loss from **- Enc. adv**, and **- - - Cycle** excludes the cycle consistency from **- - Recon**. Thus, **- - - Cycle** is a variant of our model that uses only adversarial loss to learn the mapping. However, it is important

	En→Es			Es→En			En→De			De→En			En→It			It→En		
	P@1	**P@5**	**P@10**	**P@1**	**P@5**	**P@10**	**P@1**	**P@5**	**P@10**	**P@1**	**P@5**	**P@10**	**P@1**	**P@5**	**P@10**	**P@1**	**P@5**	**P@10**
	Without Fine-Tuning																	
Conneau-18	65.3	73.8	80.6	66.7	78.3	80.8	61.5	70.1	78.2	60.3	70.2	77.0	64.8	75.3	79.4	63.8	77.1	81.8
Our (full)	71.8	81.1	85.7	72.7	81.5	83.8	64.9	74.4	81.8	63.1	71.3	79.8	68.2	78.9	83.7	67.5	77.6	82.1
- Enc. adv	70.5	79.7	83.5	71.3	80.4	83.3	63.7	73.5	79.3	62.6	70.5	79.0	67.6	77.3	82.7	66.2	78.3	82.5
- - Recon	70.1	78.9	83.4	70.8	81.1	83.4	63.1	73.8	80.5	62.2	71.7	78.7	66.9	79.7	82.1	64.8	78.6	82.1
- - - Cycle	66.8	76.5	82.1	67.2	79.9	82.7	61.4	69.7	77.8	60.1	69.8	76.5	65.3	75.1	78.9	64.4	77.6	81.7
	With Fine-Tuning																	
Conneau-18	82.3	90.8	93.2	83.7	91.9	93.5	74.2	89.0	91.5	72.6	85.7	88.8	78.3	88.4	91.1	78.1	88.2	90.6
Our (full)	82.6	91.8	93.5	84.4	92.3	94.3	75.5	90.1	92.9	73.9	86.5	89.3	78.8	89.2	91.9	78.5	88.9	91.1
- Enc. adv	82.5	91.6	93.5	84.3	92.1	94.3	75.4	89.7	92.7	73.5	86.3	89.2	78.4	89.0	91.8	78.1	88.7	91.0
- - Recon	82.5	91.6	93.4	84.1	92.2	94.3	75.3	89.4	92.6	73.2	85.9	89.0	78.2	89.1	91.9	78.2	88.8	91.2
- - - Cycle	82.4	91.0	93.1	83.6	92.2	94.0	74.3	89.7	92.6	72.7	86.1	89.1	77.8	89.2	91.8	77.4	88.3	90.8

Table 5: Ablation study of our adversarial autoencoder model on the dataset of Conneau et al. (2018).

to note that in contrast to Conneau et al. (2018), our mapping is performed at the code space.

As we compare our full model with the model of Conneau et al. (2018) in the *without fine-tuning* setting, we notice large improvements in all measures across all datasets: 5.1 - 7.3% in En→Es, 3 - 6% in Es→En, 3.4 - 4.3% in En→De, 1 - 3% in De→En, 3.4 - 4.3% in En→It, and 0.3 - 3.7% in It→En. These improvements demonstrate that our model finds a better mapping compared to Conneau et al. (2018). Among the three components, the cycle consistency is the most influential one across all languages. Training the target encoder adversarially also gives a significant boost. The reconstruction has less impact. If we compare the results of - - - *Cycle* with *Conneau-18*, we see sizeable gains for En-Es in both directions. This shows the benefits of mapping at the code level.

Now let us turn our attention to the results with fine-tuning. Here also we see gains across all datasets for our model, although the gains are not as verbose as before (about 1% on average). However, this is not surprising as it has been shown that *iterative fine-tuning* with Procrustes solution is a robust method that can recover many errors made in the initial mapping (Conneau et al., 2018). Given a good enough initial mapping, the measures converge nearly to the same point even though the differences were comparatively more substantial initially; for example, notice that the scores are very similar for P@5 and P@10 measures after fine-tuning.

6 Conclusions

We have proposed an adversarial autoencoder framework to learn the cross-lingual mapping of monolingual word embeddings of two languages in a completely unsupervised way. In contrast to the existing methods that directly map word embeddings, our method first learns to transform the embeddings into latent code vectors by pretraining an autoencoder. We apply adversarial training to map the distributions of the source and target code vectors. In our adversarial training, both the mapper and the target encoder are treated as generators that act jointly to fool the discriminator. To guide the mapping further, we include constraints for cycle consistency and post-cycle reconstruction.

Through extensive experimentations on six different language pairs comprising European, non-European and low-resource languages from two different data sources, we demonstrate that our method outperforms the method of Conneau et al. (2018) for all translation tasks in all measures (P@{1,5,10}) across all settings (with and without fine-tuning). Comparison with other existing methods also shows that our method learns better mapping (not considering the fine-tuning). With an ablation study, we further demonstrated that the cycle consistency is the most important component followed by the adversarial training of target encoder and the post-cycle reconstruction. In future work, we plan to incorporate knowledge from the similarity space in our adversarial framework.

Acknowledgments

The authors would like to thank the funding support from MOE Tier-1 (Grant M4011897.020).

References

David Alvarez 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.

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*.

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*.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471. Association for Computational Linguistics.

Ian J. Goodfellow. 2017. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160.

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.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations (ICLR)*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. 2015. Adversarial autoencoders. *CoRR*, abs/1511.05644.

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.

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.

Sebastian Ruder, Ivan Vulic, and Anders Sogaard. 2017. A survey of cross-lingual word embedding models.

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.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *HLT-NAACL*, pages 1006–1011. The Association for Computational Linguistics.

Ruochen Xu, Yiming Yang, Naoki Otani, and Yuexin Wu. 2018a. 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.

Ruochen Xu, Yiming Yang, Naoki Otani, and Yuexin Wu. 2018b. 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. 2017a. 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.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017b. Earth mover's distance minimization for unsupervised bilingual lexicon induction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1934–1945. Association for Computational Linguistics.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*.