Learning Multilingual Word Embeddings without Cross-Lingual Supervision *

Jaskaran Singh
jaskaransing@umass.edu

Soumya Saha
soumyasaha@umass.edu

1 Problem statement

Vector-space representations or word embeddings are induced to capture syntactic and semantic properties of words are widely used in statistical models of natural language. Since raw text can often be difficult to process, word embeddings are used as the input in most Natural Language Processing(NLP) tasks like sentiment analysis, document classification, and machine translation. In addition to improving the performance on standard monolingual NLP tasks, shared representation of words across languages offers intriguing possibilities.

While most early methods of generating embeddings were monolingual, i.e. they were generated for words of one language using text from the same language, the need for cross-lingual embeddings could not be more pronounced today. Consider building a text classifier like a spam filter. We would have to collect annotated training data for all possible target languages and train separate models, or alternatively translate every email to English and then perform classification. Neither of the approaches seem satisfactory. Approach 1 seems time-consuming and approach 2 may lead to inaccurate results due to bad translation.

Thus, we see the need for the use of multilingual embeddings. Multilingual word embeddings (MWEs) are obtained by projecting different monolingual word embeddings into a shared embedding space such that they retain the local relationships while also forming new cross-lingual ones. That is to say, if the words 'cat' and 'dog' are close to each other in the English embedding space, they should be close to each other and say, the French words for 'cat' and 'dog' in the shared embedding space.

Using MWEs, we can project words from a new language into the shared space and achieve better results for spam classification because spam indicators in different languages would be in the same vicinity.

2 Proposals and Accomplishments

We proposed the following:

- To study and implement the model published in (Chen and Cardie, 2018)
- To replicate the results of their experiments in two evaluation tasks (CLWS, MWT which are described later)
- To produce visualizations to demonstrate the results of projecting monolingual embeddings to a shared space.
- To demonstrate through visualizations or otherwise where the model does and does not do well.

Our results were:

- We were able to implement a version of the model by Chen and Cardie and produced commendable results compared to their scores and other baselines.
- However, we could not replicate their exact results for the subset of languages we ran our experiments on. Our initial understanding was that in training over a smaller set of languages, the model might not have been exposed to all the interdependencies between languages and thus failed to capture certain relationships.

* code: https://github.com/soumyaumass/umwe

3 Related work

Cross-lingual embedding mappings have attracted a lot of attention recent times. It proves to be an effective way to learn bilingual word embeddings(BWE) (Lazaridou et al., 2015). The underlying idea is to independently train the embeddings in different languages using monolingual corpora, and then map them to a shared space through a linear transformation (Mikolov et al., 2013). This allows to learn high-quality cross-lingual representations without expensive supervision, opening new research avenues like unsupervised neural machine translation (Wu et al., 2016; Artetxe et al., 2017b)

There is an abundance of literature on learning crosslingual word representations, focusing either on a pair of languages, or multiple languages at the same time. Most of these methods are supervised, and use a bilingual dictionary of a few thousand entries to learn the mapping. A couple of notable methods with supervision are orthogonal methods, which map the embeddings in one or both languages under the constraint of the transformation being orthogonal (Smith et al., 2017), and margin methods, which map the embeddings in one language to maximize the margin between the correct translations and the rest of the candidates (Lazaridou et al., 2015).

A related research line is to adapt these methods to the semi-supervised scenario, where the training dictionary is much smaller and used as part of a bootstrapping process. A practical approach for reducing the need of bilingual supervision is to design heuristics to build the seed dictionary. The role of the seed lexicon in learning cross-lingual embedding mappings is analyzed in depth by (Vulic and Korhonen, 2016), who propose using document-aligned corpora to extract the training dictionary.

However, while these approaches are meant to eliminate the need of bilingual data in practice, they also make strong assumptions on the writing systems of languages (e.g. that they all use a common alphabet or Arabic numerals). Closer to our work, a recent line of fully unsupervised approaches drops these assumptions completely, and attempts to learn cross-lingual embedding mappings based on distributional information alone. For that purpose, existing methods rely on adversarial training.

Adversarial Neural Networks have been successfully applied to various cross-lingual NLP tasks where annotated data is not available, such as cross-lingual text classification (Chen et al., 2016), unsupervised machine translation (Lample et al., 2018a) etc. Use of Adversarial training for these purposes were first proposed in (Barone, 2016), who combined an encoder that maps source language embeddings into the target language, a decoder that reconstructs the source language embeddings from the mapped embeddings, and a discriminator that discriminates between the mapped embeddings and the true target language embeddings.

The Multilingual Pseudo-Supervised Refinement (MPSR) method proposed by (Chen and Cardie, 2018) uses best embeddings from adversarial training to refine the remaining embeddings in a pseudo-supervised model which optimizes the embeddings using gradient descent. There has been previous work on using weak supervision (Artetxe et al., 2017a) and pseudo supervision by iterative orthogonal Procrustes method (Lample et al., 2018b) for unsupervised BWEs. In MWEs, multilingual embeddings don't have a closed-form solution for linear mapping between languages, hence MPSR is used.

4 Methodology

We assume that we have access to monolingual embeddings for each of the N languages, which can be obtained using unlabeled monolingual corpora. Our model will jointly model the monolingual embeddings of all N languages into a single vector space.

To clarify the notation used in (Chen and Cardie, 2018), we have a set of languages $\mathcal{L}$ with total number of languages equal to N. For each language $l \in \mathcal{L}$ with vocabulary $\mathcal{V}_l$, we have monolingual word embedding space $\mathcal{S}_l$ of size $|\mathcal{V}_l| \times d$ where d is vector dimension of each word.

We didn't use any complex non-linear neural mapping to map the multilingual vector spaces together as previous research (Mikolov et al., 2013) has proven linear mapping works better

as there is a strong correlation between any pair of languages. (Xing et al., 2015) have also shown that using orthogonal matrices as the linear mappings make the outcome better. Thus, we make our encoders orthogonal, $\mathcal{M}_l^{-1} = \mathcal{M}_l^T$. This also helps in avoiding complex matrix inverse calculation.

Another benefit of using linear encoders and decoders is that we can learn $N-1$ mappings instead of N by choosing the target space $\mathcal{T}$ to be the embedding space of a specific language called the target language. Given a MWE with an arbitrary $\mathcal{T}$, we can construct an equivalent one with only $N-1$ mappings by multiplying the encoders of each language $\mathcal{M}_l$ to the decoder of the chosen target language $\mathcal{M}_t^T$:

$$\mathcal{M}'_t = \mathcal{M}_t^T \mathcal{M}_t = \mathcal{I}$$
$$\mathcal{M}'_l \mathcal{E}_l = (\mathcal{M}_t^T \mathcal{M}_l)\mathcal{E}_l \vdash \mathcal{S}_t$$

where $\mathcal{I}$ is the identity matrix.

We will now present our two-step approach, whose details are as follows:

4.1 Multilingual Adversarial Training(MAT)

We set up a conventional adversarial training system (Goodfellow et al., 2014) wherein our generator has 2 parts: an encoder $\mathcal{M}_i$ and a decoder for each language. Each language also has a dedicated discriminator $\mathcal{D}_i$ which is a fully-connected (FC) net that uses dropout and has a sigmoid layer on top.

The purpose of the encoder for a language i is to project monolingual embeddings for the language into the shared target embedding space (which can be the embedding space of a language $\mathcal{T}$). Following this, the decoder for another language, say j, takes these embeddings and projects them into the monolingual embedding space of language j. These projected monolingual embeddings along with actual embeddings of language j are passed to the discriminator for j. The discriminator tries to classify which embeddings are real (actual monolingual embeddings) and which ones are fake (projected from shared embedding space), till the two become indistinguishable.

Thus, the goal for the encoder-decoder system is to try and generate embeddings which are as close to monolingual embeddings as possible so as to fool the discriminator. At the same time, the discriminator tries to classify correctly so as to push the encoder-decoder system to produce embeddings very similar to monolingual embeddings.

The loss function (L_d) used in practice was cross-entropy loss. Discriminator output was 1 for real embeddings, 0 for fake (projected). The mathematical objective function for the discriminator of a language i is to minimize:

$$J_{D_i} = L_d(1, \mathcal{D}_i(x_i)) + L_d(0, \mathcal{D}_i(\mathcal{M}_i^T \mathcal{M}_j x_i))$$

For the encoder $\mathcal{M}_i$, the objective is to minimize:

$$J_{M_i} = L_d(1, \mathcal{D}_j(\mathcal{M}_j^T \mathcal{M}_i x_i))$$

Algorithm 1: Training the discriminator

Input: Vocabulary for all languages $\mathscr{L}$

repeat
  loss $= 0$
  for $lang_i \in \mathscr{L}$ **do**
    Select a random $lang_j \in \mathscr{L}$
    Sample `batchsize` words $x_i \sim$ `Vocab(lang`$_i$`)`
    Sample `batchsize` words $x_j \sim$ `Vocab(lang`$_j$`)`
    $proj_i = \mathcal{M}_i(x_i)$
    $tgt_j = \mathcal{M}_j^T(proj_i)$
    $y_{real} = \mathcal{D}_j(x_j)$
    $y_{proj} = \mathcal{D}_j(tgt_j)$
    loss $+= L_d(y_{real}, 1) + L_d(y_{proj}, 0)$
  Update parameters to minimize loss
until *n iterations*

Both of the objectives are optimized over $\forall(Lang_i, Lang_j)$ pairs $\in \mathscr{L}$, including the cases where $i = j$.

4.2 Multilingual Pseudo-Supervised Refinement

The UMWEs trained using MAT are good at aligning the most frequently used words among a given pair of languages but the quality of alignment starts going below par when the less frequently used words are considered from the complete vocabulary space of the language pair.

Algorithm 2: Training the encoder-decoder system

Input: Vocabulary for all languages $\mathscr{L}$

repeat
  loss $= 0$
  for $lang_i \in \mathscr{L}$ **do**
    Select a random $lang_j \in \mathscr{L}$
    Sample `batchsize` words
      $x_i \sim \texttt{Vocab}(\texttt{lang}_i)$
    $proj_i = \mathcal{M}_i(x_i)$
    $tgt_j = \mathcal{M}_j^T(proj_i)$
    $y_{proj} = \mathcal{D}_j(tgt_j)$
    loss $+= L_d(y_{proj}, 1)$
  Update parameters to minimize loss
until *n iterations*

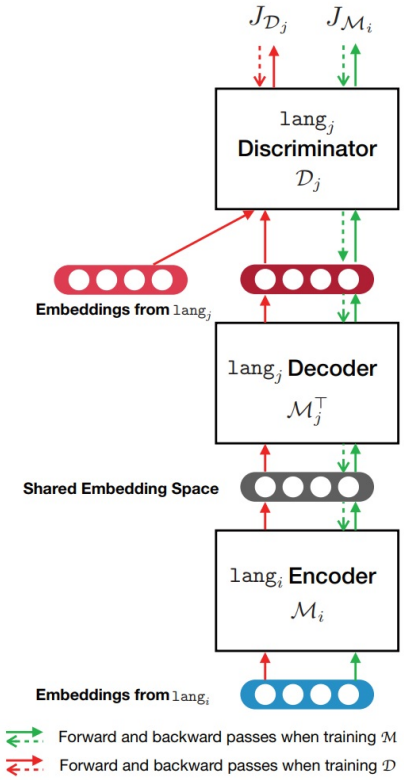

Figure 1: Multilingual Adversarial Training. $lang_i$ and $lang_j$ are two randomly selected languages. (Chen and Cardie, 2018)

Compared to unsupervised BWE methods without cross-lingual supervision, the performance is not state-of-the-art. An iterative refinement method (Artetxe et al., 2017a) can be used to refine the embeddings obtained from adversarial training. This method is used (Chen and Cardie, 2018) to refine the embeddings obtained from MAT by generalizing it to develop a multilingual objective function and minimize it.

This multilingual refinement works on the premise that once we obtain partially aligned embeddings from MAT, we can use the aligned embeddings of the most frequently used words as a form of supervision to improve the mappings from MAT, hence pseudo-supervision. Because these words are frequently used, the embeddings represent the most accurately predicted relations in the vocabulary space of the language pair. Doing this for multiple languages simultaneously essentially minimizes multiple mapping objectives simultaneously. This helps in avoiding the alignment of each language with the target space individually because that is computationally more expensive and using relations between all languages as is also done in MAT.

Algorithm 3: Multilingual Pseudo-Supervised Refinement

Input: A set of pseudo-supervised lexica of word pairs between each pair of languages $\texttt{Lex}(\texttt{lang}_i, \texttt{lang}_j)$

repeat
  loss $= 0$
  for $lang_i \in \mathscr{L}$ **do**
    Select a random $lang_j \in \mathscr{L}$
    Sample $(x_i, x_j) \sim \texttt{Lex}(\texttt{lang}_i, \texttt{lang}_j)$
    $t_i = \mathcal{M}_i(x_i)$ `// encode` x_i
    $t_j = \mathcal{M}_j(x_j)$ `// encode` x_j
    loss $+= L_r(t_i, t_j)$
      `// refinement loss`
  Update all $\mathcal{M}$ parameters to minimize loss
  $\texttt{orthogonalize}(\mathcal{M})$
until *convergence*

MAT is able to train an embedding space which is approximately aligned because of the frequent words but since rarer words generate larger differences in the discriminator, the performance for these rare words is worse. We take all language pairs $(\texttt{lang}_i, \texttt{lang}_j)$ from a set of $\mathscr{L}$ languages, find the most frequently used $15k$ words in each language pair, transform their embeddings $\mathcal{E}_i$, $\mathcal{E}_j$ into the shared embedding space $\mathcal{M}_i\mathcal{E}_i$, $\mathcal{M}_j\mathcal{E}_j$ using their respective mappings $\mathcal{M}_i$ and $\mathcal{M}_j$, find the mutual nearest neighbors between them

and use them to create the pseudo-supervised lexicon $\texttt{Lex}(\texttt{lang}_i, \texttt{lang}_j)$ for each language pair.

Then, the MPSR objective is to minimize:

$$J_r = L_r(\mathcal{M}_i x_i, \mathcal{M}_j x_j)$$

where L_r is the mean square loss function. The MPSR training is detailed in Algorithm 3.

Cross-Lingual Similarity Scaling (CSLS)
To compute nearest neighbors while constructing the pseudo-supervised lexica, a distance metric between embeddings is needed. Standard distance metrics such as the Cosine similarity or Euclidean distance can lead to the hubness problem (Radovanović et al., 2010) in high-dimensional spaces when used to calculate nearest neighbors because of some words are very likely to be the nearest neighbors of many others (hubs), while others are not the nearest neighbor of any word.
This problem can be averted successfully by using Cross-Lingual Similarity Scaling (CSLS) as a drop-in replacement for cosine similarity whenever a distance metric is needed. The CSLS similarity (whose negation is a distance metric) is calculated as follows:

$$\begin{aligned} CSLS(x, y) = 2cos(x, y) \\ - \frac{1}{n} \sum_{y' \in N_Y(x)} cos(x, y') \\ - \frac{1}{n} \sum_{x' \in N_X(y)} cos(x', y) \end{aligned}$$

where $N_Y(x)$ is the set of n nearest neighbors of x in the vector space that y comes from: $Y = \{y_1, ..., y_{|Y|}\}$, and vice versa for $N_X(y)$. Generally, $n = 10$ is used.

Orthogonalization
When learning transformations between the embedding spaces of different languages, orthogonal linear mappings are the most preferred choice. Therefore, we perform an orthogonalization update (Cisse et al., 2017) after each step to make sure the mappings are almost orthogonal to each other by using the following formula with $\beta = 0.001$:

$$\mathcal{M}_l = (1 + \beta)\mathcal{M}_l - \beta \mathcal{M}_l \mathcal{M}_l^T \mathcal{M}_l \quad : \forall l$$

5 Dataset

For each of the languages the model is supposed to work on, it needs pre-trained monolingual word embeddings. In our experiments, we used pre-trained 300 dimensional FastText embeddings released by Facebook Research trained on Wikipedia corpora using the Skipgram with Negative Sampling (SGNS) model. The pre-trained FastText embeddings data contains each token and corresponding vectors. The size of total number of tokens for different languages differ significantly. Lower resource languages like Persian contain fewer tokens than the ubiquitous ones like English.

For each language, we limited ourselves to use the embeddings for the top 200,000 most frequently used words (or more appropriately 200,000 token types since we also considered punctuation). Given the nature of the task, we believe using 200,000 tokens was necessary but the sheer volume did make training compute intensive, especially while running for more than 2-3 languages. Instead of loading data directly from the downloaded text files we saved the vectors along with their token ids in PyTorch files and loaded the data from that, which was significantly faster.

During evaluation, we used two other data sources. To calculate cross-lingual word similarity scores, we used the SEMEVAL17 task 2 dataset. This dataset contains pairs from English, German, Spanish, Italian, and Persian, along with a human-annotated similarity scores. What made the CLWS task challenging for the model was presence of peculiar words/phrases like "Clinton Obama", "GDP", "International Mathematics Olympiad", "Ban ki-Moon" and "HSBC" in the en-es list, "Woody Allen", "GTA" in the en-fa (Persian) list. Each list had a different number of pairs($\sim$ 600-800) in it.

In the Multilingual Word Translation (MWT) task, we used a dataset of 6500 bilingual translated words in multiple languages. This dataset was published by MUSE. One interesting feature that we noticed was that it had multiple possible translations for some words. For example "burned" has 3 translations in Spanish, "sandy" has 4 German translations.

6 Experiments

6.1 Tools

We used the PyTorch 0.4.1 Deep Learning library in Python 3.6 for most of the implementation and Faiss for nearest neighbour search. For creating visualizations of our results, we used the Matplotlib and Seaborn libraries in Python.

6.2 Details about training

While training, we tried a few different architectures for the discriminator but found a fully connected net to be most suited to the task because languages do not show much non-linearity for similar words in their embeddings. We tried hidden dimensions in the range of $512 - 2048$ but found that lower values give very poor performance. We also tested various values and positioning of dropout and finally settled on 0.1 for the input layer. We tried different activation functions like $ReLU$, $tanh$ and $LeakyReLU$. We found that $LeakyReLU$ with parameter 0.1 works out to be slightly better than $ReLU$.

For the final model, we trained it over 5 epochs, each of 1 million iterations for every source language. We also tested different optimizers for MAT and MPSR. We found good results with SGD with initial learning rate 0.1 for MAT and Adam with initial learning rate 0.001. After initial tests with a constant learning rate, we found that our results were not comparable with (Chen and Cardie, 2018). While sanity checking with their model, we discovered that they make use of a decaying learning rate. So we incorporated a decay of 0.95 after each epoch of (MAT+MPSR).

The discriminator was trained for 5 iterations, and then the encoder-decoder system was trained for 1 iteration. This type of unequal training was suggested in (Goodfellow et al., 2014). We also tried to train them equally but that gave inferior results. After an epoch of MAT, we ran the refinement procedure MPSR for 5 epochs of 30k iterations each. The results improved significantly after refinement, as seen in the evaluation tasks.

We have used a system with 8 core CPU and NVIDIA P100 GPU to run our model. The above discussed settings for the training along with the evaluation calculation takes around 2 hours in the system.

7 Baselines

We evaluated our model against two supervised and two unsupervised baseline models. We used both types of baselines to get an accurate representation of where our model lies on the spectrum of performance. The unsupervised methods we used were BWE-Direct and BWE-Pivot.

- BWE-Direct is limited to mapping two languages to a shared space. So we need to train it separately for each pair.
- BWE-Pivot: This method is an extension of BWE-Direct. It uses an intermediate pivot language, usually English. So a model for French-Spanish might proceed from French to English space and then from English to Spanish space.

The supervised methods were:

- NASARI (Novel Approach to a Semantically-Aware Representation of Items) (Camacho-Collados et al., 2015) : This model derives an embedding for a word through two sources, a statistical approach on the Wikipedia corpus and a semantic distribution method through WordNet(Miller, 1995). WordNet is a graph based model that represents a word, phrase or concept through a synset, which is a set of synonymous representations of the target word/phrase. It captures some senses in which a word is used.
- Luminoso (Speer and Lowry-Duda, 2017): This model uses ConceptNet 5(by Speer) which is a a multilingual, domain-general knowledge graph that connects words and phrases with labeled, weighted edges. It then uses retrofitting to adjust the values of existing word embeddings based on a new objective function that also takes a knowledge graph into account.

For training the supervised baselines for calculating MWT scores, we use 5000 pairs out of the 6500 for training, and the remaining 1500 for testing. For our model and the unsupervised baselines, only the 1500 test pairs were used.

8 Evaluation

We have evaluated the generated MWE from our models with two standard evaluation tasks used

widely, the Multilingual Word Translation task, and the SemEval2017 Cross-lingual Word Similarity task.

8.1 Cross-Lingual Word Similarity

In this task, we check how similar the word embeddings of the two words are in the shared embedding space and check how correlated the similarity score is with human scores. We use cosine similarity as the metric to measure similarity between embeddings and finally report the computed Spearman's correlation co-efficient (ρ). Spearman's ρ ranges from +1 to -1, where +1 means strong positive correlation and -1 means strong negative correlation between the embeddings of the two languages. Our CLWS scores for English (en), Spanish (es), German (de) and Perisan (fa) are compared with other baseline supervised methods discussed earlier in table 1.

Table 1: CLWS Scores

	es-en	fa-en	de-en
NASARI	0.630	0.495	0.594
Luminoso	0.774	0.559	0.76
BWE-Direct	0.65	0.6	0.7
BWE-Pivot	0.65	0.6	0.7
Chen and Cardie	0.711	0.68	0.712
This Paper	0.63	0.57	0.58

Our results indicate a commendable performance by equaling NASARI for Spanish-English CLWS task and beating NASARI and Luminoso for Persian-English CLWS task. However it does lag behind BWE-Direct and BWE-Pivot by a small margin.

8.2 Multilingual Word Translation

In this task, we are evaluating our model on the task of word translation between arbitrary pairs of a set of N languages. Results are a percentage of times the model got the translation right out of a test set of size 1500.

Table 2: MWT Scores

	es-en	de-en
BWE-Direct	81.7	74
BWE-Pivot	81.7	74
Chen and Cardie	82	74.8
This Paper	68.4	63.1

From the MWT scores it can be seen that our model performs reasonably well. Though the scores for other state-of-the-art reported by the other unsupervised methods are a little higher but it doesn't fall behind too much.

9 Visualizations

In this section we are adding some visualizations of the learned MWEs by mapping them in the same space. In Figure 2 we have taken the English word "brother" and then found out its nearest neighbors among the learned Spanish embeddings. In Figure 3 we have shown the same thing using the German word "bruder" which is translated to "brother" in English.

Another experiment we tried was to visualize the embeddings for Spanish and English before and after running our model to see the effect of projecting them. We used Principal Component Analysis (PCA) algorithm for dimensionality reduction for converting 300d vectors to 2d. The results are in figure 4.

```
# printing nearest neighbors
# English to Spanish
src_word = 'brother'
get_nn(src_word, src_embeddings,
```

```
Nearest neighbors of "brother":
0.8308 - hermano
0.7738 - hermanastro
0.7699 - sobrino
0.7683 - tío
0.7571 - hijo
```

Figure 2: English to Spanish nearest neighbor search example

```
# printing nearest neighbors
# German to English
src_word = 'bruder'
get_nn(src_word, src_embeddings,
```

```
Nearest neighbors of "bruder":
0.8414 - brother
0.8304 - nephew
0.8213 - father
0.8152 - son
0.7944 - cousin
```

Figure 3: German to English nearest neighbor search example

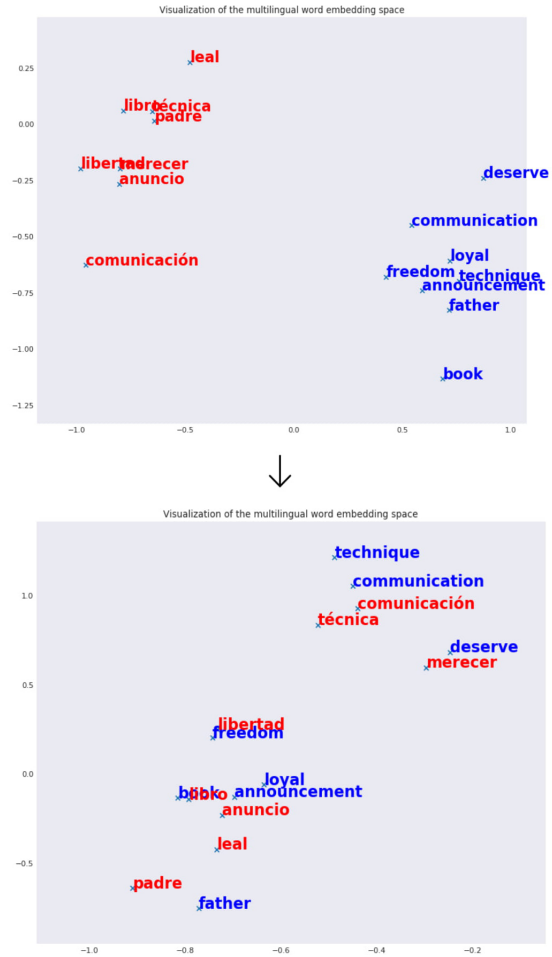

Figure 4: Spanish and English word vectors in the same space after mapping

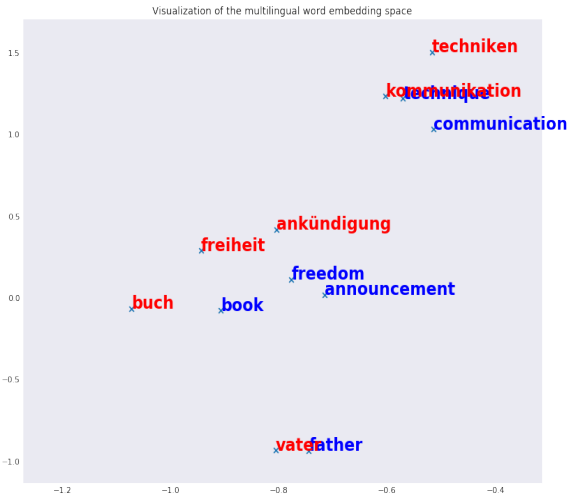

Figure 5: German and English word vectors in the same space after mapping

10 Error analysis

Our model gave decent results often but in some cases, its outputs were somewhat peculiar. We got acceptable results for frequent words but not exactly correct results for uncommon words. This was a very difficult error to debug and took us till the final week to realize that it was an indexing error in one of the MPSR loops. Other than that, we feel we could have achieved better results through extended hyperparamter training and trying out the code for more languages to better incorporate the cross-lingual dependencies. For example, incorporating Romance languages like French, Spanish together might give a better distribution in the shared space.

11 Contributions of group members

- Jaskaran: Coding MAT and CLWS, Writing report, Running Experiments on Architecture and Optimizers, Results Analysis
- Soumya: Coding MPSR and MWT, Writing report, Running experiments on Hyperparameters, Visualization, Debugging

12 Conclusion

In this project, we freshly implemented the MAT + MPSR method to generate high-quality word embeddings for different languages in an unsupervised manner. From the results it can be seen our results outperform supervised algorithms like NASARI in CLWS task for most of the language pairs, but it struggles to achieve quality results compared to another supervised top performing system Luminoso. Among the languages on which we ran our model, we can see for low-resource language pair such as Persian, our model performed really well due to the lack of available supervision in those low-resource languages which could have been employed in those supervised systems mentioned above.

For future work, we can extend our model to incorporate different refinement methods and test their accuracy over our current model. We are also yet to run the model on more low-resource languages to corroborate our performance on those languages. At the same time, we will also try to investigate how we can use our method to work with Bilingual frameworks already implemented, so as to maximize the CLWS and MWT scores for

the improvement in other natural language tasks such as machine translation, language detection.

References

Artetxe, M., Labaka, G., and Agirre, E. (2017a). 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)*, volume 1, pages 451–462.

Artetxe, M., Labaka, G., Agirre, E., and Cho, K. (2017b). Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Barone, A. V. M. (2016). Towards cross-lingual distributed representations without parallel text trained with adversarial autoencoders. *arXiv preprint arXiv:1608.02996*.

Camacho-Collados, J., Pilehvar, M. T., and Navigli, R. (2015). Nasari: a novel approach to a semantically-aware representation of items. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 567–577. Association for Computational Linguistics.

Chen, X. and Cardie, C. (2018). Unsupervised multilingual word embeddings. *arXiv preprint arXiv:1808.08933*.

Chen, X., Sun, Y., Athiwaratkun, B., Cardie, C., and Weinberger, K. (2016). Adversarial deep averaging networks for cross-lingual sentiment classification. *arXiv preprint arXiv:1606.01614*.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., and Usunier, N. (2017). Parseval networks: Improving robustness to adversarial examples. *arXiv preprint arXiv:1704.08847*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Lample, G., Conneau, A., Denoyer, L., and Ranzato, M. (2018a). Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*.

Lample, G., Conneau, A., Ranzato, M., Denoyer, L., and Jgou, H. (2018b). Word translation without parallel data. In *International Conference on Learning Representations*.

Lazaridou, A., Dinu, G., and Baroni, M. (2015). Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 270–280.

Mikolov, T., Le, Q. V., and Sutskever, I. (2013). Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

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

Smith, S. L., Turban, D. H. P., Hamblin, S., and Hammerla, N. Y. (2017). Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *CoRR*, abs/1702.03859.

Speer, R. and Lowry-Duda, J. (2017). ConceptNet at SemEval-2017 Task 2: Extending Word Embeddings with Multilingual Relational Knowledge. *arXiv e-prints*, page arXiv:1704.03560.

Vulic, I. and Korhonen, A. (2016). On the role of seed lexicons in learning bilingual word embeddings. In *ACL*.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xing, C., Wang, D., Liu, C., and Lin, Y. (2015). Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011.