Computational Linguistics Research Paper

Carter Langen
Haverford College
clangen@haverford.edu

Dylan Soemitro
Haverford College
dsoemitro@haverford.edu

Abstract

We train multilingual word embeddings on a small corpus: the Aeneid in Latin, and three English translations of it. We treat each translation as its own language. We present this as progress towards new way of analysing text. However, there are still some problems to work out before this kind of model will be suited to this use.

1 Introduction

Word embeddings are dense vector representations of words. We have made high quality embeddings for high-resource languages using large data and the Skip-gram (SGNS) algorithm (Mikolov et al., 2013a). Here, we make a set of bilingual embeddings based on the Latin Aeneid, and three English translations that are available through Wikisource. To train these embeddings, we fuse three existing extensions of Skip-gram: training them from small data, training on morphologically complex data, and extending them to cross lingual semantic spaces.

2 Related Work

There has been a little work on doing Latin embeddings at all and on doing embeddings on really small Latin data (Bloem et al., 2020). using Nonce2Vec (Herbelot and Baroni, 2019) essentially an optimization of Word2Vec (Mikolov et al., 2013a), for tiny data. Using these as monolingual bases, we can try various methods of mapping them to a bilingual space. There are multiple methods of doing this, requiring different levels of aligned data. Since this is poetry, and translators of Latin into English have been complaining for 500 years about how much more meaning is packed into each Latin word, we will mainly focus on techniques dealing with sentence aligned data, such as in (Coulmance et al., 2015; Hermann and Blunsom, 2014; Upadhyay et al., 2016), rather than word aligned ones.

2.1 Embeddings from Small Data

Herbelot and Baroni (2019) modify Skip-gram to try to make it work on less data, as little as 2 to 6 sentences. To test this, they use a modified Wikipedia dataset and a new chimera dataset (Herbelot and Baroni, 2019). They use the Wikipedia dataset to get a gold vector (gold as in the gold standard) for a word, which they saved, and then randomize the vector so they could relearn it. The vector obtained from a definition sentence in Wikipedia is a high information sentence. The other data set is more interesting: it consists of made up words that are used in regular sentences. The model then tries to guess what the word means, and humans rate the model's choices based on how similar they are to their own. Here is a summary of the model that makes these guesses.

1. **Initialization**: They initialize the Chimera's vector to the sum of the vectors of the rest of the words in the sentences. They know there are better ways to do this, but this is fast and easy.

2. **Parameters**: They make the learning rate start higher and decay faster, and tested making the context window size larger. Recall from the initial discussion of Word2Vec that they found a larger context window tends to improve accuracy, at the cost of training time. (Mikolov et al., 2013a).

The higher learning rate is risky: the first time a word is seen will have a huge impact on its vector value; however, they introduce an exponential decay function for the learning rate to prevent wild swings. They concluded that having a better way of telling the model to take risks at the right time will improve the quality of their vectors. (Herbelot and Baroni, 2019).

2.2 Fast Text

FastText (Bojanowski et al., 2017), in addition to creating word embeddings whole words, creates embeddings of character n-grams. Special word boundary characters ("<" and ">") are introduced to prevent confusion: for example, the n-gram "her" in "where" is different from the n-gram "<her>", the full string representation of the word *her*. Each set of **character n-grams** then gets embeddings trained with SGNS, and the new word embeddings are the sum of their component subwords' embeddings.

Bojanowski et al. (2017) show that this works much better than basic SGNS on multiple morphologically rich Indo-European languages, including but not limited to German, Czech, Russian, and Romanian, and it even made better embeddings for rare words in morphologically deprived languages like English, and it could achieve a similar quality from less (though still large) data. They were worried about running out of memory for the n-grams, and so limited themselves to capturing 2.10^6 of them in a hash-table (Bojanowski et al., 2017). Since Latin is also a morphologically rich language, we expect FastText to produce better Latin embeddings.

2.3 Latin Word Embeddings

To our knowledge, there have only been two previous attempts at making Latin word embeddings: Sprugnoli et al. (2019) and Bloem et al. (2020). Sprugnoli et al. (2019) used a reasonably sized corpus, around 5 million words, and used FastText. It should also be noted that the corpus had been human lemmatized. The idea is that this would effectively lower the morphological complexity of the language. Bloem et al. (2020) used an even smaller corpus than we do to try to analyze Neo-Latin philosophy. They managed to generate consistent embeddings from tiny data, but also had evaluation problems. They considered using synonym selection, but were, like us, worried that domain specific uses of words that the model captures would not be measured. In an earlier paper, they define **consistency** as "if its output does not vary when its input should not trigger variation (e.g. because it is sampled from the same text)" (Bloem et al., 2019).

2.4 Multilingual Embeddings

To make embeddings multilingual means that we represent words from multiple languages in the same semantic space, or have a mapping between them. Coulmance et al. (2015) show that a pivot language can be used: this means that if we want a space that represents English, French, and Spanish, we only need mappings from French to English and Spanish to English (and from English to the other two). We do not need a direct mapping from French to Spanish, because we can get there through English. Alignment is learned as part of running Trans-gram. Unfortunately, their implementation is not public. Other older methods rely on an explicit bilingual dictionary (Mikolov et al., 2013b). Chen and Cardie (2018) publish a tool for training fully unsupervised multilingual embeddings. They build on Mikolov et al. (2013b)'s observation that embedding spaces across languages tend to have a strong linear correlation. They build an encoder-decoder model, where they make orthogonal linear mappings to a shared space, between an arbitrary number of languages. For optimal evaluation, they still require a bilingual lexicon. Neither of these have been tested on small data. Due to the lack of a public implementation from Coulmance et al. (2015), we ended up using Chen and Cardie (2018) instead.

3 Our Work

We want to fuse these ideas to create word embeddings from a single text, and from multiple translations of it. We treat each translation as its own language. Thus, we have four small languages: Vergil which is a subset of Latin, Dryden which is a subset of English from 1697, Connington which is a subset of English from 1866, and Williams which is a subset of English from 1911. We use Vergil as our pivot language, draw new comparisons into Dryden, Williams, and Connington. We further subdivide Latin into a lemmatized and unlemmatized version. We opt to do this for several reasons:

1. To reduce word rarity. A particular word form may only appear in the poem a handful of times, even if it is an otherwise frequent word.

The embeddings for these word forms should be nearly identical, but occur so rarely that they end up far apart. Bojanowski et al. (2017) targets this

2. To make quantitative evaluation easier. We need a bilingual lexicon to evaluate, and it is much simpler if we use lemmatized Latin instead.

3. To make qualitative evaluation easier. Frequent word's nearest neighbors are flooded with other forms of the same word, which obscures more interesting analysis.

4. Finally, Latin morphology largely encodes syntactic relationships, and we are more interested in using these to investigate semantics.

4 Process

First, we make the monolingual embeddings for each translation and the original, using FastText (Bojanowski et al., 2017). The FastText package allows us to change the learning rate, and set it to decay by a percent amount each time. While not exactly the exponential decay function shown in Herbelot and Baroni (2019), it is close enough. We then take the monolingual embeddings map them into a multilingual space using Chen and Cardie (2018). We use Chen and Cardie (2018) for several practical reasons: first, the code for Trans-gram is not public, and re-implementing it ourselves nearly from scratch takes too much time. Second, both Chen and Cardie (2018); Bojanowski et al. (2017) were developed at Facebook, and Chen and Cardie (2018)'s models were designed assuming Bojanowski et al. (2017)'s monolingual vectors as the input.

5 Model Architecture

Each language model used the default FastText settings, except:

1. Dimensions: the default is 100, but we wanted 300

2. Learning Rate: the default is 0.05. We knew we needed to start higher, and experimented with this. Eventually, we settled on 0.25.

3. Learning Rate Update: the default is 100. This is a percentage value for how much to change the learning rate by each epoch, so we experiment with various values less than 100. Eventually, we settled on 20 for Latin, so the learning rate would fall by 80 percent each time (documentation unclear, this is our interpretation).

4. For the Latin model, we change the minimum number of word appearances to 1, so that we model all word forms. This is especially relevant to the non-lemmatized model.

6 Results

We evaluate our bilingual embeddings using the built in tools provided in Chen and Cardie (2018). For this we would need an expertly created bilingual lexicon—essentially a word aligned version of the data. For Chen and Cardie (2018)'s tool, we need a tab separated file where each column has exactly one from Latin, and one word from English. This does not exist for Latin and English. Indeed, a direct word to word correspondence does not exist between Latin and any of its English translations. What we do have is a vocabulary list from Bridge.[1] With some modification, we force this into the correct format, splitting separate definitions of the same word on to their own line, and making multi-word definitions a single word with underscores. We evaluate our model both with our hacked bilingual lexicon, and qualitatively. We provide this lexicon as well to make our results replicable.

6.1 Quantitative Analysis

Many of the words in the Bridge dictionary had multi-word phrases or explanations for a singular Latin word, meaning that the evaluator function provided could not match them (as it expected a single word to single word mapping). The Williams translation seems particularly hard hit by this. However, with access to more neighbors, the Williams translation scores by far the highest. Low precision values at $k = 1$ are expected with the low word-word correspondence, but they are still lower than expected. We suspect the dramatically higher precision values at higher k mean that it is able to use larger phrase groups well to make more accurate guesses, with few false positives. Chen and Cardie

[1] https://bridge.haverford.edu/select/Latin/result/vergil_aeneid/start-end/non_running/

	Unique Words Not Matched	Unique Words Matched	kNN k =1	kNN k =5	kNN k =10
Williams	5937	437	0	0.457666	0.915332
Conington	5114	1081	0.092507	0.462535	0.555042
Dryden	5055	1102	0.181488	0.272232	0.635209

Table 1: Note that nearly all the English translations had over 5,000 words that did not match our simplistic lexicon. Some of those are probably whole phrases used to translate a single word, as English translators must often do with Latin and poetry in general.

(2018) do not provide a way to test for kNN recall—assessing false negatives. No one else has tried this particular task, so we have nothing to compare it to directly. But getting a zero on a kNN precision measure is bad even with no reference of comparison. We were unable to test the non-lemmatized Latin to English model in the same way, due to so few of the surface word forms being their lemma. Future work with Latin embeddings should imitate Sprugnoli et al. (2019), and strictly use lemmatized text.

6.2 Qualitative Analysis

We find some of these matches by manually examining nearest neighbors, but not many. The best we found in Dryden's translation was *imperium*, which had paired to *commands*, *commanded* and *command*. While these are close, the similarity measure is still low. We checked the other translations for a few of the frequent words, because frequent words should have higher quality embeddings. We did not find others that immediately made sense the way that Dryden's *imperium* does. We see some others that are close to making sense. For example, in the Conington-Latin embedding space, *breath* is the second closest neighbor to *vox*. We would expect it to be most similar to *voice*. We see more of these in Figure .

CosineSim	Word
0.4928	commands
0.4927	commanded
0.4835	command
0.4572	unite

Table 2: Words close to *imperium* when mapped to Dryden's English. While the cosine similarity itself is not great, it is good that these are the closest words, since command is the core meaning of *imperium*.

It seems some words are indeed close to their definitions, as black is direct translation to ater—but even many translation pairs that seem uncomplicated like *flamma* and *flame* are nowhere near each other. In a good multilingual space, words that mean the same thing in different languages, should be close to each other. There is clearly at least one fault here. Yet, there are interesting truths captured here. Notice the pair $honor_{latin}$ and *command*. While the core meaning of *honor* is quite similar in Latin and English, in Latin it is also heavily associated with political office, the higher levels of which have *imperium*. In that context, *imperium* mainly means command of an army. This is a similarity that we expect to be captured in a much larger corpus, but not from our small corpus here, especially given the quality of the rest of the embedding space. Perhaps their proximity is coincidence.

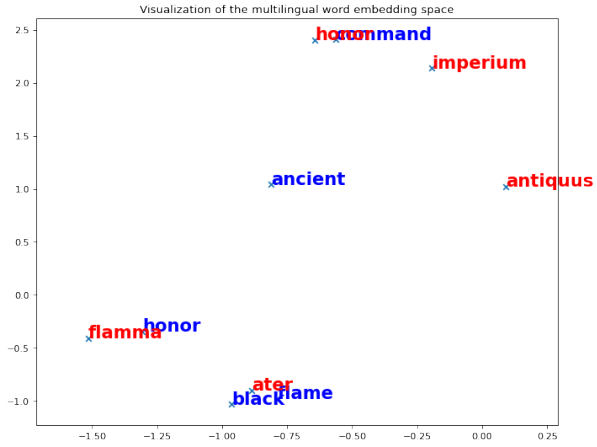

Figure 1: A visualization of the multilingual word embedding space with source language of Dryden's English (words denoted with blue) and target language of Latin (words denoted as red)

6.3 Error Analysis

We did not leverage the work of Herbelot and Baroni (2019) as much as we could have in our final implementation. (Bojanowski et al., 2017) and (Chen and Cardie, 2018) code work so well together and at such a high level that it was hard to fit in other's work. In the end, we were only able to have a high overall learning rate decay over time. We did not have the robust background embeddings

from a Wikipedia corpus the way that they did. We could possibly use Sprugnoli et al. (2019)'s embeddings for some background, but their embeddings are still from fairly small data to begin with, and their data come from works that span over a thousand years. The high learning rate, and high learning rate decay do seem to have helped, but this could be investigated even further. Aligning six 300 dimensional, 5000 to 6000 word embedding spaces with Chen and Cardie (2018) took almost 2 hours on a GPU, which was longer than initially anticipated.

We could have more thoroughly analyzed the monolingual Latin embeddings before trying to make the bilingual ones. Sprugnoli et al. (2019) created a Latin synonym set to test their embeddings, but because they trained on a much larger, more diverse corpus, they were not suitable for our use: many words were out of vocabulary, and other's meanings shifted. In the future, we should curate their synonym set so we have a better metric for validating the Latin monolingual embeddings, before moving them into a multilingual space. Similarly, we could have done more robust analysis of the English monolingual embeddings, to give us a better idea of their quality. We did some informal qualitative analysis as we experimented with learning rate and changing the other hyper parameters, but this was highly subjective, and could only get us some good vectors.

7 Future Work and Conclusion

It would benefit the whole research community if we successfully re-implemented Trans-gram and provided our implementation of it. Coulmance et al. (2015)'s big promised advantage over Gouws et al. (2015) was that it did not require explicitly aligned sentences, and claimed to out perform it. Once we realized we needed this kind of data anyway, sticking with Gouws et al. (2015) may have been better, because their process is more similar to Coulmance et al. (2015) than Chen and Cardie (2018) is.

If we were to keep our model architecture as is, and use the tools provided in Chen and Cardie (2018), it is clear we need to improve our simplistic bilingual lexicon. An option is to combine multiple Latin-English dictionaries, such that we could get more pairs of words that could be matched by our model. In terms of adjusting our model, if we had sentence-aligned data to evaluate on instead of generating it ourselves, it would likely improve our precision due to sentences aligning much better than individual words.

Another interesting area of future work for digitally inclined classicists could be modifying not only our lexicon, but also the training data of Sprugnoli et al. (2019) to leverage FastText's advantages even more. As it is, we notice that lemmatized words sometimes end up close to words that just have similar prefixes and endings. In Latin, the common prefixes (like ad, a, pre, prae, in, con, per, and trans) largely correspond to prepositions. They almost always act as intensifiers, with slightly different connotations. That FastText over capitalizes on these similarities is evident from qualitative analysis. It would be interesting to see what would happen if we split the prefixes from their base words in the lemmatized data set, and treated them more like enclitics than really part of the word.

Overall, this is more a proof of concept than a major break through. We demonstrate the feasibility of training embeddings from a small work of literature, and aligning a literary translation with the original work, and provide code for how to do it again. With sentence aligned data to properly train and evaluate a supervised model, and a larger background dictionary, we expect better, more interpretable, results. As it is, we cannot find a correlation between the high precision k-NN and any qualitative, understandable results. We have some interesting results that seem like good starting points for further literary analysis. but our Latin, and possibly even English monolingual embeddings are too inconsistent to be sure. At this point, the failure seems to be more in the lack of a good evaluation metric, and lack of the data needed to fit existing ones, than the small data. We got ahead of ourselves, and probably would have gotten better results if we had stuck to making monolingual embeddings from small data, following Bloem et al. (2020) and Herbelot and Baroni (2019) more closely than Bojanowski et al. (2017) and Chen and Cardie (2018).

We present our lexicon as a start, not an end to making at evaluating English-Latin bilingual pairs. We present our Google Colaboratory in the **??** as a full pipeline for training bilingual Latin-English and embedding spaces. We suspect the largest improvement would come from properly merging this with pipelines for training and evaluating monolingual embeddings from tiny data.

References

Jelke Bloem, Antske Fokkens, and Aurélie Herbelot. 2019. Evaluating the consistency of word embeddings from small data. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 132–141.

Jelke Bloem, Maria Chiara Parisi, Martin Reynaert, Yvette Oortwijn, and Arianna Betti. 2020. Distributional semantics for neo-latin. In *Proceedings of LT4HALA 2020 - 1st Workshop on Language Technologies for Historical and Ancient Languages*, pages 84–93. European Language Resources Association (ELRA).

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

Xilun Chen and Claire Cardie. 2018. Unsupervised multilingual word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 261–270.

Jocelyn Coulmance, Jean-Marc Marty, Guillaume Wenzek, and Amine Benhalloum. 2015. Trans-gram, fast cross-lingual word-embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1109–1113. Association for Computational Linguistics.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2015. Bilbowa: Fast bilingual distributed representations without word alignments. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 748–756, Lille, France. PMLR.

Aurélie Herbelot and Marco Baroni. 2019. High-risk learning: acquiring new word vectors from tiny data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 304–309. Association for Computational Linguistics.

Karl Moritz Hermann and Phil Blunsom. 2014. Multilingual models for compositional distributed semantics. *arXiv preprint arXiv:1404.4641*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013b. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Rachele Sprugnoli, Marco Passarotti, and Giovanni Moretti. 2019. Vir is to moderatus as mulier is to intemperans lemma embeddings for latin. page 7.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. 2016. Cross-lingual models of word embeddings: An empirical comparison. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1670. Association for Computational Linguistics.

Appendix

Code

Google Drive with Code