Deep Neural Networks for Bilingual Lexicon Extraction

Jon Gauthier,* **Arthur Tsang,†** **Christopher Potts‡**
* Symbolic Systems Program, †Department of Computer Science, ‡Department of Linguistics
Stanford University, Stanford, CA 94035
{jgauthie,atsang2,cgpotts}@stanford.edu

Abstract

We explore a method for producing bilingual lexicons from non-parallel corpora. In contrast with most previous approaches, our method separates the steps of (1) constructing distributed word representations from monolingual corpora, and (2) building a translational mapping.

This gives us precise control over the independent development of the word representations and the mapping between them.

We demonstrate that a deep neural network can effectively learn a translational mapping between these representations for the purposes of bilingual lexicon extraction more accurately than a closely related linear model.

1 Introduction

In bilingual lexicon extraction (BLE), we accept as input some pair of corpora in different languages—perhaps parallel or perhaps not—and use them to learn associations of translational equivalence between a *source language* and a *target language*.

Bilingual lexicon extraction is of most utility when applied to under-resourced language pairs which do not already benefit from a surplus of lexicon materials. These same language pairs, lacking resources as simple as quality translation dictionaries, also lack sufficient amounts of manually aligned text to train BLE models. For this reason, most recent work on this task focuses on minimally supervised approaches which do not require parallel corpora (Rapp, 1995; Peirsman and Padó, 2010).

Rapp (1995) suggested that there is "a correlation between the patterns of word co-occurrences in different languages." In line with this hypothesis, recent work has related the likelihood of two words being translations of one another to the similarities between the contexts in which the words appear in their corresponding languages. The standard approach under this theory is to manually construct a *bilingual vector space* which contains distributed representations of words in both the source and target language (Fung and Yee, 1998; Vulić and Moens, 2013). The axes of this space are built from bilingual seed pairs. A given word representation has a large value along an axis if it co-occurs often with the seed word corresponding to that axis. Because axes are defined with bilingual pairs, word representations for both source and target language words can lie in the same space. With a bilingual vector space constructed, translations for a given source-language word can be retrieved by simply finding the nearest target-language word representation neighboring the word's own vector representation in the bilingual vector space.

The present paper is motivated by the hypothesis that this standard bilingual vector space approach to BLE is inherently limited by the design of the space itself. We suggest that the construction of the space along axes composed of a hand-picked set of bilingual seeds has the potential to limit the expressiveness of the word representations contained therein. Consider, for example, a vector space constructed in this fashion using just two seeds: *hot* and *cold*. Such a space may be effective at representing the difference between words like *fire* and *ice*. However, with these two axes it would fail to capture many other distinctions, such as the difference between the words *up* and *down*. The quality of the vector space is thus heavily dependent on the quality of its seeds.

Mikolov et al. (2013b) suggest an alternate method for bilingual lexicon extraction, centered around two monolingual vector spaces E (corresponding to the source language) and F (corresponding to the target language). These vector spaces are constructed using a standard neural language model (Mikolov et al., 2013a). This is a

promising alternative to the aforementioned bilingual vector space approach, as it allows for the construction of robust distributed representations whose quality is not dependent on manually selected seeds. The authors find that a linear model trained via stochastic gradient descent to map between these two vector spaces can effectively find word-level translations from the source to the target language. We extend this work in attempting to approximate a *translation function* $t : E \rightarrow F$ using a deep neural network, trained to accept as input word vectors from the source language space E and output word vectors in the target language space F. We capitalize on the fact that these vector spaces may be independently constructed, experimenting with different configurations for each space.

2 Data

Our translation algorithm makes use of two monolingual corpora: one consisting of text in the source language and the other containing text in the target language. We require only that the corpora have the following properties:

Large and broad-domain. With this property given, we can safely assume that for any common word which appears in the source-language corpus, we will also find its translation somewhere in the target-language corpus. If large corpora are unavailable, we can make use of smaller narrow-domain corpora (e.g., religious texts, famous novels).

Non-parallel. This means that our algorithm does not depend on any sort of alignment, lexical or topical. This relative flexibility allows us to demonstrate that the method can be applied in situations where parallel corpora are not available.

In our experiments we use English and Spanish corpora sourced from Wikipedia.[1] We choose not to take advantage of the document alignments available in the Wikipedia corpus in order to show that our method is generalizable to other corpora.

In addition to the two monolingual corpora, we train on a minimal set of word translation pairs. The specifics of these training pairs are described in Section 3.2.

3 Model

Figure 1 illustrates the process of translating a single word from a source language to a target language (here English and Spanish, respectively) in this model. We begin with an input source-language word and retrieve a corresponding precomputed word representation in the source VSM. We use this source-language word vector as an input to a neural network, which outputs a vector in the target-language VSM. We perform a simple nearest-neighbor search in the target-language VSM to find the closest existing word vector, and take its corresponding target-language word to be the translation of the original source-language word.

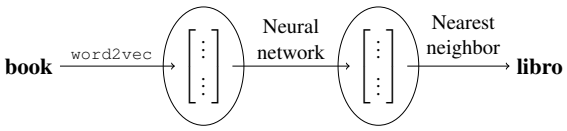

Figure 1: The process of word translation, using a neural network to map between two monolingual VSMs

3.1 Word representations

For each of the two corpora introduced in Section 2, we train a continuous vector space model on 10-word contexts, using an implementation in the *Gensim* software package of `word2vec`, a neural language model (Řehůřek and Sojka, 2010; Mikolov et al., 2013a). This neural language model attempts to learn word representations which minimize error in the task of predicting a word's context given the word itself. The resulting word vectors are thus theoretically optimal (i.e., minimum-cost) representations of the meanings underlying their corresponding words. This is in contrast with the word representations built by a standard BLE approach, which define themselves in terms of manually constructed axes, and thus do not necessarily account for a maximal amount of variance in meaning among the words represented. We experiment with different word vector dimensionalities, and detail our specific choices later in Section 4.2.

While we focus on constructing a neural language model with `word2vec`, we believe any distributional model of word meaning would suffice for this task (Bengio et al., 2003; Turian et al., 2010; Collobert et al., 2011).

3.2 Translation with a neural network

The central computation is the transformation of a source-language word representation into a target-language word representation via a deep neural

[1] Available at `http://dumps.wikimedia.org/`.

network. We learn this transformation from a small set of training pairs.

The standard bilingual vector space approach detailed in the introduction would base its word representations on these training pairs. In contrast with this method, our model uses these translation pairs only in its final learning step, while training the neural network which models the translation function. For this reason, the model can benefit from effective word representations developed independently from the translation system itself. Furthermore, it can provide direct feedback through a training cost function on the quality of the translation pairs.

We investigate several different methods of producing translation pairs. In line with previous approaches, we fill a set of translation pairs `Cog` with cognates shared between the two languages (Koehn and Knight, 2002). We also build a small separate set `ConcNN` consisting of concrete nouns, and attempt to restrict the set to word pairs for which both the source- and target-language words have just one sense in their respective languages.

For each training pair of words (e, f), we provide as a training example the input and output (w_e, w_f), where w_e and w_f are word representations drawn from their respective VSMs constructed as described in Section 3.1.

At inference time, for a given English word e, we provide as input to the neural network the corresponding word representation w_e. The output of the network, w_f, is some vector in the target-language vector space. We search for the nearest known neighbor of w_f, and treat the word corresponding to this nearest neighbor as the translation f of the input word e.

3.3 Comparison: Translation with a linear transformation

Mikolov et al. (2013b) found that a linear model could learn a translation mapping between two monolingual vector spaces. We construct such a linear model for purposes of comparison with the neural network method detailed above. It utilizes the same basic idea as presented in Figure 1: it only differs in the type of transformation used to map between the two monolingual vector spaces. Whereas our approach uses a deep neural network, this linear model builds a linear transformation between the two spaces. Given source-language and target-language training data matrices W_E, W_F (where the ith columns of the two matrices are the vector word representations corresponding to the ith training pair) we assert the existence of a translation matrix A, where $W_F = AW_E$.

We compute the linear transformation by taking $A = W_F W_E^+$, where W_E^+ is the Moore-Penrose pseudoinverse of W_E. It is necessary to use the pseudoinverse of W_E, as the matrix is unlikely to be square and invertible. Then given any source-language word vector w_e, we calculate the transformed target-language word vector as $w_f = Aw_e$.

4 Evaluation

We evaluate our translation model on an unseen set of English-Spanish word pairs. These test sets are customized to match the algorithm's training environment. For example, a translation algorithm trained on concrete nouns from the training set `ConcNN` will be tested on other unseen concrete nouns. Since our model builds a ranked list of nearest-neighbor translation candidates, we can calculate the mean reciprocal rank (MRR) of the correct translation within the list:

$$MRR = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{r_j} \quad (1)$$

where r_j is the rank of the correct translation for pair j out of n. For example, a mean reciprocal rank of 1 would indicate that every translation output ranked the correct translation in the first position. A mean reciprocal rank of $\frac{1}{2}$ would indicate that the average translation output placed the correct translation as the second most likely in a ranked list.

4.1 Baseline

We establish a lower bound for translation performance with an algorithm that associates source- and target-language words by their relative corpus frequencies. Given an input source-language word, we compute its corpus frequency percentile. We then predict translations in the target language by returning the words with the closest percentile frequencies in the target language corpus.

4.2 Test data

We establish three separate test environments, each defined by a choice of training data, input corpus format, and VSM design. The vector space dimensionalities given below were derived through trial and error. We find that different VSM dimensionalities are optimal for different corpus inputs. Overall,

Model	**Cog**	**ConcNN**	**CogLem**	**CogFix**
NN (125)	0.216	0.199	0.215	0.389
NN (250)	**0.295**	0.193	0.269	**0.480**
NN (500)	0.264	**0.203**	0.279	0.432
NN (1000)	0.263	0.196	0.265	0.451
Linear	0.280	0.195	**0.286**	0.449
Baseline	0.006	0.002	0.002	0.006

Table 2: Performance on multiple test environments (hidden layer dimensionalities given in parentheses). MRR values (higher is better).

we see that performance varies significantly based on these test environment properties. The environments are as follows:

Cog: Minimally preprocessed corpora, Cog training data, 500-dimensional word vectors

ConcNN: Minimally preprocessed corpora, ConcNN training data, 500-dimensional word vectors

CogLem: Lemmatized Spanish corpus, Cog training data, 1000-dimensional word vectors

Our evaluation metric looks for a single possible translation of a test input word in ranked output. As we explain in Section 4.4, this may needlessly penalize the algorithm, as it often suggests reasonable translations earlier in the ranked output which happen to not match our strict search requirements. For example, our algorithm is tested on a cognate pair (*concept*, *concepto*), and is penalized for suggesting the translation *idea*, a suitable synonym of the desired target word. A fourth test run **CogFix** considers the algorithm's output on the Cog dataset, and accepts such sensible alternate translations (rather than stipulating in this case that all source-language cognates translate to their corresponding target-language cognates).

4.3 Results

Mean reciprocal rank values for each test environment and a number of training conditions are in Table 2. The linear transformation model based on Mikolov et al. (2013b) was very competitive with the neural network model. This suggests that many—but not all—of the geometric relationships in the target language vector space can be modeled through a simple linear transform of the geometric structures in the source language space.

The best absolute performance is achieved by a neural network with 250 hidden-layer neurons in the **CogFix** training environment. Interestingly, the hidden layer of this model is of lower dimensionality than the test input and output word vectors (each 500 dimensions). This suggests that the model learns some useful compressed intermediate representation of the input source-language word in the translation training process. Its MRR shows us that on average the algorithm suggests an acceptable translation in the second or third position of the ranked list of possible translations.

4.4 Error analysis

While the translation algorithm fails to predict the exactly correct translation a significant amount of the time, an analysis of common prediction errors is encouraging. We identify several common error patterns in the algorithm's output in Table 1, and elaborate on the classes of errors below:

Synonyms. With many of the "errors" of this form, the algorithm provided an acceptable alternate translation that we simply did not expect. These cases are treated as successes in the **CogFix** training environment, as described in Section 4.3.

Near misses. In these cases the algorithm mapped into a precise neighborhood of related words, but picked the wrong element from the set. For example, the output word *chocolate* is strongly related to the expected word *vainilla*.

Right neighborhood, wrong spot. We occasionally mapped into a correct larger area of the target language vector space, but still made a significant semantic error. With the input word *election*, for example, the algorithm predicts as translations target-language words related to political campaigns.

Alternate senses. In several cases on the Cog training set and its derivatives, the algorithm learned to predict translations of the input words which corresponded to valid alternate interpretations of the source language word. For example, it interprets the English *division* in the sense of a military unit (rather than in the mathematical sense as we expected). These cases were not treated as successes in the **CogFix** training environment. We hope to eliminate this class of errors in future work by using multi-prototype VSMs (see Section 5).

Diagnosis	Input	Expected cognate output	Predicted ranked output (translations)
Synonyms	*acceptance*	*aceptación*	*aprobación (approval), ...*
Near misses	*vanilla*	*vainilla*	*chocolate, vainilla, ...*
Right neighborhood, wrong spot	*election*	*elección*	*candidato, candidatura (candidacy), ...*
Alternate sense interpretations	*division*	*división*	*unidad (unit), élite, comando (command), ...*

Table 1: Examples of common error patterns observed in the algorithm's output for the **Cog** environment

5 Conclusion

In this paper, we presented an algorithm which learns word translations from unlabeled, non-parallel corpora by building a mapping between two independently constructed monolingual vector spaces. This algorithm can be used to perform bilingual lexicon extraction in scenarios where the source and target language in question do not benefit from widely available parallel corpora. We demonstrated encouraging translation performance with large, broad-domain corpora as text sources.

For simplicity, our current VSM design associates a single word vector with each word in the source or target language corpus. However, training error could be greatly reduced by allowing polysemous words to be associated with multiple word vectors (Reisinger and Mooney, 2010; Huang et al., 2012). We believe such an extension will remove much of the remaining ambiguity in the input vector spaces.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, February.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Pascale Fung and Lo Yuen Yee. 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 17th International Conference on Computational linguistics-Volume 1*, pages 414–420. Association for Computational Linguistics.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 873–882, Stroudsburg, PA, USA. Association for Computational Linguistics.

Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition-Volume 9*, page 9–16. Association for Computational Linguistics.

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*.

Yves Peirsman and Sebastian Padó. 2010. Cross-lingual induction of selectional preferences with bilingual vector spaces. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 921–929. Association for Computational Linguistics.

Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 320–322. Association for Computational Linguistics.

Joseph Reisinger and Raymond J. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 109–117, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden, July. ACL.

Ivan Vulić and Marie-Francine Moens. 2013. A study on bootstrapping bilingual vector spaces from non-parallel data (and nothing else). In *Proceedings of EMNLP 2013: Conference on Empirical Methods in Natural Language Processing*.

Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 workshop on New Challenges for NLP Frameworks*, pages 46–50, Valletta, Malta. University of Malta.