Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

Handling Missing Words by Mapping Across Word Vector Representations

Rajendra Banjade, Nabin Maharjan, Dipesh Gautam,Vasile Rus

Department of Computer Science / Institute for Intelligent Systems
The University of Memphis, USA
{rbanjade, nmhrjan, dgautam, vrus}@memphis.edu

Abstract

Vector based word representation models are often developed from very large corpora. However, we often encounter words in real world applications that are not available in a single vector model. In this paper, we present a novel Neural Network (NN) based approach for obtaining representations for words in a target model from another model, called the source model, where representations for the words are available, effectively pooling together their vocabularies. Our experiments show that the transformed vectors are well correlated with the native target model representations and that an extrinsic evaluation based on a word-to-word similarity task using the Simlex-999 dataset leads to results close to those obtained using native model representations.

Introduction

Different approaches have been proposed over the years to represent the semantics of words, phrases, sentences, or even larger texts in continuous vector representations (Landauer et al., 1998; Turian et al., 2010; Collobert et al., 2011; Mikolov et al. 2013b; Pennington et al., 2014). These vector representations have been widely used in many NLP applications (Manning 2008; Collobert et al., 2011; Lei et al., 2014; Socher et al., 2013; Banjade et al., 2015).

Preferably, and which is often the case, meaning representations of words are derived in an unsupervised way from extremely large collections of texts. For instance, the pre-trained word2vec (Mikolov et al. 2013b) and GloVe (Pennington et al., 2014) word vector representations trained on texts containing billions of tokens and cover millions of unique words: the pre-trained word2vec model covers 3 million unique words, and the GloVe model has coverage of 1.9 million words (see Data section for specific details about the models). Similarly, an LSA model developed from the whole set of Wikipedia articles (LSAwiki)[1] (Stefanescu et al., 2014; Rus et al., 2013) contains word representations for 1.1 million words.

While these are impressive numbers compared to manually created resources such as WordNet (Miller, 1995), it is interesting to observe that the previously mentioned unsupervised vector models share a limited number of words, as illustrated next. The GloVe and word2vec have about 154,000 words in common. Only about 107,000 words are common to all three models, which equates to only 3 to 10% of the words depending which model's vocabulary size is used as a reference. This clearly indicates that a significant chunk of words in each of these models are unique to the respective models and that they are missing from the other models. One question arises about how to handle this acute problem of missing words in one particular model and the related question of how to make the best use of existing word vector representations in various models as together they cover a significantly larger vocabulary, i.e. 5,226,598 unique words altogether in the three models discussed above.

In this paper, we provide a solution to the above issue of missing word representations in vector based models. The basic idea is to train Neural Network models to map word vector representations from one model (where they are present; the source model) to another (where they are missing; the target model). That is, we exploit existing resources in combination with the NN-based mapping approach to extend the coverage of a given target model such that there will be less words missing in that model when used by real world applications. For instance, given an arbitrary word X and let's assume that it is not present in a model A but is present in model B. By developing a transformation model from B to A, we can obtain the representation of word X in model A. The benefit of our approach is that we extend the coverage of a target model without

[1]Wiki_NVAR_f7 at http://semanticsimilarity.org/

the need to collect any extra texts and retrain the model, which is non-trivial.

We intrinsically and extrinsically evaluated our approach (c.f. Evaluation Methodology) on source-target model permutations of two different pre-trained word vector models: word2vec, and GloVe. The results show that it is possible to obtain word representation for one model from another without loss of meaning representation power relative to the native target vectors.

Related Work

Research in the area of vector based meaning representations has gained momentum in recent years. However, only a few research works that address the problem of handling missing word representations are found in the literature.

Bengio et al. (2003) and Alexandrescu and Kirchhoff (2006) proposed deriving continuous word representations for unknown or missing words in Neural Language Models (NLMs) based on the words in context. However, (full) context of a word is not always available and the process can be computationally costly. Mikolov et al. (2013a) demonstrated that word2vec vectors capture enough syntactic and semantic linguistic regularity to derive vector representations of missing words based on simple vector operations. For example, the following expression illustrates a singular/plural relation: *v('cats') = v('dogs') – v('dog') + v('cat')*. However, such nice linguistic features might not hold for complex and rare words and their vector representations might not be properly estimated (Luong et al., 2013). It will also not work if certain word representations that are needed on the right hand side of an expression like the one above are not available. Recursive Neural Networks (RNNs) have also been used to construct missing word representations from the vectors of its morphemes (Luong et. al., 2013). This approach works only if the missing word can be broken into morphemes and representations for morphemes are available. Banjade et al. (2015) used representation of one of the synonyms when the given word is missing. Though this approach works well for the words available in the thesaurus, such as WordNet, we generally need more than that. For instance, there can be millions of named entities that are not found in a single model and also they do not have the notion of synonyms.

In our approach, we directly transform representations from one model to another model effectively making any single model covering large vocabulary.

Approach

As discussed earlier, words missing from a target model may be present in another model, called the source model. Therefore by learning a word vector mapping model that can map one vector representation onto another, representations for missing words in the target model can be obtained from a source model where the word is present.

As illustrated in Figure 1, we have developed Neural Network based models to learn a mapping function from one representation model to another. The input to the model is in the form of source model vectors (SrcV) and the output of the transformation model (TrV) is similar to target model vectors (TgV). The source vectors and target vectors can be of different types. For instance, the source vectors can be Glove vector representations while the target vectors can be word2vec vector representations and vice versa. Also, the dimensionality of the source and target vectors may be different.

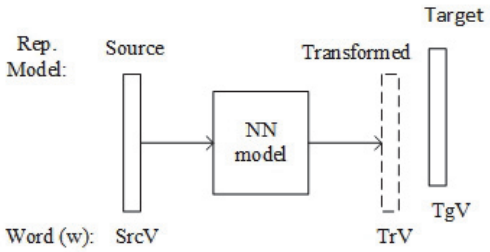

Fig 1: Schematic diagram of transformation model.

Specifically, we learned NNs to map between two models: word2Vec model and GloVe model. It is important to note that these basic models are quite different in their underlying principles to derive meaning representations and that they both are unsupervised (see Data section for additional details).

Evaluation Methodology

We evaluated our mapping approach intrinsically and extrinsically. We used a simulation based approach in both cases, i.e. we simulated a set of missing words from the target model by removing them from it. This simulation based approach enables us to assess the transformed representations with respect to the native representations in target model.

For **intrinsic evaluation**, we calculated the average correlation (AvgCorr) across a set of words, where the correlation for a given word was computed between the corresponding values in the transformed vector (TrV) and the native target vector (TgV). We chose as our simulated missing words a set of words that are present in both the source and target models so that the transformed vectors could be directly compared to the native vectors, i.e. the representations generated using the underlying target model itself. In the ideal case, the transformed vectors would be same as the native vectors.

For **extrinsic evaluation**, we used a word-to-word similarity task which is one of the approaches to measure the quality of word meaning representations. We used word pairs from the Simlex-999 dataset (described in Data section) and computed their similarities using the normalized dot product (cosine) based on the transformed vector representations and also the target representations. Then, correlations between the similarity scores and human judgments were computed in each case.

Baseline. We have also generated results using a baseline method which randomly selects vectors from the source model to be mapped onto the target model using our trained NN mapping model. In fact, we preselected these random vectors before training the NN mapping model such that none of the randomly selected vectors for the words in Simlex-999 dataset would be used for training. These random baselines help detect whether the system is actually learning something or is simply a random mapping.

Data

Simlex-999 (Simlex; Hill et al., 2014) is a recently released dataset for word-to-word similarity evaluation. In this dataset, related but semantically less equivalent word pairs are rated with low similarity scores by human judges. The dataset consists of 999 word pairs.

Word2vec[2]: word2vec is a neural-probabilistic word representations model developed by Mikolov et al. (2013). We used the 300-dimensional word vectors developed by training distributed representations of words with the Skip-gram model on part of Google News dataset (about 100 billion words).

GloVe[3]: it is a word representation model proposed by Pennington et al. (2013) and the model we used was trained on 42 billion Common Crawl words. The corpus was built over several years of web crawling. We used the 300-dimensional word representation model.

Training, validation, and test datasets. We used existing, pre-trained models (those discussed above in this section) for word2vec, and GloVe from which we extracted 106,028 vectors corresponding to the common words in both models and excluding 1,028 (unique) words of the Simlex dataset. A randomly selected set of 95,000 pairs of vectors was used for training, and 5,000 pairs for validation. We chose to use only the common words to both models to train, validate, and test the transformation models because it allows simulating missing words and comparing the transformed output with the native target model vectors for evaluation purpose. In ideal case, the vectors obtained using transformation model should look alike the actual target model vectors. Then 1028 Simlex words were used for both intrinsic as well as extrinsic evaluation. The remaining 1,028 vectors from the common vocabulary were used for developing the baseline transformations.

[2] https://code.google.com/p/word2vec/

[3] http://nlp.stanford.edu/data/glove.42B.300d.zip

Experiments and Results

We built NN models with a number of input units and output units equal to the size of the vectors in corresponding source and target models, respectively. They both were 300-dimension vectors (which was a pure coincidence and not a constraint of our mapping model). Therefore, the number of input units and output units were 300.

We added only one hidden layer keeping the models relatively simple and performed experiments with varying number of hidden units. We used the neural network toolbox in Matlab (R2015a; Demuth & Beale, 1993) to build the models. The NN learning algorithm was set to the Scaled Conjugate Gradient Algorithm (Moller, 1993) and the number of iterations was fixed to 1,000. The activation function used was the logistic function.

Source-Target	Word Sim. correlations		AvgCorr (TrV-TgV)
	TgSim	TrSim	
Baseline Systems	-	~ 0.000	0.0-0.12
Word2vec-GloVe	0.427	0.360	0.663
GloVe-Word2vec	0.469	0.398	0.644

Table 1: Word-to-Word similarity results (Pearson correlation, r) with Simlex data and average correlation (AvgCorr) between transformed vectors (TrV) and target vectors (TgV)

Each source-target transformation model was trained using the training dataset of 95k pairs of vectors. The performance (AvgCorr) on the validation set was used to calibrate the number of hidden units in the NN models. We experimented with different number of hidden units from 100 to 800 incrementing by 100. The results were improving with an increasing number of hidden units up to 600. Therefore, we chose to use 600 hidden unit models for the experiment for all pairs of source-target models. However, the differences among the results with 400-600 hidden units were very small and in order to reduce the complexity of the model (or risk of overfitting), the number of hidden units could be set to 500 or 400 with small reduction in performance. We then evaluated the learned models on the test data. The results are summarized in Table 1.

The TgSim column presents the Pearson (*r*) correlations between the word similarities computed using target vectors and the human annotated similarity scores, respectively, for the word pairs in the Simlex dataset. The TrSim column shows the same correlations for word similarities but this time using transformed vectors, thus, indicating

how well the transformed vectors can act as a substitute for word representations in the target model.

The word-to-word similarity results using the transformed vectors (TrSim) are comparable or better in some cases with the results obtained using the native target model vectors (TgSim). For instance, the word-similarity correlation between human similarity judgments and the similarity scores obtained using the native GloVe vectors is 0.427 while the correlation obtained using transformed vectors from the word2vec model is close at 0.360.

The average correlation scores of TrVs with corresponding TgVs (in AvgCorr columns) were 0.663 for Simlex words. These correlation scores indicate that the transformed vectors closely resemble the target model vectors.

Results for the baseline transformations are presented as a range because the results were similar for the two different transformations corresponding to the two combinations of of source-target models. The highest average correlation (AvgCorr) was 0.12 for the GloVe to word2vec transformation of Simlex words. That means that providing random vectors from the source model as input or using randomly selected words for missing words in the target model has no significant outcome. Furthermore, we checked the direct correlation between native source and target vectors but it was approximately zero when tested on the Simlex words, indicating that learning a mapping function is needed

Conclusions

This paper showed that the neural network model can be used effectively to map from one word representation to another. Such a mapping that vastly increases the coverage of a target model can be very useful in many NLP applications which most likely need to handle missing words. Additionally, the proposed solution can be used to obtain phrase representations which are even sparser than words, a topic for future research.

In the future, we intend to apply this approach in other types of word representation models, such as LSA and analyze situations where this approach works best.

References

Alexandrescu, A., and Katrin K. (2006). "Factored neural language models." *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*.

Banjade, R., Niraula, N. B., Maharjan, N., Rus, V., Stefanescu, D., Lintean, M., & Gautam, D. (2015). NeRoSim: A System for Measuring and Interpreting Semantic Textual Similarity. *SemEval-2015*, 164.

Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, *3*, 1137-1155.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, *12*, 2493-2537.

Demuth, H., & Beale, M. (1993). Neural network toolbox for use with MATLAB.

Gabrilovich, Evgeniy, and Shaul Markovitch. "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis." *IJCAI*. Vol. 7. 2007.

Hill, F., Reichart, R., & Korhonen, A. (2014). Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *arXiv preprint arXiv:1408.3456*.

Landauer, Thomas K., Peter W. Foltz, and Darrell Laham. "An introduction to latent semantic analysis." *Discourse processes* 25.2-3 (1998): 259-284.

Lei, Tao, Yu Xin, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. "Low-rank tensors for scoring dependency structures." In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 1381-1391. 2014.

Luong, Minh-Thang, Ilya Sutskever, Quoc V. Le, Oriol Vinyals, and Wojciech Zaremba. "Addressing the rare word problem in neural machine translation." In *Proceedings of ACL*. 2015.

Luong, Minh-Thang, Richard Socher, and Christopher D. Manning. "Better word representations with recursive neural networks for morphology." *CoNLL-2013*104 (2013).

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Vol. 1. Cambridge: Cambridge university press, 2008.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39-41.

Moller, *Neural Networks*, Vol. 6, 1993, pp. 525–533

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, *12*, 1532-1543.

Rus, V., Lintean, M. C., Banjade, R., Niraula, N. B., & Stefanescu, D. (2013, August). SEMILAR: The Semantic Similarity Toolkit. In *ACL (Conference System Demonstrations)* (pp. 163-168).

Socher, R., Huang, E. H., Pennin, J., Manning, C. D., & Ng, A. Y. (2011). Dynamic pooling and unfolding recursive auto encoders for paraphrase detection. In *Advances in Neural Information Processing Systems* (pp. 801-809).

Ştefănescu, D., Banjade, R., & Rus, V. (2014). Latent semantic analysis models on wikipedia and tasa. *LREC 2014.*

Turian, J., Ratinov, L., & Bengio, Y. (2010, July). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics* (pp. 384-394).