Re-embedding Words

Igor Labutov
Cornell University
iil4@cornell.edu

Hod Lipson
Cornell University
hod.lipson@cornell.edu

Abstract

We present a fast method for re-purposing existing semantic word vectors to improve performance in a supervised task. Recently, with an increase in computing resources, it became possible to learn rich word embeddings from massive amounts of unlabeled data. However, some methods take days or weeks to learn good embeddings, and some are notoriously difficult to train. We propose a method that takes as input an existing embedding, some labeled data, and produces an embedding in the same space, but with a better predictive performance in the supervised task. We show improvement on the task of sentiment classification with respect to several baselines, and observe that the approach is most useful when the training set is sufficiently small.

1 Introduction

Incorporating the vector representation of a word as a feature, has recently been shown to benefit performance in several standard NLP tasks such as language modeling (Bengio et al., 2003; Mnih and Hinton, 2009), POS-tagging and NER (Collobert et al., 2011), parsing (Socher et al., 2010), as well as in sentiment and subjectivity analysis tasks (Maas et al., 2011; Yessenalina and Cardie, 2011). Real-valued word vectors mitigate sparsity by "smoothing" relevant semantic insight gained during the unsupervised training over the rare and unseen terms in the training data. To be effective, these word-representations — and the process by which they are assigned to the words (i.e. embedding) — should capture the semantics relevant to the task. We might, for example, consider *dramatic* (term X) and *pleasant* (term Y) to correlate with a review of a good movie (task A), while finding them of opposite polarity in the context of a dating profile (task B). Consequently, good vectors for X and Y should yield an inner product close to 1 in the context of task A, and -1 in the context of task B. Moreover, we may already have on our hands embeddings for X and Y obtained from yet another (possibly unsupervised) task (C), in which X and Y are, for example, orthogonal. If the embeddings for task C happen to be learned from a much larger dataset, it would make sense to reuse task C embeddings, but adapt them for task A and/or task B. We will refer to task C and its embeddings as the *source task* and the *source embeddings*, and task A/B, and its embeddings as the *target task* and the *target embeddings*.

Traditionally, we would learn the embeddings for the target task jointly with whatever unlabeled data we may have, in an instance of semi-supervised learning, and/or we may leverage labels from multiple other related tasks in a multi-task approach. Both methods have been applied successfully (Collobert and Weston, 2008) to learn task-specific embeddings. But while joint training is highly effective, a downside is that a large amount of data (and processing time) is required a-priori. In the case of deep neural embeddings, for example, training time can number in days. On the other hand, learned embeddings are becoming more abundant, as much research and computing effort is being invested in learning word representations using large-scale deep architectures trained on web-scale corpora. Many of said embeddings are published and can be harnessed in their raw form as additional features in a number of supervised tasks (Turian et al., 2010). It would, thus, be advantageous to learn a task-specific embedding directly from another (source) embedding.

In this paper we propose a fast method for re-embedding words from a source embedding S to a target embedding T by performing unconstrained optimization of a convex objective. Our objective is a linear combination of the dataset's log-

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 489–493,
Sofia, Bulgaria, August 4-9 2013.

likelihood under the target embedding and the Frobenius norm of the distortion matrix — a matrix of component-wise differences between the target and the source embeddings. The latter acts as a regularizer that penalizes the Euclidean distance between the source and target embeddings. The method is much faster than joint training and yields competitive results with several baselines.

2 Related Work

The most relevant to our contribution is the work by Maas *et.al* (2011), where word vectors are learned specifically for sentiment classification. Embeddings are learned in a semi-supervised fashion, and the components of the embedding are given an explicit probabilistic interpretation. Their method produces state-of-the-art results, however, optimization is non-convex and takes approximately 10 hours on 10 machines[1]. Naturally, our method is significantly faster because it operates in the space of an existing embedding, and does not require a large amount of training data a-priori.

Collobert and Weston (2008), in their seminal paper on deep architectures for NLP, propose a multilayer neural network for learning word embeddings. Training of the model, depending on the task, is reported to be between an hour and three days. While the obtained embeddings can be "fine-tuned" using backpropogation for a supervised task, like all multilayer neural network training, optimization is non-convex, and is sensitive to the dimensionality of the hidden layers.

In machine learning literature, joint semi-supervised embedding takes form in methods such as the LaplacianSVM (LapSVM) (Belkin et al., 2006) and Label Propogation (Zhu and Ghahramani, 2002), to which our approach is related. These methods combine a discriminative learner with a non-linear manifold learning technique in a joint objective, and apply it to a combined set of labeled and unlabeled examples to improve performance in a supervised task. (Weston et al., 2012) take it further by applying this idea to deep-learning architectures. Our method is different in that the (potentially) massive amount of unlabeled data is not required a-priori, but only the resultant embedding.

[1] as reported by author in private correspondence. The runtime can be improved using recently introduced techniques, see (Collobert et al., 2011)

3 Approach

Let $\Phi_S, \Phi_T \in \mathbb{R}^{|V| \times K}$ be the source and target embedding matrices respectively, where K is the dimension of the word vector space, identical in the source and target embeddings, and V is the set of embedded words, given by $V_S \cap V_T$. Following this notation, ϕ_i – the i^{th} row in Φ – is the respective vector representation of word $w_i \in V$. In what follows, we first introduce our supervised objective, then combine it with the proposed regularizer and learn the target embedding Φ_T by optimizing the resulting joint convex objective.

3.1 Supervised model

We model each document $d_j \in D$ (a movie review, for example) as a collection of words w_{ij} (i.i.d samples). We assign a sentiment label $s_j \in \{0, 1\}$ to each document (converting the star rating to a binary label), and seek to optimize the conditional likelihood of the labels $(s_j)_{j \in \{1,...,|D|\}}$, given the embeddings and the documents:

$$p(s_1, ..., s_{|D|} | D; \Phi_T) = \prod_{d_j \in D} \prod_{w_i \in d_j} p(s_j | w_i; \Phi_T)$$

where $p(s_j = 1 | w_i, \Phi_T)$ is the probability of assigning a positive label to document j, given that $w_i \in d_j$. As in (Maas et al., 2011), we use logistic regression to model the conditional likelihood:

$$p(s_j = 1 | w_i; \Phi_T) = \frac{1}{1 + \exp(-\psi^T \phi_i)}$$

where $\psi \in \mathbb{R}^{K+1}$ is a regression parameter vector with an included bias component. Maximizing the log-likelihood directly (for ψ and Φ_T), especially on small datasets, will result in severe overfitting, as learning will tend to commit neutral words to either polarity. Classical regularization will mitigate this effect, but can be improved further by introducing an external embedding in the regularizer. In what follows, we describe *re-embedding regularization*— employing existing (source) embeddings to bias word vector learning.

3.2 Re-embedding regularization

To leverage rich semantic word representations, we employ an external *source* embedding and incorporate it in the regularizer on the supervised objective. We use Euclidean distance between the source and the target embeddings as the regular-

ization loss. Combined with the supervised objective, the resulting log-likelihood becomes:

$$\underset{\psi,\Phi_T}{\text{argmax}} \sum_{d_j \in D} \sum_{w_i \in d_j} \log p(s_j|w_i;\Phi_T) - \lambda||\Delta\Phi||_F^2 \quad (1)$$

where $\Delta\Phi = \Phi_T - \Phi_S$, $||\cdot||_F$ is a Frobenius norm, and λ is a trade-off parameter. There are almost no restrictions on Φ_S, except that it must match the desired target vector space dimension K. The objective is convex in ψ and Φ_T, thus, yielding a unique target re-embedding. We employ L-BFGS algorithm (Liu and Nocedal, 1989) to find the optimal target embedding.

3.3 Classification with word vectors

To classify documents, re-embedded word vectors can now be used to construct a document-level feature vector for a supervised learning algorithm of choice. Perhaps the most direct approach is to compute a weighted linear combination of the embeddings for words that appear in the document to be classified, as done in (Maas et al., 2011) and (Blacoe and Lapata, 2012). We use the document's binary bag-of-words vector v_j, and compute the document's vector space representation through the matrix-vector product $\Phi_T v_j$. The resulting $K+1$-dimensional vector is then cosine-normalized and used as a feature vector to represent the document d_j.

4 Experiments

Data: For our experiments, we employ a large, recently introduced IMDB movie review dataset (Maas et al., 2011), in place of the smaller dataset introduced in (Pang and Lee, 2004) more commonly used for sentiment analysis. The dataset (50,000 reviews) is split evenly between training and testing sets, each containing a balanced set of highly polar (≥ 7 and ≤ 4 stars out of 10) reviews.
Source embeddings: We employ three external embeddings (obtained from (Turian et al., 2010)) induced using the following models: 1) hierarchical log-bilinear model (HLBL) (Mnih and Hinton, 2009) and two neural network-based models – 2) Collobert and Weston's (C&W) deep-learning architecture, and 3) Huang *et.al*'s polysemous neural language model (HUANG) (Huang et al., 2012). C&W and HLBL were induced using a 37M-word newswire text (Reuters Corpus 1). We also induce a Latent Semantic Analysis (LSA) based embedding from the subset of the English project Gutenberg collection of approximately 100M words. No pre-processing (stemming or stopword removal), beyond case-normalization is performed in either the external or LSA-based embedding. For HLBL, C&W and LSA embeddings, we use two variants of different dimensionality: 50 and 200. In total, we obtain seven source embeddings: HLBL-50, HLBL-200, C&W-50, C&W-200, HUANG-50, LSA-50, LSA-200.
Baselines: We generate two baseline embeddings – NULL and RANDOM. NULL is a set of zero vectors, and RANDOM is a set of uniformly distributed random vectors with a unit L2-norm. NULL and RANDOM are treated as source vectors and re-embedded in the same way. The NULL baseline is equivalent to regularizing on the target embedding without the source embedding. As additional baselines, we use each of the 7 source embeddings directly as a target without re-embedding.
Training: For each source embedding matrix Φ_S, we compute the optimal target embedding matrix Φ_T by maximizing Equation 1 using the L-BFGS algorithm. 20 % of the training set (5,000 documents) is withheld for parameter (λ) tuning. We use LIBLINEAR (Fan et al., 2008) logistic regression module to classify document-level embeddings (computed from the $\Phi_T v_j$ matrix-vector product). Training (re-embedding and document classification) on 20,000 documents and a 16,000 word vocabulary takes approximately 5 seconds on a 3.0 GHz quad-core machine.

5 Results and Discussion

The main observation from the results is that our method improves performance for smaller training sets (≤ 5000 examples). The reason for the performance boost is expected – classical regularization of the supervised objective reduces overfitting. However, comparing to the NULL and RANDOM baseline embeddings, the performance is improved noticeably (note that a percent difference of 0.1 corresponds to 20 correctly classified reviews) for word vectors that incorporate the source embedding in the regularizer, than those that do not (NULL), and those that are based on the random source embedding (RANDOM). We hypothesize that the external embeddings, generated from a significantly larger dataset help "smooth" the word-vectors learned from a small labeled dataset alone. Further observations include:

Features	Number of training examples					
				+ Bag-of-words features		
	.5K	5K	20K	.5K	5K	20K
A. Re-embeddings (our method)						
HLBL-50	74.01	79.89	80.94	78.90	84.88	85.42
HLBL-200	74.33	80.14	81.05	79.22	85.05	85.95
C&W-50	74.52	79.81	80.48	78.92	84.89	85.87
C&W-200	**74.80**	**80.25**	**81.15**	**79.34**	**85.28**	**86.15**
HUANG-50	74.29	79.90	79.91	79.03	84.89	85.61
LSA-50	72.83	79.67	80.67	78.71	83.44	84.73
LSA-200	73.70	80.03	80.91	79.12	84.83	85.31
B. Baselines						
RANDOM-50 w/ re-embedding	72.90	79.12	80.21	78.29	84.01	84.87
RANDOM-200 w/ re-embedding	72.93	79.20	80.29	78.31	84.08	84.91
NULL w/ re-embedding	72.92	79.18	80.24	78.29	84.10	84.98
HLBL-200 w/o re-embedding	67.88	72.60	73.10	79.02	83.83	85.83
C&W-200 w/o re-embedding	**68.17**	**72.72**	**73.38**	**79.30**	**85.15**	**86.15**
HUANG-50 w/o re-embedding	67.89	72.63	73.12	79.13	84.94	85.99
C. Related methods						
Joint training (Maas, 2011)	—	—	84.65	—	—	88.90
Bag of Words SVM	—	—	—	79.17	84.97	86.14

Table 1: Classification accuracy for the sentiment task (IMDB movie review dataset (Maas et al., 2011)). Subtable A compares performance of the re-embedded vocabulary, induced from a given source embedding. Subtable B contains a set of baselines: *X-w/o re-embedding* indicates using a source embedding X directly without re-embedding.

BORING	
source:	lethal, lifestyles, masterpiece ...
target:	**idiotic**, **soft-core**, **gimmicky**
BAD	
source:	past, developing, lesser, ...
target:	**ill**, **madonna**, **low**, ...
DEPRESSING	
source:	versa, redemption, townsfolk ...
target:	**hate**, **pressured**, **unanswered** ,
BRILLIANT	
source:	high-quality, obsession, hate ...
target:	**all-out**, **bold**, **smiling** ...

Table 2: A representative set of words from the 20 closest-ranked (cosine-distance) words to (*boring*, *bad*, *depressing*, *brilliant*) extracted from the *source* and *target* (C&W-200) embeddings. Source embeddings give higher rank to words that are related, but not necessarily indicative of sentiment, e.g. *brilliant* and *obsession*. Target words tend to be tuned and ranked higher based on movie-sentiment-based relations.

Training set size: We note that with a sufficient number of training instances for each word in the test set, additional knowledge from an external embedding does little to improve performance.

Source embeddings: We find C&W embeddings to perform best for the task of sentiment classification. These embeddings were found to perform well in other NLP tasks as well (Turian et al., 2010).

Embedding dimensionality: We observe that for HLBL, C&W and LSA source embeddings (for all training set sizes), 200 dimensions outperform 50. While a smaller number of dimensions has been shown to work better in other tasks (Turian et al., 2010), re-embedding words may benefit from a larger initial dimension of the word vector space. We leave the testing of this hypothesis for future work.

Additional features: Across all embeddings, appending the document's binary bag-of-words representation increases classification accuracy.

6 Future Work

While "semantic smoothing" obtained from introducing an external embedding helps to improve performance in the sentiment classification task, the method does not help to re-embed words that do not appear in the training set to begin with. Returning to our example, if we found *dramatic* and *pleasant* to be "far" in the original (source) embedding space, but re-embed them such that they are "near" (for the task of movie review sentiment classification, for example), then we might expect words such as *melodramatic*, *powerful*, *striking*, *enjoyable* to be re-embedded nearby as well, even if they did not appear in the training set. The objective for this optimization problem can be posed by requiring that the distance between every pair of words in the source and target embeddings is preserved as much as possible, i.e. min $(\hat{\phi}_i\hat{\phi}_j - \phi_i\phi_j)^2 \quad \forall i, j$ (where, with some abuse of notation, ϕ and $\hat{\phi}$ are the source and target embeddings respectively). However, this objective is no longer convex in the embeddings. Global re-embedding constitutes our ongoing work and may pose an interesting challenge to the community.

7 Conclusion

We presented a novel approach to adapting existing word vectors for improving performance in a text classification task. While we have shown promising results in a single task, we believe that the method is general enough to be applied to a range of supervised tasks and source embeddings. As sophistication of unsupervised methods grows, scaling to ever-more massive datasets, so will the representational power and coverage of induced word vectors. Techniques for leveraging the large amount of unsupervised data, but indirectly through word vectors, can be instrumental in cases where the data is not directly available, training time is valuable and a set of easy low-dimensional "plug-and-play" features is desired.

8 Acknowledgements

This work was supported in part by the NSF CDI Grant ECCS 0941561 and the NSF Graduate fellowship. The content of this paper is solely the responsibility of the authors and does not necessarily represent the official views of the sponsoring organizations. The authors would like to thank Thorsten Joachims and Bishan Yang for helpful and insightful discussions.

References

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *The Journal of Machine Learning Research*, 7:2399–2434.

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

William Blacoe and Mirella Lapata. 2012. A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 546–556. Association for Computational Linguistics.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

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

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

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*, pages 873–882. Association for Computational Linguistics.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 142–150. Association for Computational Linguistics.

Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. *Advances in neural information processing systems*, 21:1081–1088.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.

Joseph Turian, Lev 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. Association for Computational Linguistics.

Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. 2012. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer.

Ainur Yessenalina and Claire Cardie. 2011. Compositional matrix-space models for sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 172–182. Association for Computational Linguistics.

Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University.