Reconstruction of Word Embeddings from Sub-Word Parameters

Karl Stratos
Toyota Technological Institute at Chicago
stratos@ttic.edu

Abstract

Pre-trained word embeddings improve the performance of a neural model at the cost of increasing the model size. We propose to benefit from this resource without paying the cost by operating strictly at the sub-lexical level. Our approach is quite simple: before task-specific training, we first optimize sub-word parameters to reconstruct pre-trained word embeddings using various distance measures. We report interesting results on a variety of tasks: word similarity, word analogy, and part-of-speech tagging.

1 Introduction

Word embeddings trained from a large quantity of unlabled text are often important for a neural model to reach state-of-the-art performance. They are shown to improve the accuracy of part-of-speech (POS) tagging from 97.13 to 97.55 (Ma and Hovy, 2016), the F1 score of named-entity recognition (NER) from 83.63 to 90.94 (Lample et al., 2016), and the UAS of dependency parsing from 93.1 to 93.9 (Kiperwasser and Goldberg, 2016). On the other hand, the benefit comes at the cost of a bigger model which now stores these embeddings as additional parameters.

In this study, we propose to benefit from this resource without paying the cost by operating strictly at the sub-lexical level. Specifically, we optimize the character-level parameters of the model to reconstruct the word embeddings prior to task-specific training. We frame the problem as distance minimization and consider various metrics suitable for different applications, for example Manhattan distance and negative cosine similarity.

While our approach is simple, the underlying learning problem is a challenging one; the sub-word parameters must reproduce the topology of word embeddings which are not always morphologically coherent (e.g., the meaning of `fox` does not follow any common morphological pattern). Nonetheless, we observe that the model can still learn useful patterns. We evaluate our approach on a variety of tasks: word similarity, word analogy, and POS tagging. We report certain, albeit small, improvement on these tasks, which indicates that the word topology transformation based on pre-training can be beneficial.

2 Related Work

Faruqui et al. (2015) "retrofit" embeddings against semantic lexicons such as PPDB or WordNet. Cotterell et al. (2016) leverage existing morphological lexicons to incorporate sub-word components. The aim and scope of our work are clearly different: we are interested in training a strictly sub-lexical model that only operates over characters (which has the benefit of smaller model size) and yet somehow exploit pre-trained word embeddings in the process.

Our work is also related to *knowledge distillation* which refers to training a smaller "student" network to perform better by learning from a larger "teacher" network. We adopt this terminology and refer to pre-trained word embeddings as the teacher and sub-lexical embeddings as the student. This problem has mostly been considered for classification and framed as matching the probabilities of the student to the probabilities of the teacher (Ba and Caruana, 2014; Li et al., 2014; Kim and Rush, 2016). In contrast, we work directly with representations in Euclidean space.

3 Reconstruction Method

Let $\mathcal{W}$ denote the set of word types. For each word $w \in \mathcal{W}$, we assume a pre-trained word embedding $x^w \in \mathbb{R}^d$ and a representation $h^w \in \mathbb{R}^d$ computed by sub-word model parameters Θ; we defer how to define h^w until later. The reconstruction error with respect to a distance function $D : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is given by

$$L_D(\Theta) = \sum_{w \in \mathcal{W}} D(x^w, h^w) \tag{1}$$

where x^w is constant and h^w is a function of Θ. Since we use gradient descent to optimize (1), we can define $D(u, v)$ to be any continuous function measuring the discrepency between u and v, for example,

$$D_1(u,v) := \sum_{i=1}^{d} |u_i - v_i| \qquad \text{(Manhattan)}$$

$$D_{\sqrt{2}}(u,v) := \sqrt{\sum_{i=1}^{d} |u_i - v_i|^2} \qquad \text{(Euclidean)}$$

$$D_2(u,v) := \sum_{i=1}^{d} |u_i - v_i|^2 \qquad \text{(squared error)}$$

$$D_\infty(u,v) := \max_{i=1}^{d} |u_i - v_i| \qquad (l_\infty \text{ distance})$$

$$D_{\cos}(u,v) := \frac{-u^\top v}{||u||_2 \, ||v||_2} \qquad \text{(negative cosine)}$$

Unlike other common losses used in the neural network literature such as negative log likelihood or the hinge loss, L_D has a direct geometric interpretation illustrated in Figure 1. We first optimize (1) over sub-word model parameters Θ for a set number of epochs, and then proceed to optimize a task-specific loss $L(\Theta, \Theta')$ where Θ' denotes all other model parameters.

3.1 Analysis of a Linear Model

In general, h^w can be a complicated function of Θ. But we can gain some insight by analyzing the simple case of a linear model, which corresponds to the top layer of a neural network. More specifically, we assume the form

$$h_i^w = \theta_i^\top z^w \qquad \forall i = 1 \ldots d$$

where $z^w \in \mathbb{R}^{d'}$ is fixed and $\Theta = \{\theta_1 \ldots \theta_d\} \subset \mathbb{R}^{d'}$ is the only parameter to be optimized.

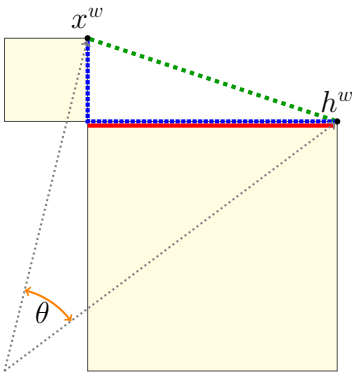

Figure 1: Geometric losses corresponding to different distance metrics: Manhattan distance (blue), Euclidean distance (green), squared error (yellow), l_∞ distance (red), and negative cosine similarity ($-\cos\theta$).

Manhattan distance The error $L_{D_1}(\Theta)$ is now

$$L_{D_1}(\Theta) = \sum_{w \in \mathcal{W}} \sum_{i=1}^{d} \left| x_i^w - \theta_i^\top z^w \right| = \sum_{i=1}^{d} \text{LAD}_i(\theta_i)$$

where $\text{LAD}_i(\theta) := \sum_{w \in \mathcal{W}} \left| x_i^w - \theta_i^\top z^w \right|$ is least absolute deviations (LAD). It is well-known that the LAD criterion is robust to outliers. To see this, if $z^w = (1/d')\mathbf{1}$ for all $w \in \mathcal{W}$, then a minimizer of $\text{LAD}_i(\theta)$ is given analytically by

$$\theta_i^* = \text{median}\{x_i^w : w \in \mathcal{W}\}$$

where the median resists extreme values (e.g., the median of both $\{1, 2, 3\}$ and $\{1, 2, 999\}$ is 2). Thus using Manhattan distance can be useful when teacher word embeddings are noisy or there are occasional exceptions in morphological patterns that are best ignored.

Squared error The error $L_{D_2}(\Theta)$ is now

$$L_{D_2}(\Theta) = \sum_{w \in \mathcal{W}} \sum_{i=1}^{d} \left| x_i^w - \theta_i^\top z^w \right|^2 = \sum_{i=1}^{d} \text{OLS}_i(\theta_i)$$

where $\text{OLS}_i(\theta) := \sum_{w \in \mathcal{W}} \left| x_i^w - \theta^\top z^w \right|^2$ is ordinary least squares (OLS). Thus if the matrix $Z \in \mathbb{R}^{|\mathcal{W}| \times d'}$ with z^w as rows has rank d', the unique solution is given by $\theta_i^* = \left(Z^\top Z\right)^{-1} Z^\top x_i^w$. Let $\bar{h}_i^w = (\theta_i^*)^\top z^w$ denote the optimal sub-word embedding value. It is well-known that the change in $\bar{h}_i^w$ caused by removing x_i^w from the dataset is proportional to the residual $x_i^w - \bar{h}_i^w$ (Davidson et al.,

1993). In other words, squared error is sensitive to outliers and may not be as suitable as Manhattan distance for fitting noisy or incoherent word embeddings.

Other distance metrics Euclidean distance is geometrically intuitive but less mathematically convenient than squared error, thus we choose not to focus on it. l_∞ distance penalizes the dimension with maximum absolute difference and can be useful if calculating one coordinate at a time is convenient. Finally, negative cosine similarity penalizes the angle between embeddings. This is suitable when we only care about directions and not magnitude, for instance in word similarity where we measure cosine similarities between word embeddings.

There are distance metrics not discussed here that may be appropriate in certain situations. For instance, the KL divergence is a natural (assymetric) measure if word embeddings are distributions (e.g., over context words). More generally, we can consider the wide class of metrics in the Bregman divergence (Banerjee et al., 2005).

4 Sub-Word Architecture

We now describe how we define word embedding $h^w \in \mathbb{R}^d$ from sub-word parameters. We use a character-based embedding scheme closely following Lample et al. (2016). We use an LSTM simply as a mapping $\phi : \mathbb{R}^d \times \mathbb{R}^{d'} \to \mathbb{R}^{d'}$ that takes an input vector x and a state vector h to output a new state vector $h' = \phi(x, h)$. See Hochreiter and Schmidhuber (1997) for a detailed description.

4.1 Character Model

Let $\mathcal{C}$ denote the set of character types. The model parameters Θ associated with this layer are

- $e^c \in \mathbb{R}^{d_c}$ for each $c \in \mathcal{C}$
- Character LSTMs $\phi^{\mathcal{C}}_f, \phi^{\mathcal{C}}_b : \mathbb{R}^{d_c} \times \mathbb{R}^{d_c} \to \mathbb{R}^{d_c}$
- $W^f, W^b \in \mathbb{R}^{d \times d_c}$, $b^{\mathcal{C}} \in \mathbb{R}^d$

Let $w(j) \in \mathcal{C}$ denote the character of $w \in \mathcal{W}$ at position j. The model computes $h^w \in \mathbb{R}^d$ as

$$
\begin{aligned}
f^{\mathcal{C}}_j &= \phi^{\mathcal{C}}_f\left(e^{w(j)}, f^{\mathcal{C}}_{j-1}\right) && \forall j = 1 \dots |w| \\
b^{\mathcal{C}}_j &= \phi^{\mathcal{C}}_b\left(e^{w(j)}, b^{\mathcal{C}}_{j+1}\right) && \forall j = |w| \dots 1 \\
z^w &= W^f f^{\mathcal{C}}_{|w|} + W^b b^{\mathcal{C}}_1 + b^{\mathcal{C}} \\
h^w_i &= \max\{0, z^w_i\} \quad \forall i = 1 \dots d && (2)
\end{aligned}
$$

We also experiment with a highway network (Srivastava et al., 2015) which has been shown to be beneficial for image recognition (He et al., 2015) and language modeling (Kim et al., 2016). In this case, Θ includes additional parameters $W^{\text{highway}} \in \mathbb{R}^{d \times d}$ and $b^{\text{highway}} \in \mathbb{R}^d$. A new character-level embedding $\tilde{h}^w$ is computed as

$$
\begin{aligned}
t &= \sigma\left(W^{\text{highway}} h^w + b^{\text{highway}}\right) \\
\tilde{h}^w &= t \odot h^w + (\mathbf{1} - t) \odot z^w \qquad (3)
\end{aligned}
$$

where $\sigma(\cdot) \in [0, 1]$ denotes an element-wise sigmoid function and $\odot$ the element-wise multiplication. This allows the network to skip nonlinearity by making t_i close to 0. We find that the additional highway network is beneficial in certain cases. We will use either (2) or (3) in our experiments depending on the task.

5 Experiments

Implementation We implement our models using the DyNet library. We use the Adam optimizer (Kingma and Ba, 2014) and apply dropout at all LSTM layers (Hinton et al., 2012). For POS tagging and parsing, we perform a 5×5 grid search over learning rates $0.0001 \dots 0.0005$ and dropout rates $0.1 \dots 0.5$ and choose the configuration that gives the best performance on the dev set. We use the highway network (3) for word analogy and parsing and (2) for others. Note that the character embedding dimension d_c must match the dimension of the pre-trained word embeddings.

Teacher Word Embeddings We use 100-dimensional word embeddings identical to those used by Dyer et al. (2015) which are computed with a variant of the skip n-gram model (Ling et al., 2015). These embeddings have been shown to be effective in various tasks (Dyer et al., 2015; Lample et al., 2016).

5.1 Word Similarity and Analogy

Data For word similarity, we use three public datasets WordSim-353, MEN, and Stanford Rare Word. Each contains 353, 3000, and 2034 word pairs annotated with similarity scores. The evaluation is conducted by computing the cosine of the angle θ between each word pair (w_1, w_2) under the model (2):

$$
\cos(\theta) = \frac{\left(h^{w_1}\right)^\top h^{w_2}}{||h^{w_1}||_2 \, ||h^{w_2}||_2} \qquad (4)
$$

	number of reconstruction epochs				
metric	0	10	20	30	50
D_1	0.03	0.09	0.11	0.12	0.13
D_2	0.03	0.12	0.12	0.14	**0.15**
D_∞	0.03	0.12	0.10	0.09	0.10
$D_{\cos}$	0.03	**0.13**	**0.15**	**0.15**	**0.15**

Table 1: Effect of reconstruction on word similarity: the teacher word embeddings obtain score 0.50.

and computing the Spearman's correlation coefficient with the human scores. We report the average correlation across these datasets. For word analogy, we use the 8000 syntactic analogy questions from the dataset of Mikolov et al. (2013b) and 8869 semantic analogy questions from the dataset of Mikolov et al. (2013a). We use the multiplicative technique of Levy and Goldberg (2014) for answering analogy questions.

Result Table 1 shows word similarity scores for different numbers of reconstruction training epochs. The teacher word embeddings obtain 0.5. The sub-word model improves performance from the initial score of 0.03 up to 0.16. In particular, the negative cosine distance metric which directly optimizes the relevant quantity (4) is consistently best performing.

Table 2 shows the accuracy on the syntactic and semantic analogy datasets. An interesting finding in our experiment is that for syntactic analogy, **a randomly initialized character-based model outperforms the pre-trained embeddings** and thus reconstruction only decreases the performance. We suspect that this is because much of the syntactic regularities is already captured by the architecture. Many questions involves only simplistic transformation, for instance adding `r` in `wise : wiser` $\sim$ `free` $: x$. The model correctly answers such questions simply by following its architecture, though it is unable to answer less regular questions (e.g., `see : saw` $\sim$ `keep` $: x$).

Semantic analogy questions have no such morphological regularities (e.g., `Athens : Greece` $\sim$ `Havana` $: x$) and are challenging to sub-lexical models. Nonetheless, the model is able to make a minor improvement in accuracy.

5.2 POS Tagging

We perform POS tagging on the Penn WSJ treebank with 45 tags using a BiLSTM model described in Lample et al. (2016). Given a vector sequence $(v^{w_1} \dots v^{w_n})$ corresponding to a sentence $(w_1 \dots w_n) \in \mathcal{W}^n$, the BiLSTM model produces feature vectors $(h_1 \dots h_n)$. We adhere to the simplest approach of making a local prediction at each position i by a feedforward network on h_i,

$$p(t_i|h_i) \propto \exp\left(W^2 f(W^1 h_i + b^1) + b^2\right)$$

where $f_i(v) = \max\{0, v_i\}$ and W^1, W^2, b^1, b^2 are additional parameters. The model is trained by optimizing log likelihood. We consider the following choices of v^w:

Embedding	Syntactic	Semantic
random	**65.21**	1.13
D_1	26.32	2.20
D_2	27.56	**2.47**
D_∞	45.68	1.74
$D_{\cos}$	23.77	2.22
teacher	57.42	59.58

Table 2: Effect of reconstruction on word analogy (10 reconstruction epochs).

model	accuracy	lookup
FULL	97.20	43211
FULL+EMB	97.32	252365
CHAR	96.93	80
CHAR(D_1)	**97.17**	93
CHAR(D_2)	97.08	93
CHAR(D_∞)	97.06	93
CHAR($D_{\cos}$)	97.08	93

Table 3: POS tagging accuracy with different definitions of v^w (see the main text). The final column shows the number of lookup parameters.

- FULL: $v^w = e^w \oplus h^w$ uses both word-level lookup parameter e^w and character-level embedding h^w (2).
- FULL+EMB: Same as FULL but the lookup parameters e^w are initialized with pre-trained word embeddings.
- CHAR: $v^w = h^w$ uses characters only.
- CHAR(D): Same as CHAR but optimized for 10 epochs to reconstruct pre-trained word embeddings with distance metric D.

Table 3 shows the accuracy of these models. We see that pre-trained word embeddings boost the

beautiful	wonderful	prettiest	gorgeous	smartest	jolly	famous	sensual
	baleful	bagful	basketful	bountiful	boastful	bashful	behavioural
	bountiful	peaceful	disdainful	perpetual	primaeval	successul	purposeful
amazing	incredible	wonderful	remarkable	terrific	marvellous	astonishing	unbelievable
	awaking	arming	aging	awakening	angling	agonizing	among
	arousing	amusing	awarding	applauding	allaying	awaking	assaying
Springfield	Glendale	Kennesaw	Gainesville	Lynchburg	Youngstown	Kutztown	Harrisburg
	Spanish-ruled	Serbian-held	Serbian-led	Spangled	Serbian-controlled	Schofield	Sharif-led
	Stubblefield	Smithfield	Stansfield	Butterfield	Littlefield	Bitterfeld	Sinfield

Table 4: Nearest neighbor examples: for each word, the three rows respectively show its nearest neighbors using pre-trained word embeddings, student embeddings at random initialization (3), and student embeddings optimized for 10 epochs using D_1.

performance of FULL from 97.20 to 97.32. When we use the strictly character-based model CHAR without reconstruction, the performance drops to 96.93. But with reconstruction, the model recovers some of the lost accuracy. In particular, reconstructing with the Manhattan distance metric gives the largeset improvement and yields 97.17.

5.3 Analysis of Student Embeddings

Table 4 shows examples of nearest neighbors. For each example, the first row corresponds to the teacher, the second to the student (3) at random initialization, and the third to the student optimized for 10 epochs using D_1. The student embeddings at random initialization are already capable of capturing morphological regularities such as `-ful` and `-ing`. With reconstruction, there is a subtle change in the topology. For instance, the nearest neighbors of `beautiful` change from `baleful` and `bagful` to `bountiful` and `peaceful`. For `Springfield`, nearest neighbors change from unrelated words such as `Spanish-ruled` to fellow nouns such as `Stubblefield`.

6 Conclusion

We have presented a simple method for a sub-lexical model to leverage pre-trained word embeddings. We have shown that by recontructing the embeddings before task-specific training, the model can improve over random initialization on a variety of tasks. The reconstruction task is a challenging learning problem; while our model learns useful patterns, it is far from perfect. An important future direction is to improve reconstruction with other choices of architecture.

References

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*. pages 2654–2662.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. 2005. Clustering with bregman divergences. *Journal of machine learning research* 6(Oct):1705–1749.

Lyle Campbell and Mauricio J Mixco. 2007. *A glossary of historical linguistics*. Edinburgh University Press.

Jason PC Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308* .

Ryan Cotterell, Hinrich Schütze, and Jason Eisner. 2016. Morphological smoothing and extrapolation of word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. volume 1, pages 1651–1660.

Russell Davidson, James G MacKinnon, et al. 1993. Estimation and inference in econometrics .

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075* .

Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of NAACL*.

Alex Graves. 2012. Neural networks. In *Supervised Sequence Labelling with Recurrent Neural Networks*, Springer, pages 15–35.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385* .

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947* .

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics* 4:313–327.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360* .

Geunbae Lee, Jong-Hyeok Lee, and Kyunghee Kim. 1994. Phonemie-level, speech and natural, language integration for agglutinative languages. *GGGGGGGG 0* .

Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Computational Natural Language Learning*. page 171.

Jinyu Li, Rui Zhao, Jui-Ting Huang, and Yifan Gong. 2014. Learning small-size dnn with output-distribution-based criteria. In *INTERSPEECH*. pages 1910–1914.

Wang Ling, Lin Chu-Cheng, Yulia Tsvetkov, and Silvio Amir. 2015. Not all contexts are created equal: Better word representations with variable attention .

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* .

Ryan T McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al. 2013. Universal dependency annotation for multilingual parsing. In *ACL (2)*. pages 92–97.

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, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *HLT-NAACL*. volume 13, pages 746–751.

Sakriani Sakti, Andrew Finch, Ryosuke Isotani, Hisashi Kawai, and Satoshi Nakamura. 2010. Korean pronunciation variation modeling with probabilistic bayesian networks. In *Universal Communication Symposium (IUCS), 2010 4th International*. IEEE, pages 52–57.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681.

Jae Jung Song. 2006. *The Korean language: Structure, use and context*. Routledge.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387* .

Karl Stratos, Michael Collins, and Daniel Hsu. 2015. Model-based word embeddings from decompositions of count matrices. 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)*. Association for Computational Linguistics, Beijing, China, pages 1282–1291.