**CSLT TECHNICAL REPORT-20150033 [JANUARY 6, 2016]**

# Learning Ordered Word Representations

Xiaoxi Wang*, Chao Xing, Dong Wang, Rong Liu and Yiqiao Pan

*Correspondence:
wxx@cslt.riit.tsinghua.edu.cn
Center for Speech and Language Technology, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China
Full list of author information is available at the end of the article

**Abstract**

Learning distributed word representations, or word embeddings, has gained much popularity. Current learning approaches treat all dimensions of the embeddings as homogeneous, which leads to non-structured representations where the dimensions are neither interpretable nor comparable. This paper presents ordered word embeddings where the significance of the dimensions is in descending order. The order in dimensions may benefit a wide range of applications such as fast search and vector tailor.

Three algorithms are proposed to learn the ordered embeddings, based on dropout, learning rate decay and sparse penalties, respectively. Additionally, a sweeping approach based on the $\chi^2$ distribution is proposed to ensure sufficient training for all dimensions. The experimental results on the WordSimilarity-353 task confirmed that the proposed methods indeed produce ordered embeddings, and better performance can be achieved with ordered representations when compared to the non-ordered counterparts.

**Keywords:** word embedding; deep neural network; nested dropout

## 1 Introduction

Learning distributed representations for words has gained much attention [1, 2, 3, 4, 5, 6]. These distributed representations, or word embeddings, represent words in a low-dimensional continuous space, where semantic and syntactic structures can be easily defined and inferred. Current learning approaches treat the embedding space as dimensional homogeneous, which leads to non-structured representations where the dimensions are neither interpretable nor comparable. Moreover, learning all dimensions simultaneously may trap in suboptimal local minima, considering the complex semantic and syntactic relations among words. The key idea of this paper is to impose some heterogeneous constrains on the dimensions of embeddings so that the low dimensions learn more significant information than high dimensions. This leads to ordered word representations where the significance of the dimensions is in descending order. The order in dimensions is desirable for many applications, for example fast search and vector tailor. Additionally, the ordered learning offers an incremental optimization method that can learn better representations than the conventional non-ordered learning.

We propose three approaches to learn ordered word embeddings. In the dropout approach, lower dimensions are dropped out with lower probabilities; in $\alpha$-decay, lower dimensions are assigned larger learning rates; in $\lambda$-growth, higher dimensions are penalized by more aggressive $\ell$-1 regularization. All these approaches encourage lower dimensions to learn more effectively and so more significant in the resultant embeddings.

A particular problem in ordered learning is that high dimensions tend to be less trained. This paper proposes a new sweeping algorithm which smoothly adjusts the learning rate function following the $\chi^2$ distribution.

The paper is organized as follows: Section 2 describes the related works, and Section 3 presents the ordered learning algorithms. The sweeping methods are presented in Section 4, and the entire paper is concluded by Section 5.

## 2 Related Works

The idea of ordered representations is largely inspired by the work proposed by [7], where the authors learned ordered representations by auto-encoders. The ordered learning is achieved by a nested dropout approach that randomly removes units of the bottle-neck layer, where lower dimensions are removed with smaller probabilities. This approach has been implemented in this paper to achieve ordered word representations. However, it is highly inefficient since only part of the dimensions are updated for each training data. The $\alpha$-decay and $\lambda$-growth algorithms proposed in this work update all the dimensions and so lead to more efficient learning.

## 3 Ordered Learning

Our algorithms are developed based upon the CBOW model proposed by [4], where the learning is based on the stochastic gradient descent (SGD). For each training example denoted by $\mathbf{x}_i$, the gradient of the cost $\mathscr{L}(i)$ with respect to the target word embedding $\mathbf{w}_i$ is computed as $\Delta\mathbf{w}_i = \frac{\partial\mathscr{L}(i)}{\partial\mathbf{w}_i}$. The learning is then formulated by:

$$\mathbf{w}_i = \mathbf{w}_i + \alpha\Delta\mathbf{w}_i \tag{1}$$

where $\alpha$ is the learning rate. Note that $\alpha$ is shared by all the dimensions, and there are no additional constraints over the dimensions. This leads to dimensional homogeneous embeddings $\{\mathbf{w}_i\}$, which means (1) No dimension is more preferable in learning; (2) No dimension is more informative; (3) Dimensions are exchangeable. We propose several algorithms to break the homogeneity and learn ordered embeddings where the significance of dimensions is in descending order.

### 3.1 Nested Dropout

This first ordered learning method is based on the nested dropout approach proposed by [7]. In this method, only the first $k$ dimensions are updated for each training sample, where $k$ is randomly sampled from a geometric distribution $P(k) = \frac{\beta^k}{Z}$ where $\beta$ is a constant to control the distribution decay, and $Z$ is the partition function to ensure that $P(k)$ is normalized. The learning can be formulated by:

$$k \sim \frac{\beta^k}{Z}$$

$$\mathbf{w}_i(d) = \begin{cases} \mathbf{w}_i(d) + \alpha\Delta\mathbf{w}_i(d) & d \leq k \\ \mathbf{w}_i(d) & d > k \end{cases} \tag{2}$$

where $\mathbf{w}_i(d)$ is the $d$-th dimension of the word embedding $\mathbf{w}_i$.

### 3.2 $\alpha$-decay Algorithm

Note that in the nested dropout, the probability that dimension $d$ is updated is $1-\sum_{k=1}^{d-1}\frac{\beta^k}{Z}$. It can be verified that this is another geometric distribution $P(d)=\frac{\beta^d}{Z'}+b$ where $Z'$ is the partition function and $b$ is a constant. Simply ignoring $b$, the expectation of Eq. (2) with respect to $P(k)$ is:

$$\mathbb{E}_{P(d)}[\mathbf{w}_i(d)] = \mathbf{w}_i(d) + \alpha\frac{\beta^d}{Z'}\Delta\mathbf{w}_i(d),$$

which indicates that the nested dropout can be simply implemented as a geometric decay on the learning rate over dimensions. This is formulated as follows:

$$\mathbf{w}_i(d) = \mathbf{w}_i(d) + \alpha\beta^{d-1}\Delta\mathbf{w}_i(d) \tag{3}$$

where $\beta/Z'$ has been absorbed in the learning rate $\alpha$.

This new learning rule uses the training data in a more efficient way, since all the dimensions are updated for every training sample.

### 3.3 $\lambda$-growth Algorithm

Another approach to learn ordered word representations is to place more aggressive $\ell$-1 penalties on higher dimensions. It is well known that the $\ell$-1 penalty leads to sparse codes, which means that most of the information in training data will be learned by less penalized dimensions, i.e., lower dimensions. In this study, the penalty on dimension $d$ is set to be $\lambda_d|\mathbf{w}_i(d)|$ where $\lambda_d=\gamma\beta^{D-d}$.

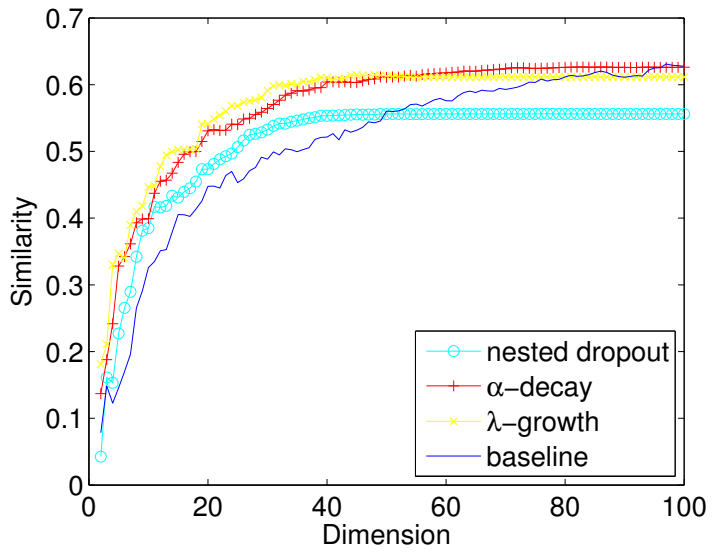



Figure 1: Results with different ordered learning approaches, where the iteration is 100.

### 3.4 Empirical Results

The first experiment compares the conventional non-ordered word representations (baseline) and the ordered representations learned by the three methods described above. The

text8 data set from Matthew Mahoney[1] is used to train the word embeddings. The word2vec tool (plus our modifications) is used to conduct the training.[2] The result is evaluated on the similarity task and the WordSimilarity-353 dataset [8] is used as the test data[3]. In all the experiments, the basic learning rate $\alpha$=0.05. In nested dropout, the parameter $\beta$ is set to be $0.9$. In $\alpha$-decay, the parameter $\beta$ is set to be $0.95$. In $\lambda$-growth, $\beta$ is set to be $0.9$, and the sparsity factor $\gamma$ is $0.01$.

To examine if the dimensions of the learned representations are ordered, 100-dimensional embeddings are first trained, and then the first $k$ dimensions are picked out to conduct the similarity test. Fig. 1 presents the results with different training approaches, where the number of iterations is set to $100$ to ensure the learning has been converged for all methods. It can be observed that with all the three ordered learning methods, better performance than the baseline is obtained if the selected number of dimensions is small ($k \leq 50$). Particularly with the first several dimensions ($k \leq 10$), the ordered learning shows significant superiority. This indicates that the ordered learning methods indeed learned orders over dimensions. Comparing the three ordered learning methods, $\alpha$-decay and $\lambda$-growth are clearly superior to nested dropout, which confirms the advantage of our proposal.

In order to further investigate the convergence property of the proposed methods, performance of the ordered learning methods with different iterations are compared. Fig. 2 presents the results, where only $\alpha$-decay and $\lambda$-growth are presented due to their advantage shown in the previous experiment. From these results, it can be seen that for both $\alpha$-decay and $\lambda$-growth, if the number of iterations is small, then only low dimensions are learned. When the number of iterations increases, more and more dimensions are learned. With a large number of iterations ($100$ for example), the two learning approaches almost converge, and the asymptotic performance with the full set of dimensions is almost the same as the conventional representations. When comparing the two approaches, it seems that $\lambda$-growth converges faster, possibly due to the non-decayed learning rate it uses. However, $\alpha$-decay seems to converge to a better asymptotic performance.

## 4 Sweeping Methods

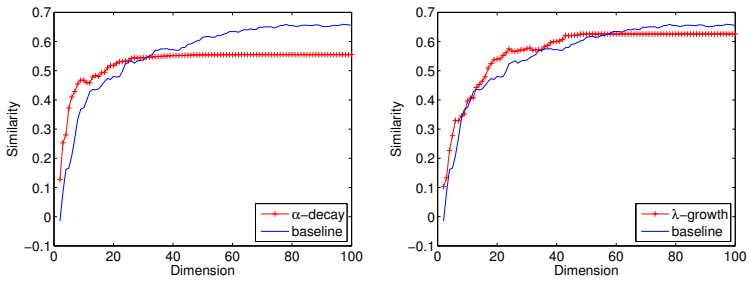
As mentioned, a potential problem of ordered learning is the low convergence rate of high dimensions. [7] introduced the unit sweeping approach which fixes converged dimensions and boosts learning for the subsequent dimensions. In practice, however, we find this approach is quite inefficient. This paper presents two new approaches to perform sweeping. For simplicity, only the $\alpha$-decay algorithm is considered.

### 4.1 Linear Sweeping

In the linear sweeping method, we assume that the number of converged dimensions is linear to the training process. Let $N$ denote the total number of data to process, and $n$ denote the number of data that have been processed. Define $r = \frac{n}{N}$ as the training progress ratio, the number of converged dimensions is then assumed to be $M = \lfloor rD \rfloor$, where $D$ is the total number of dimensions. The learning rule is given as follows.

$$\mathbf{w}_i(d) = \begin{cases} \mathbf{w}_i(d) + \alpha\beta^{d-M-1}\Delta\mathbf{w}_i(d) & d \geq M \\ \mathbf{w}_i(d) & d < M \end{cases}$$

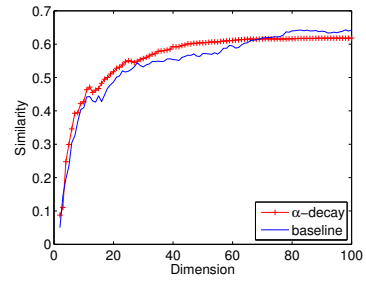
[1]http://mattmahoney.net/dc/text8.zip
[2]https://code.google.com/p/word2vec
[3]http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/wordsim353.html



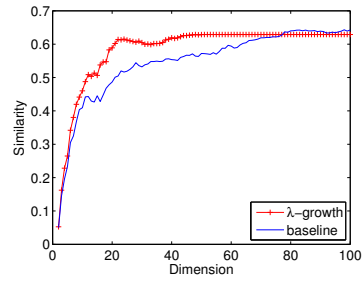

(a) $\alpha$-decay, iteration=10



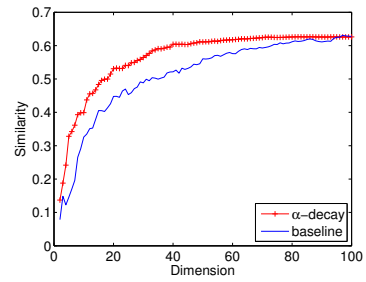

(b) $\lambda$-growth, iteration=10



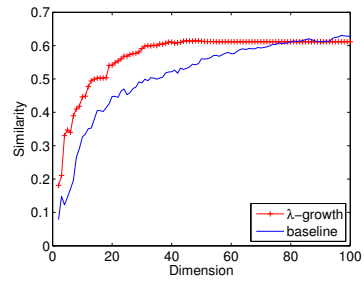

(c) $\alpha$-decay, iteration=50




(d) $\lambda$-growth, iteration=50




(e) $\alpha$-decay, iteration=100




(f) $\lambda$-growth, iteration=100

Figure 2: Results with word representations learned by $\alpha$-decay and $\lambda$-growth, with 10, 50, 100 iterations.

## 4.2 $\chi^2$ Sweeping

The sweeping approach can be regarded as a deformation of the learning rate function caused by the training progress. For the linear sweeping, the deformed learning rate function is given by:

$$\alpha(d;r) = \begin{cases} \alpha\beta^{d-M-1} & d \geq M \\ 0 & d < M \end{cases} \tag{4}$$

where $M = \lfloor rD \rfloor$. Note that this function is not continuous, corresponds to the fact that low dimensions cannot be improved once the sweeping passes.

We propose a 'soft' sweeping based on the $\chi^2$ distribution as follows:

$$\alpha(d;r) = \frac{\alpha}{\chi^2(1,3)}\chi^2(d, M+3), \tag{5}$$

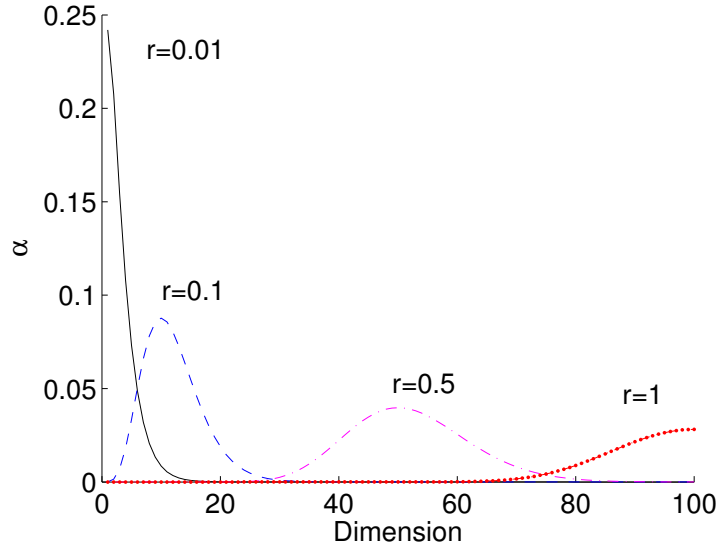



Figure 3: The learning rate functions at different progress ratios with the $\chi^2$ sweeping.

where the $\chi^2$ distribution is defined by:

$$\chi^2(x;k) = \begin{cases} \frac{x^{(k/2-1)}e^{-x/2}}{2^{k/2}\Gamma\left(\frac{k}{2}\right)}, & x \geq 0; \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $k$ is a positive integer.

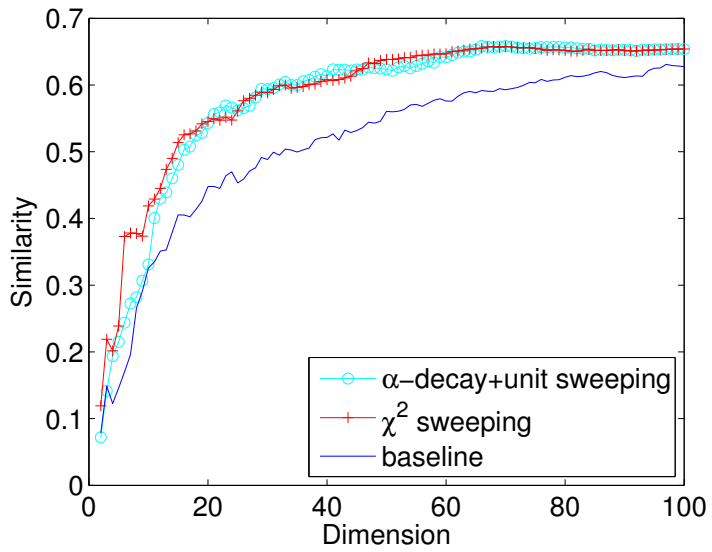



Figure 4: Results with linear sweeping and $\chi^2$ sweeping

Since the mode of $\chi^2$ is $\max\{k-2, 0\}$, the form $\chi^2(d, M+3)$ in Eq. (5) ensures that the $(M+1)$-th dimension is the mostly trained when the first $M$ dimensions have converged. The factor $\chi^2(1; 3)$ is used to scale the learning rate of the first dimension to $\alpha$. Fig. 3 shows the deformed learning rate functions at different progress ratios of the training. It is clear to see that the mode of the learning rate function moves slowly from low dimensions to high dimensions. By this method, all the dimensions have chance to be updated at any time of the training. This is a big advantage compared to the linear sweeping. Additionally, the variance of the learning rates becomes larger with the training progresses, which means that the learning rates become smaller with the training towards the end, which generally helps to find a good local minimum.

### 4.3 Empirical Results

The performance with the linear and $\chi^2$ sweeping is presented in Fig. 4. It can be seen that the two sweeping methods offer performance improvement. Particularly on low dimensions, the $\chi^2$ sweeping significantly outperforms the linear sweeping. Interestingly, with the sweeping methods, the ordered representations perform better than the conventional non-ordered representations. We argue that this is attributed to the incremental nature of the ordered learning.

## 5 Conclusion

This paper presented several algorithms to learn ordered word representations, by placing various heterogeneous constrains on dimensions of the embedding space, such as dropout probabilities, learning rates and sparse penalties. The experiments on the WordSimilarity task demonstrated that the proposed methods can learn ordered representations, and with the proposed sweeping methods, ordered representations can even outperform conventional non-ordered representations. Future work involves investigating performance of ordered representations on other tasks and developing the theory of partial learning.

**References**


1. GE Hinton, JL McClelland, and DE Rumelhart, "Distributed representations," in *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1*. MIT Press, 1986, pp. 77–109.
2. Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011, pp. 142–150.
3. Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng, "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*. Association for Computational Linguistics, 2012, pp. 873–882.
4. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
5. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, Eds., pp. 3111–3119. Curran Associates, Inc., 2013.
6. Jeffrey Pennington, Richard Socher, and Christopher D Manning, "Glove: Global vectors for word representation," *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, vol. 12, 2014.
7. Oren Rippel, Michael A Gelbart, and Ryan P Adams, "Learning ordered representations with nested dropout," *arXiv preprint arXiv:1402.0915*, 2014.
8. Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin, "Placing search in context: The concept revisited," in *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001, pp. 406–414.