Parameterized context windows in Random Indexing

Tobias Norlund
Schibsted Products and Technology
Västra Järnvgsgatan 21
111 64 Stockholm
Sweden
tobias.norlund@schibsted.com

David Nilsson
Nepa
Maria Skolgata 83
118 53 Stockholm
Sweden
david.nilsson@nepa.com

Magnus Sahlgren
Gavagai
Slussplan 9
111 30 Stockholm
Sweden
magnus.sahlgren@gavagai.se

Abstract

This paper introduces a parameterization for word embeddings produced by the Random Indexing framework. The parameterization introduces position specific weights in the context windows, and the approach is shown to improve the performance in both word similarity and sentiment classification tasks. We also demonstrate the relation between Random Indexing and Convolutional Neural Networks.

1 Introduction

Quantifying the importance of contextual information for semantic representation is the goal of *distributional semantics*, in which contextual information is used to quantify semantic similarities between words (Turney and Pantel, 2010). However, standard practice in distributional semantics is to weight the importance of context items based on either its frequency (Sahlgren et al., 2016), its distance to the focus word (Lund et al., 1995), or its global co-occurrence statistics (Niwa and Nitta, 1994). Thus far, there has not been much work on applying machine learning to this in order to select useful context items for distributional semantics.

The idea with the proposed parameterization is to weight the items in the context window based on their usefulness for accomplishing some specific task, such as sentiment classification or word similarity rating. In this paper, we introduce a simple parameterization for the Random Indexing processing model. We first show that Random Indexing can be formulated in terms of a convolution, in order to situate the framework in the context of neural networks. We then introduce a simple parameterization of the positions on the context windows, and we show that it improves the performance of the embeddings in some word similarity and sentiment classification tasks. [1]

2 Notation

Using a vocabulary $\mathcal{V}$ of words w_i for $i = 1, \ldots, |\mathcal{V}|$, we seek word embeddings $\mathbf{v}_i \in \mathbb{R}^d$ by collecting statistics from a corpus $\mathcal{C} = \{w_1, \ldots, w_t, \ldots, w_N\}$. We will interchangeably mix subscripts i and t of words and embeddings to index the vocabulary and corpus respectively.

3 Random Indexing as Convolution

Random Indexing (RI) (Kanerva et al., 2000; Kanerva, 2009) is a distributional semantic model that updates the embedding vectors $\mathbf{v}_*$ in an online fashion by summing the sparse random vectors $\mathbf{e}_*$ that represent the context items (these vectors are called *random index vectors* and act as unique identifiers for the context items, words in this case):

$$\mathbf{v}_t \leftarrow \mathbf{v}_t + \sum_{\substack{l=-k \\ l\neq 0}}^{k} h(w_{t+l})\mathbf{e}_{t+l} \qquad (1)$$

k is the context window size and $h(w)$ some weight that quantifies the importance of the context item (the standard setting is $h(w) = 1$ for all w). Here $\mathbf{e}_{t+l}$ is the random index vector to corpus item w_{t+l}.

Equation (1) describes the update rule of RI and the final embeddings $\mathbf{v}_i$ can be expressed as:

$$\mathbf{v}_i = \sum_{\substack{l=-k \\ l\neq 0}}^{k} \sum_{\substack{t=1 \\ w_t=w_i}}^{N} h(w_{t+l})\mathbf{e}_{t+l} \qquad (2)$$

To establish the equivalence between RI and convolution, we can reformulate the update rule

[1] The code is available at: http://github.com/TobiasNorlund/Attention_RI

in Equation (1) as follows; let $\mathbf{h} \in \mathbb{R}^{(2k+1)\times d}$ be a filter function where

$$h_{lj} = \begin{cases} 1, & \forall (l,j) \in \{(0,\frac{d}{2}),\ldots,(k-1,\frac{d}{2}),(k+1,\frac{d}{2}),\ldots,(2k,\frac{d}{2})\} \\ 0, & \text{else} \end{cases}$$

Furthermore, let $\mathbf{S} \in \mathbb{R}^{N\times d}$ be a matrix of stacked sparse random vectors $\mathbf{e}_t$. Now, if we use $\mathbf{h}$ and $\mathbf{S}$, we can rewrite the second term of the random indexing update rule in Equation (1):

$$\mathbf{Z}_{tv} = \sum_{l=0}^{2k}\sum_{j=0}^{d}(\mathbf{e}_{t-l+k})_{v-j+d}\mathbf{h}_{lj}. \tag{3}$$

Equation (3) is a 2D discrete convolution between $\mathbf{S}$ and $\mathbf{h}$, hence:

$$\mathbf{Z}_{tv} = (\mathbf{S} * \mathbf{h})[t,v] = \textstyle\sum_{l=0}^{2k}\sum_{j=0}^{d}(\mathbf{e}_{t-l+k})_{v-j+d}\mathbf{h}_{lj}. \tag{4}$$

Because $\mathbf{h}$ has been defined with zeros everywhere except for column $\frac{d}{2}$, Equation (4) can been seen as a 1D convolution over each column vector in $\mathbf{S}$,

$$\mathbf{Z}_{tv} = (\mathbf{S}_v^T * \mathbf{h}_{\frac{d}{2}}^T)[t] = \sum_{l=0}^{2k}(\mathbf{e}_{t-l+k})_v\mathbf{h}_{t\frac{d}{2}}. \tag{5}$$

4 Dealing with Redundant Features

Since word embeddings (produced by RI or some other distributional model) are constructed unsupervised by collecting co-occurrence information from a large corpus, it is likely that the resulting embeddings are very general, which may lower the expressiveness of the embeddings if they are going to be used in a very specific domain. Take the example of training a text categorization classifier within a financial context; corpus occurrences of the words "bank" and "stock" in the senses of LARGE COLLECTION and INVENTORY will likely not provide useful information for the embeddings in this domain. In word embeddings, different senses are represented by co-occurrences with different context items (Cuba Gyllensten and Sahlgren, 2015). We refer to context items that are less useful for a specific task as *redundant features* of the embeddings.

Unfortunately, it is not, in the general case, possible to know a priori which context items will be useful to construct embeddings for a particular task. Such context (i.e. feature) selection instead needs to be performed jointly with training the classifier. When backpropagation is used as optimization strategy of the classifier, one can also treat the word embeddings as parameters to update. It is straightforward to take the derivatives of the objective function with respect to the input and apply Stochastic Gradient Descent (SGD) updates just as for the model parameters. This strategy is well known (Zhang and Wallace, 2015), and will be referred to as SGD Random Indexing (SGD-RI).

5 Parameterization of context window

Another strategy is to parameterize the word embeddings, and to optimize those parameters jointly with the task using backpropagation. The RI algorithm, as defined in (Sahlgren et al., 2016), weights the importance of context items based on their relative frequency according to Equation (6):

$$h(w_t) = \exp\left(-c \cdot \frac{f(w_t)}{|\mathcal{V}|}\right) \tag{6}$$

where c is a constant, $f(w_t)$ is the corpus frequency of item w_t, and $|\mathcal{V}|$ is the size of the vocabulary (i.e. the number of unique words seen thus far).

We would however like to parameterize context items not only depending on relative frequency but also on their usefulness for the specific task at hand. To describe the suggested parametrization, recall the Random Indexing algorithm in Equation (1) where we look at each word and its context in the corpus in a streaming fashion, and construct embedding vectors by summing the index vectors of all words occurring in the context. A fairly obvious refinement of this algorithm would be to parameterize the relative positions within the context window depending on their usefulness for the task at hand. Equation (7) formalizes the parameterization by introducing an additional factor to the weighting scheme:

$$h(w_{t+l}) = \theta_l^{w_t}\exp\left(-c \cdot \frac{f(w_t)}{|\mathcal{V}|}\right) \tag{7}$$

Inserting this parameterization into the update rule in Equation (1), we get:

$$\mathbf{v}_t \leftarrow \mathbf{v}_t + \sum_{\substack{l=-k \\ l\neq 0}}^{k}\theta_l^{w_t}\exp\left(-c \cdot \frac{f(w_{t+l})}{|\mathcal{V}|}\right)\mathbf{e}_{t+l} \tag{8}$$

which is equivalent to:

$$\mathbf{v}_i = \sum_{\substack{l=-k \\ l\neq 0}}^{k} \sum_{\substack{t=1 \\ w_t=w_i}}^{N} \theta_l^{w_t} \exp\left(-c \cdot \frac{f(w_{t+l})}{|\mathcal{V}|}\right) \mathbf{e}_{t+l} \tag{9}$$

By careful inspection, the $\theta_l^{w_t}$ can be moved outside the inner sum, while swapping the subscript to i since $w_t = w_i$:

$$\mathbf{v}_i = \sum_{\substack{l=-k \\ l\neq 0}}^{k} \theta_l^{w_i} \underbrace{\sum_{\substack{t=1 \\ w_t=w_i}}^{N} \exp\left(-c \cdot \frac{f(w_{t+l})}{|\mathcal{V}|}\right) \mathbf{e}_{t+l}}_{\tilde{\mathbf{v}}_i^l} \tag{10}$$

The rewrite now allows the inner sum to be calculated before fitting the $\theta_l^{w_i}$s which makes the algorithm much more efficient. In practice, this means we aggregate an embedding vector $\tilde{\mathbf{v}}_i^l$ *for each relative window position* l, for each word w_i. Stacking these $2k$ context vectors into a matrix $\mathbf{V}_i$ and collecting the $\theta_l^{w_i}$s in a vector yields:

$$\mathbf{V}_i = \begin{bmatrix} \tilde{\mathbf{v}}_i^{-k} & \dots & \tilde{\mathbf{v}}_i^{-1} & \tilde{\mathbf{v}}_i^{+1} & \dots & \tilde{\mathbf{v}}_i^{+k} \end{bmatrix} \tag{11}$$

$$\boldsymbol{\theta}_i = \begin{bmatrix} \theta_{-k}^{w_i} \\ \vdots \\ \theta_{-1}^{w_i} \\ \theta_{+1}^{w_i} \\ \vdots \\ \theta_{+k}^{w_i} \end{bmatrix} \tag{12}$$

Equation (10) can now be rewritten as a matrix vector multiplication:

$$\mathbf{v}_i = \mathbf{V}_i \boldsymbol{\theta}_i. \tag{13}$$

In other words, this suggests instead of aggregating embedding vectors $\mathbf{v}_i$ according to (9), to aggregate matrices $\mathbf{V}_i$ upon parsing the corpus. The embedding vectors are then calculated as a multiplication with a parameter vector $\boldsymbol{\theta}_i$ according to (13). Note that when $\boldsymbol{\theta}_i = \mathbf{1}$ you recover the vanilla Random Indexing embeddings.

We will refer to this strategy as Parameterized Random Indexing (PAR-RI).

6 Example: Word Similarity

To exemplify the effectiveness of the proposed parameterization, we use the SimLex-999 (Hill et al., 2015) test in order to see how much the Spearman rank correlation can be improved by fitting the $\boldsymbol{\theta}_i$s such that cosine similarity between the embedding vectors correspond to the similarity ratings. Formally, we seek to minimize the following objective function:

$$\min_{\boldsymbol{\theta}_*} \sum_{(w_i,w_j)\in\mathcal{S}} \underbrace{\frac{1}{2}\left(\cos\alpha_{ij} - s(w_i,w_j)\right)^2}_{f(w_i,w_j)} \tag{14}$$

where $(w_i, w_j) \in \mathcal{S}$ corresponds to each word pair in SimLex. $s(w_i, w_j)$ is the SimLex similarity score for the word pair (scaled to $[0, 1]$) and $\cos\alpha_{ij}$ is the cosine similarity between the word's corresponding vectors:

$$\alpha_{ij} = \frac{\mathbf{v}_i^T \mathbf{v}_j}{\|\mathbf{v}_i\|_2 \|\mathbf{v}_j\|_2} \tag{15}$$

where $\mathbf{v}_i$ and $\mathbf{v}_j$ are w_i and w_j's corresponding word vectors, calculated as in equation (13). Since this is a non-convex problem, SGD is applied as optimization strategy. Calculating the gradient of f with respect to $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$ is straightforward:

$$\frac{\delta f}{\delta \boldsymbol{\theta}_i} = \left(\cos\alpha_{ij} - s(w_i,w_j)\right) \frac{\delta \cos\alpha_{ij}}{\delta \boldsymbol{\theta}_i} \tag{16}$$

$$\frac{\delta f}{\delta \boldsymbol{\theta}_j} = \left(\cos\alpha_{ij} - s(w_i,w_j)\right) \frac{\delta \cos\alpha_{ij}}{\delta \boldsymbol{\theta}_j}. \tag{17}$$

Applying the chain rule, the gradient of $\cos\alpha_{ij}$ becomes:

$$\begin{aligned} \frac{\delta \cos\alpha_{ij}}{\delta \boldsymbol{\theta}_i} &= \frac{\delta \cos\alpha_{ij}}{\delta \mathbf{v}_i} \frac{\delta \mathbf{v}_i}{\delta \boldsymbol{\theta}_i} \\ \frac{\delta \cos\alpha_{ij}}{\delta \mathbf{v}_i} &= \frac{\mathbf{v}_j \|\mathbf{v}_i\|_2 - \mathbf{v}_i \frac{\mathbf{v}_i^T \mathbf{v}_j}{\|\mathbf{v}_i\|_2}}{\|\mathbf{v}_i\|_2^2 \|\mathbf{v}_j\|_2} \\ \frac{\delta \mathbf{v}_i}{\delta \boldsymbol{\theta}_i} &= \mathbf{V}_i. \end{aligned} \tag{18}$$

The expression for $\frac{\delta \cos\alpha_{ij}}{\delta \boldsymbol{\theta}_j}$ is the same, but with the subscripts interchanged. We now apply SGD to optimize the $\boldsymbol{\theta}_i$s iteratively using the following update rules:

$$\begin{aligned} \boldsymbol{\theta}_i^{(t+1)} &= \boldsymbol{\theta}_i^t - \eta \frac{\delta f}{\delta \boldsymbol{\theta}_i} \\ \boldsymbol{\theta}_j^{(t+1)} &= \boldsymbol{\theta}_j^t - \eta \frac{\delta f}{\delta \boldsymbol{\theta}_j}. \end{aligned} \tag{19}$$

This procedure is performed using $\mathbf{V}_*$ matrices generated from a dump of Wikipedia with the Random Indexing hyper-parameters listed in Table 1. The $\boldsymbol{\theta}_i$s are initialized to one-vectors ($\boldsymbol{\theta}_* = \mathbf{1}$) and updated according to equation (19) with a learning rate $\eta = 1.0$ until convergence.

Table 1: Hyper-parameters for PAR-RI.

Parameter	Value	Description
d	2,000	Dimensionality
k	10	Window size
c	60	Constant in frequency weight
ϵ	10	Non-zero elements in index vectors randomly drawn from $\{-1, +1\}$

The results are summarized in Table 2. We can see that the Spearman correlation is drastically improved with the optimized $\boldsymbol{\theta}_i$s. This experiment can be seen as, for each word w_i, finding a linear combination in the column space of $\mathbf{V}_i$ that optimizes the cosine similarity of the word vectors to match the SimLex similarity scores. It is remarkable that optimizing the $\boldsymbol{\theta}_i$s in the relatively small 20-dimensional ($\mathbf{R}^{2k}$) subspaces of the full word space ($\mathbf{R}^{2000}$) yields such a big improvement.

Table 2: Results of the SimLex experiment.

	Avg. error	Spearman
Initial $\boldsymbol{\theta}_i$s ($\boldsymbol{\theta}_* = \mathbf{1}$)	0.28	0.21
Optimized $\boldsymbol{\theta}_i$s	0.19	**0.62**

7 Example: Sentiment Classification

The improvements reported in the previous section should motivate the parameterization to be viable for improving the performance in text classification as well. In this section, we parameterize the embeddings for sentiment classification using two standard benchmarks; the Pang and Lee Sentence Polarity Dataset v1.0 (PL05) (Pang and Lee, 2005) and the Stanford Sentiment Treebank (SST) (Socher et al., 2013). The PL05 data consists of 10,662 short movie reviews that are classified as either positive or negative. Experiments using this dataset are split into 25% test and 75% train/validation sets and evaluated by 5-fold cross validation on the training/validation set. We make two consecutive runs, in total 10 trainings, and report their maximum, minimum and mean accuracy as well as their standard deviation. The SST data is an extension of PL05 with train/validation/test splits provided. The dataset also provides fine-grained labels (very positive, positive, neutral, negative, very negative). In this study we have however omitted the neutral labels and treated it as a binary classification problem by merging the very positive, positive, very negative and negative classes into two. We report the maximum, minimum and mean accuracy as well as the standard deviation of 10 consecutive runs using the provided train/val/test splits.

We use two different classifiers in these experiments. The first is a standard neural network (referred to as MLP for Multi-Layer Perceptron) (Rumelhart et al., 1986) with one hidden layer of 120 nodes with sigmoid activations and one sigmoid output unit. All word vectors are normalized to an l_2 norm of 1 and naively summed to produce document vectors. The weights in the neural network are also l_2 regularized with a constant factor of $\lambda = 0.001$. The second classifier is the model proposed by Kim (2014) which implements a Convolutional Neural Network (CNN). The hyper-parameters used are listed in table 3. Like the MLP model, the word embeddings are also normalized to unit length.

Table 3: Hyper-parameters for CNN.

Parameter	Value	Description
n	300	Number of filters, 100 of height 3,4,5 respectively
p	0.5	Dropout rate
s	3	Filter max l_2-norm

As comparison with the different RI-based embeddings (RI, SGD-RI, and PAR-RI), we also include results using embeddings produced with SGNS (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), both with 300-dimensional vectors and a window size of 2. We also include results using embeddings randomly sampled from a uniform $U(-0.25, 0.25)$ distribution (RAND). All word embeddings (except for the RAND vectors) are pre-trained (unsupervised) on a dump of Wikipedia. We list all hyper-parameters for RI, SGNS and GloVe in table 4, 5 and 6

Table 4: Hyper-parameters for RI

Parameter	Value	Description
d	2,000	Dimensionality
k	2	Window size
c	60	Constant
ϵ	10	# non-zero elements in index vectors randomly drawn from $\{-1, +1\}$

The results of all experiments are shown in ta-

Table 5: Hyper-parameters for SGNS

Parameter	Value	Description
d	300	Dimensionality
k	2	Window size
negative	5	Number of negative samples per positive
down sampling	no	No down sampling is applied
α	0.025	Initial learning rate
iter	5	Number of iterations

Table 6: Hyper-parameters for GloVe

Parameter	Value	Description
d	300	Dimensionality
k	2	Window size
iter	15	Number of training iterations
x-max	10	Cutoff in weighting function
α	0.75	Constant in exponent of weighting function
η	0.05	Initial learning rate

ble 7 (next page).[2] Comparing the various embeddings, it is obvious that the performance differences are very small, and thus not likely to be of any significant practical importance. This is especially true for the MLP experiments where the variances reach over two points in many cases. On the other hand, all embeddings outperform the randomized RAND vectors, which demonstrates that classification performance is improved when the model can take semantic information into account. The best performing embedding for the MLP classifier is PAR-RI, while SGNS performs better using the CNN model. The GloVe embeddings, despite their theoretical similarities to the SGNS embeddings (Suzuki and Nagata, 2015), consistently underperform both in comparison with SGNS and PAR-RI (and, in the case of the MLP classifier, also the SGD-RI embeddings). This is in contrast to the experiments performed by (Zhang and Wallace, 2015) where the difference was minor.

Comparing the PAR-RI embeddings with SGD-RI and standard RI, it seems PAR-RI performs well, with the highest mean accuracy on the SST dataset, using the MLP model. SGD-RI improves the results compared to the standard RI embeddings for the MLP model, but not for the CNN model. Updating of the SGNS embeddings just like SGD-RI for the CNN have also been studied in Zhang and Wallace (2015), who report a performance boost of about $\sim$0.8%. This also contrasts to our results with SGD-RI using the CNN model, which instead decrease the performance compared to standard RI. This could be due to the RI embeddings being more high dimensional than SGNS, yielding a larger and harder parameter space to optimize.

Comparing our results to other reported results in the literature, Kim (2014) and Zhang and Wallace (2015) manage to push the boundaries up to 80.10 for the PL05 data, and up to 84.88 for the SST data using SGNS embeddings pre-trained on a much larger 100 billion tokens Google News dataset. We believe this somewhat increased performance is partly due to the bigger dataset. Another factor could also be that the language style in news articles is more similar to the movie reviews compared to Wikipedia, arguably yielding better-suited embeddings.

8 Optimized Context Profiles

When the PAR-RI parametrization was proposed, the hypothesis was that certain relative positions in the context windows would be more important in describing the context of a word than others. The results in the two previous sections demonstrate that the proposed parameterization is able to improve the embeddings in both a word similarity task, and (to a lesser extent) a sentiment classification task. This indicates that the parameterization is actually able to find useful context profiles for terms used in the various test settings. In this section, we exemplify the kinds of context profiles learned when trained for the sentiment classification task.

Figure 1 (on page 7) shows the learned weights per context window position for four different adjectives (top row), four different determiners (middle row), and four different nouns (bottom row). The parameterization obviously has a larger effect for some words than for others; as an example, the windows for "good" and "bad" is much more parameterized than the windows for "reliable" and "positive", and the windows for the determiners are in general much more parameterized than the windows for nouns. It is interesting to note that there is a small tendency that the windows for the adjectives have a higher weight in the $+1$ position, which is consistent with a linguistic analysis of adjectives as qualifiers of succeeding nouns. By contrast, the window positions for the determiners seem to have a higher weight in the positions just preceding the focus word, while the windows for

[2] Since our focus in this paper is the effect of the word embeddings, we will not comment further on the performance differences between the MLP and CNN classifiers.

Table 7: Experiment results. Mean accuracies in %. The parentheses contain the maximum achieved accuracy, the standard deviation and the minimum achieved accuracy of the consecutive runs

Emb + Model	PL05	SST
RAND MLP	68.23 (↑ 69.58, ±0.98, ↓ 66.32)	69.89 (↑ 71.11, ±0.83, ↓ 68.64)
RI MLP	72.45 (↑ **74.91**, ±2.99, ↓ 66.54)	75.13 (↑ 77.98, ±2.54, ↓ 73.75)
SGD-RI MLP	73.62 (↑ 74.16, ±0.77, ↓ 71.42)	77.91 (↑ 78.80, ±0.82, ↓ 76.11)
PAR-RI MLP	72.45 (↑ 74.83, ±2.20, ↓ 68.90)	**78.03** (↑ **79.63**, ±1.60, ↓ 74.46)
SGNS MLP	**73.84** (↑ 74.76, ±1.14, ↓ 71.57)	77.27 (↑ 79.57, ±3.13, ↓ 70.02)
GLOVE MLP	71.29 (↑ 73.67, ±1.67, ↓ 68.60)	76.26 (↑ 77.32, ±1.66, ↓ 71.61)
RAND CNN	72.12 (↑ 72.91, ±0.50, ↓ 71.28)	76.99 (↑ 77.94, ±0.76, ↓ 75.50)
RI CNN	76.18 (↑ 76.60, ±0.35, ↓ 75.51)	81.83 (↑ 82.72, ±0.39, ↓ 81.39)
SGD-RI CNN	75.67 (↑ 76.26, ±0.63, ↓ 74.22)	81.31 (↑ 81.89, ±0.38, ↓ 80.78)
PAR-RI CNN	77.55 (↑ 78.08, ±0.31, ↓ 77.09)	81.77 (↑ 82.64, ±0.60, ↓ 80.66)
SGNS CNN	**77.92** (↑ **78.34**, ±0.24, ↓ 77.55)	**83.44** (↑ **84.00**, ±0.51, ↓ 82.00)
GLOVE CNN	77.35 (↑ 77.77, ±0.34, ↓ 76.79)	81.56 (↑ 82.06, ±0.34, ↓ 80.78)

nouns seem to have very small parameterization. It thus seems as if the parameterization is able to learn slightly different window profiles for different parts of speech.

As noted in the introduction, is is common practice in distributional semantics to weight the context windows by the distance to the focus word. If this is an optimal strategy, we should see a bell-like curve leaning to zero at the edges. Such a shape is partially present for some of the words, for example in "and", "of", "good" and "bad", but for most words, the weights are almost unchanged. We believe this could be due to the vanishing gradient problem where the gradient seems to vanish deeper down the model. In addition, the less common the word is in the training set, the less it is updated. Another interesting aspect of the learned weights is that by inspecting the l_1 norm of the weight vectors, we get a hint of the words' relative importance for the given task. We can see that the l_1 norm for the words "good" and "bad" are larger than for "the" and "of", which feels natural for the sentiment classification task.

9 Conclusion

This paper has introduced a simple parameterization for the RI framework, which has also been derived in terms of convolution. It parameterizes the positions in the context windows and optimizes with respect to the performance of the embeddings in some given task, such as word similarity or text classification. Our experiments show that the proposed PAR-RI model is able to improve the performance of the embeddings in many cases, and that the results are competitive in comparison with other well-known embeddings. The idea of parameterizing the window positions could also be applied to other distributional semantic models, such as SGNS.

We note that all embeddings used in the sentiment classification task produce very similar results. This indicates that in practice, the word embeddings included in this paper are more or less equivalent. It is therefore doubtful whether it is possible to draw any conclusions based on these results regarding the question whether any single embedding is superior to the others in the general case.

The examples of context profiles provided as examples of the parameterization shows some interesting effects. However, training the position-dependent weights is non-trivial, and one could probably think of better initializations of the weights than just one-vectors, for example using a bell-like shape. The vanishing gradient problem would however remain, and the weights for uncommon words will not change significantly.

The conclusion of the experiments using SGD-RI is that updating the embeddings jointly with the classification model using SGD does not necessarily improve generalization. This is in fact not so strange. Moving around only a subset of the words (i.e. the words present in the training set), while leaving the rest untouched produces an inconsistent space with undefined distributional properties between updated and non-updated embeddings. It could therefore be an idea to use randomized embeddings for all words not present in the training

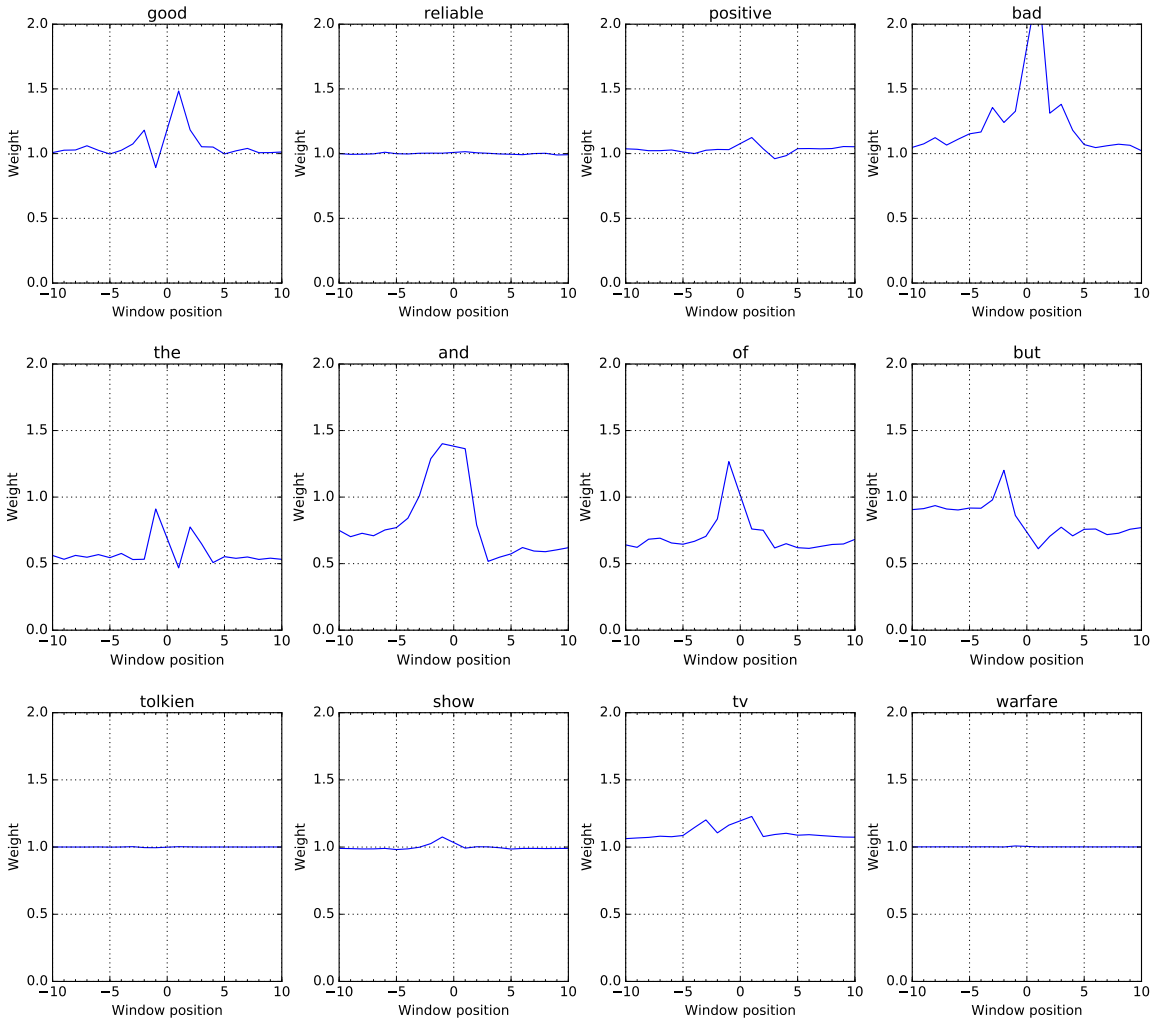

Figure 1: The learned weights for four adjectives (top row), four determiners (middle row), and four nouns (bottom row).

set because they then can be regarded as approximately orthogonal, and thus should not interfere with the semantic structure.

References

Amaru Cuba Gyllensten and Magnus Sahlgren. 2015. Navigating the semantic horizon using relative neighborhood graphs. In *Proceedings of EMNLP*, pages 2451–2460.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Pentti Kanerva, Jan Kristofersson, and Anders Holst. 2000. Random Indexing of text samples for Latent Semantic Analysis. In *Proceedings of CogSci*, page 1036.

Pentti Kanerva. 2009. Hyperdimensional computing. *Cognitive Computation*, 1(2):139–159.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October. Association for Computational Linguistics.

Kevin Lund, Curt Burgess, and Ruth A. Atchley. 1995. Semantic and associative priming in high-dimensional semantic space. In *Proceedings of the 17th Annual Conference of the Cognitive Science Society*, pages 660–665. Hillsdale, NJ: Erlbaum.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Yoshiki Niwa and Yoshihiko Nitta. 1994. Co-occurrence vectors from corpora vs. distance vectors from dictionaries. In *Proceedings of the 15th Conference on Computational Linguistics - Volume*

1, COLING '94, pages 304–309, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Learning Internal Representations by Error Propagation, pages 318–362. MIT Press, Cambridge, MA, USA.

Magnus Sahlgren, Amaru Cuba Gyllensten, Fredrik Espinoza, Ola Hamfors, Anders Holst, Jussi Karlgren, Fredrik Olsson, Per Persson, and Akshay Viswanathan. 2016. The Gavagai Living Lexicon. In *Proceedings of LREC*.

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer.

Jun Suzuki and Masaaki Nagata. 2015. A unified learning framework of skip-grams and global vectors. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 186–191, Beijing, China, July. Association for Computational Linguistics.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, January.

Ye Zhang and Byron Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *CoRR*, abs/1510.03820.