Latent Vector Weighting for Word Meaning in Context

Tim Van de Cruys
RCEAL
University of Cambridge
tv234@cam.ac.uk

Thierry Poibeau
LaTTiCe, UMR8094
CNRS & ENS
thierry.poibeau@ens.fr

Anna Korhonen
Computer Laboratory & RCEAL
University of Cambridge
alk23@cam.ac.uk

Abstract

This paper presents a novel method for the computation of word meaning in context. We make use of a factorization model in which words, together with their window-based context words and their dependency relations, are linked to latent dimensions. The factorization model allows us to determine which dimensions are important for a particular context, and adapt the dependency-based feature vector of the word accordingly. The evaluation on a lexical substitution task – carried out for both English and French – indicates that our approach is able to reach better results than state-of-the-art methods in lexical substitution, while at the same time providing more accurate meaning representations.

1 Introduction

According to the distributional hypothesis of meaning (Harris, 1954), words that occur in similar contexts tend to be semantically similar. In the spirit of this by now well-known adage, numerous algorithms have sprouted up that try to capture the semantics of words by looking at their distribution in texts, and comparing those distributions in a vector space model.

Up till now, the majority of computational approaches to semantic similarity represent the meaning of a word as the aggregate of the word's contexts, and hence do not differentiate between the different senses of a word. The meaning of a word, however, is largely dependent on the particular context in which it appears. Take for example the word *work* in sentences (1) and (2).

(1) The painter's recent work is a classic example of art brut.

(2) Equal pay for equal work!

The meaning of *work* is quite different in both sentences. In sentence (1), *work* refers to the product of a creative act, viz. a painting. In sentence (2), it refers to labour carried out as a source of income. The NLP community's standard answer to the ambiguity problem has always been some flavour of word sense disambiguation (WSD), which in its standard form boils down to choosing the best-possible fit from a pre-defined sense inventory. In recent years, it has become clear that this is in fact a very hard task to solve for computers and humans alike (Ide and Wilks, 2006; Erk et al., 2009; Erk, 2010).

With these findings in mind, researchers have started looking at different methods to tackle language's ambiguity, ranging from coarser-grained sense inventories (Hovy et al., 2006) and graded sense assignment (Erk and McCarthy, 2009), over word sense induction (Schütze, 1998; Pantel and Lin, 2002; Agirre et al., 2006), to the computation of individual word meaning in context (Erk and Padó, 2008; Thater et al., 2010; Dinu and Lapata, 2010). This research inscribes itself in the same line of thought, in which the meaning disambiguation of a word is not just the assignment of a pre-defined sense; instead, the original meaning representation of a word is adapted 'on the fly', according to – and specifically tailored for – the particular context in which it appears. To be able to do so, we build a factorization model in which words, together with their window-based context words and their dependency

Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 1012–1022,
Edinburgh, Scotland, UK, July 27–31, 2011. ©2011 Association for Computational Linguistics

relations, are linked to latent dimensions. The factorization model allows us to determine which dimensions are important for a particular context, and adapt the dependency-based feature vector of the word accordingly. The evaluation on a lexical substitution task – carried out for both English and French – indicates that our method is able to reach better results than state-of-the-art methods in lexical substitution, while at the same time providing more accurate meaning representations.

The remainder of this paper is organized as follows. In section 2, we present some earlier work that is related to the research presented here. Section 3 describes the methodology of our method, focusing on the factorization model, and the computation of meaning in context. Section 4 presents a thorough evaluation on a lexical substitution task, both for English and French. The last section then draws conclusions, and presents a number of topics that deserve further exploration.

2 Related work

One of the best known computational models of semantic similarity is latent semantic analysis — LSA (Landauer and Dumais, 1997; Landauer et al., 1998). In LSA, a term-document matrix is created, that contains the frequency of each word in a particular document. This matrix is then decomposed into three other matrices with a mathematical factorization technique called singular value decomposition (SVD). The most important dimensions that come out of the SVD are said to represent latent semantic dimensions, according to which nouns and documents can be represented more efficiently. Our model also applies a factorization technique (albeit a different one) in order to find a reduced semantic space.

The nature of a word's context is a determining factor in the kind of the semantic similarity that is induced. A broad context window (e.g. a paragraph or document) yields broad, topical similarity, whereas a small context window yields tight, synonym-like similarity. This has lead a number of researchers (e.g. Lin (1998)) to use the dependency relations that a particular word takes part in as context features. An overview of dependency-based semantic space models is given in Padó and Lapata (2007).

A number of researchers have exploited the notion of context to differentiate between the different senses of a word in an unsupervised way (a task labeled word sense induction or WSI). Schütze (1998) proposed a context-clustering approach, in which context vectors are created for the different instances of a particular word, and those contexts are grouped into a number of clusters, representing the different senses of the word. The context vectors are represented as second-order co-occurrences (i.e. the contexts of the target word are similar if the words they in turn co-occur with are similar). Van de Cruys (2008) proposed a model for sense induction based on latent semantic dimensions. Using a factorization technique based on non-negative matrix factorization, the model induces a latent semantic space according to which both dependency features and broad contextual features are classified. Using the latent space, the model is able to discriminate between different word senses. Our approach makes use of a similar factorization model, but we extend the approach with a probabilistic framework that is able to adapt the original vector according to the context of the instance.

Recently, a number of models emerged that aim to model the individual meaning of words in context. Erk and Padó (2008, 2009) make use of selectional preferences to express the meaning of a word in context; the meaning of a word in the presence of an argument is computed by multiplying the word's vector with a vector that captures the inverse selectional preferences of the argument. Thater et al. (2009) and Thater et al. (2010) extend the approach based on selectional preferences by incorporating second-order co-occurrences in their model; their model allows first-order co-occurrences to act as a filter upon the second-order vector space, which allows for the computation of meaning in context.

Erk and Padó (2010) propose an exemplar-based approach, in which the meaning of a word in context is represented by the activated exemplars that are most similar to it. And Mitchell and Lapata (2008) propose a model for vector composition, focusing on the different functions that might be used to combine the constituent vectors. Their results indicate that a model based on pointwise multiplication achieves better results than models based on vector addition.

Finally, Dinu and Lapata (2010) propose a probabilistic framework that models the meaning of words as a probability distribution over latent dimensions ('senses'). Contextualized meaning is then modeled as a change in the original sense distribution. The model presented in this paper bears some resemblances to their approach; however, while their approach computes the contextualized meaning directly within the latent space, our model exploits the latent space to determine the features that are important for a particular context, and adapt the original (out-of-context) dependency-based feature vector of the target word accordingly. This allows for a more precise and more distinct computation of word meaning in context. Secondly, Dinu and Lapata use window-based context features to build their latent model, while our approach combines both window-based and dependency-based features.

3 Methodology

3.1 Non-negative Matrix Factorization

Our model uses non-negative matrix factorization (Lee and Seung, 2000) in order to find latent dimensions. There are a number of reasons to prefer NMF over the better known singular value decomposition used in LSA. First of all, NMF allows us to minimize the Kullback-Leibler divergence as an objective function, whereas SVD minimizes the Euclidean distance. The Kullback-Leibler divergence is better suited for language phenomena. Minimizing the Euclidean distance requires normally distributed data, and language phenomena are typically not normally distributed. Secondly, the non-negative nature of the factorization ensures that only additive and no subtractive relations are allowed. This proves particularly useful for the extraction of semantic dimensions, so that the NMF model is able to extract much more clear-cut dimensions than an SVD model. And thirdly, the non-negative property allows the resulting model to be interpreted probabilistically, which is not straightforward with an SVD factorization.

The key idea is that a non-negative matrix $\mathbf{A}$ is factorized into two other non-negative matrices, $\mathbf{W}$ and $\mathbf{H}$

$$\mathbf{A}_{i\times j} \approx \mathbf{W}_{i\times k}\mathbf{H}_{k\times j} \tag{1}$$

where k is much smaller than i, j so that both instances and features are expressed in terms of a few components. Non-negative matrix factorization enforces the constraint that all three matrices must be non-negative, so all elements must be greater than or equal to zero.

Using the minimization of the Kullback-Leibler divergence as an objective function, we want to find the matrices $\mathbf{W}$ and $\mathbf{H}$ for which the Kullback-Leibler divergence between $\mathbf{A}$ and $\mathbf{WH}$ (the multiplication of $\mathbf{W}$ and $\mathbf{H}$) is the smallest. This factorization is carried out through the iterative application of update rules. Matrices $\mathbf{W}$ and $\mathbf{H}$ are randomly initialized, and the rules in 2 and 3 are iteratively applied – alternating between them. In each iteration, each vector is adequately normalized, so that all dimension values sum to 1.

$$\mathbf{H}_{a\mu} \leftarrow \mathbf{H}_{a\mu} \frac{\sum_i \mathbf{W}_{ia} \frac{\mathbf{A}_{i\mu}}{(\mathbf{WH})_{i\mu}}}{\sum_k \mathbf{W}_{ka}} \tag{2}$$

$$\mathbf{W}_{ia} \leftarrow \mathbf{W}_{ia} \frac{\sum_\mu \mathbf{H}_{a\mu} \frac{\mathbf{A}_{i\mu}}{(\mathbf{WH})_{i\mu}}}{\sum_v \mathbf{H}_{av}} \tag{3}$$

3.2 Combining syntax and context words

Using an extension of non-negative matrix factorization (Van de Cruys, 2008), it is possible to jointly induce latent factors for three different modes: words, their window-based context words, and their dependency-based context features. As input to the algorithm, three matrices are constructed that capture the pairwise co-occurrence frequencies for the different modes. The first matrix contains co-occurrence frequencies of words cross-classified by dependency-based features, the second matrix contains co-occurrence frequencies of words cross-classified by words that appear in the word's context window, and the third matrix contains co-occurrence frequencies of dependency-based features cross-classified by co-occurring context words. NMF is then applied to the three matrices, and the separate factorizations are interleaved (i.e. the results of the former factorization are used to initialize the factorization of the next matrix). A graphical representation of the interleaved factorization algorithm is given in figure 1.

When the factorization is finished, the three different modes (words, window-based context words and dependency-based features) are all represented according to a limited number of latent factors.

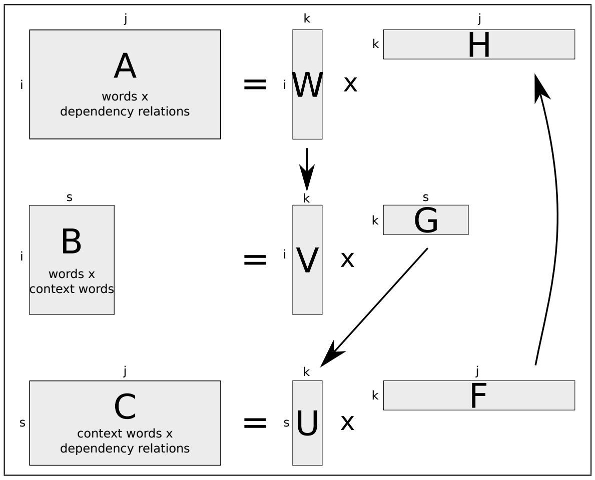

Figure 1: A graphical representation of the interleaved NMF

The factorization that comes out of the NMF model can be interpreted probabilistically (Gaussier and Goutte, 2005; Ding et al., 2008). More specifically, we can transform the factorization into a standard latent variable model of the form

$$p(w_i, d_j) = \sum_{z=1}^{K} p(z)p(w_i|z)p(d_j|z) \qquad (4)$$

by introducing two $K \times K$ diagonal scaling matrices $\mathbf{X}$ and $\mathbf{Y}$, such that $\mathbf{X}_{kk} = \sum_i \mathbf{W}_{ik}$ and $\mathbf{Y}_{kk} = \sum_j \mathbf{H}_{kj}$. The factorization $\mathbf{WH}$ can then be rewritten as

$$\begin{aligned} \mathbf{WH} &= (\mathbf{WX}^{-1}\mathbf{X})(\mathbf{YY}^{-1}\mathbf{H}) \\ &= (\mathbf{WX}^{-1})(\mathbf{XY})(\mathbf{Y}^{-1}\mathbf{H}) \end{aligned} \qquad (5)$$

such that $\mathbf{WX}^{-1}$ represents $p(w_i|\mathbf{z})$, $(\mathbf{Y}^{-1}\mathbf{H})^T$ represents $p(d_j|\mathbf{z})$, and $\mathbf{XY}$ represents $p(\mathbf{z})$. Using Bayes' theorem, it is now straightforward to determine $p(\mathbf{z}|w_i)$ and $p(\mathbf{z}|d_j)$.

$$p(\mathbf{z}|w_i) = \frac{p(w_i|\mathbf{z})p(\mathbf{z})}{p(w_i)} \qquad (6)$$

$$p(\mathbf{z}|d_j) = \frac{p(d_j|\mathbf{z})p(\mathbf{z})}{p(d_j)} \qquad (7)$$

3.3 Meaning in Context

3.3.1 Overview

Using the results of the factorization model described above, we can now adapt a word's feature vector according to the context in which it appears. Intuitively, the contextual features of the word (i.e. the window-based context words or dependency-based context features) pinpoint the important semantic dimensions of the particular instance, creating a probability distribution over latent factors. For a number of context words of a particular instance, we determine the probability distribution over latent factors given the context, $p(\mathbf{z}|C)$, as the average of the context words' probability distributions over latent factors (equation 8).

$$p(\mathbf{z}|C) = \frac{\sum_{w_i \in C} p(\mathbf{z}|w_i)}{|C|} \qquad (8)$$

The probability distribution over latent factors given a number of dependency-based context features can be computed in a similar fashion, replacing w_i with d_j. Additionally, this step allows us to combine both windows-based context words and dependency-based context features in order to determine the latent probability distribution (e.g. by taking a linear combination).

The resulting probability distribution over latent factors can be interpreted as a semantic fingerprint of the passage in which the target word appears. Using this fingerprint, we can now determine a new probability distribution over dependency features given the context.

$$p(\mathbf{d}|C) = p(\mathbf{z}|C)p(\mathbf{d}|\mathbf{z}) \qquad (9)$$

The last step is to weight the original probability vector of the word according to the probability vector of the dependency features given the word's context, by taking the pointwise multiplication of probability vectors $p(\mathbf{d}|w_i)$ and $p(\mathbf{d}|C)$.

$$p(\mathbf{d}|w_i, C) = p(\mathbf{d}|w_i) \cdot p(\mathbf{d}|C) \qquad (10)$$

Note that this final step is a crucial one in our approach. We do not just build a model based on latent factors, but we use the latent factors to determine which of the features in the original word vector are the salient ones given a particular context. This allows us to compute an accurate adaptation of the original word vector in context.

Also note the resemblance to Mitchell and Lapata's best scoring vector composition model which, likewise, uses pointwise multiplication. However, the model presented here has two advantages. First of all, it allows to take multiple context features into account, each of which contributes to the probability distribution over latent factors. Secondly, the target word and its features do not need to live in the same vector space (i.e. they do not need to be defined according to the same features), as the connections and the appropriate weightings are determined through the latent model.

3.3.2 Example

Let us exemplify the procedure with an example. Say we want to compute the distributionally similar words to the noun *record* in the context of example sentences (3) and (4).

(3) Jack is listening to a record.

(4) Jill updated the record.

First, we extract the context features for both instances, in this case $C_1 = \{listen^{-1}_{prep(to)}\}$ for sentence (3), and $C_2 = \{update^{-1}_{obj}\}$ for sentence (4).[1] Next, we compute $p(\mathbf{z}|C_1)$ and $p(\mathbf{z}|C_2)$ – the probability distributions over latent factors given the context – by averaging over the latent probability distributions of the individual context features.[2] Using these probability distributions over latent factors, we can now determine the probability of each dependency feature given the different contexts – $p(\mathbf{d}|C_1)$ and $p(\mathbf{d}|C_2)$.

The former step yields a general probability distribution over dependency features that tells us how likely a particular dependency feature is given the context that our target word appears in. Our last step is now to weight the original probability vector of the target word (the aggregate of dependency-based context features over all contexts of the target word) according to the new distribution given the context in which the target word appears. For the first sentence, features associated with the music sense of *record* (or more specifically, the dependency features associated with latent factors that are related to the feature $\{listen^{-1}_{prep(to)}\}$) will be emphasized, while features associated with unrelated latent factors are leveled out. For the second sentence, features that are associated with the administrative sense of *record* (dependency features associated with latent factors that are related to the feature $\{update^{-1}_{obj}\}$) are emphasized, while unrelated features are played down.

We can now return to our original matrix $\mathbf{A}$ and compute the top similar words for the two adapted vectors of *record* given the different contexts, which yields the results presented below.

1. **record**$_N$, C_1: *album*, *song*, *recording*, *track*, *cd*

2. **record**$_N$, C_2: *file*, *datum*, *document*, *database*, *list*

4 Evaluation

In this section, we present a thorough evaluation of the method described above, and compare it with related methods for meaning computation in context. In order to test the applicability of the method to multiple languages, we present evaluation results for both English and French.

4.1 Datasets

For English, we make use of the SEMEVAL 2007 English Lexical Substitution task (McCarthy and Navigli, 2007; McCarthy and Navigli, 2009). The task's goal is to find suitable substitutes for a target word in a particular context. The complete data set contains 200 target words (about 50 for each part of speech, viz. nouns, verbs, adjectives, and adverbs). Each target word occurs in 10 different sentences, which yields a total of 2000 sentences. Five annotators provided suitable substitutes for each target word in the different contexts.

For French, we developed a small-scale lexical substitution task ourselves, closely following the guidelines of the original English task. We manually selected 10 ambiguous French nouns, and for each noun we selected 10 different sentences from the FRWaC corpus (Baroni et al., 2009). Four different native French speakers were then asked to provide suitable substitutes for the nouns in context.[3]

[1] In this example we use dependency features, but the computations are similar for window-based context words.

[2] In this case, the sets of context features contain only one item, so the average probability distribution of the sets is just the latent probability distribution of their respective item.

[3] The task is provided as supplementary material to this paper; it is also available from the first author's website.

4.2 Implementational details

The model for English has been trained on part of the UKWaC corpus (Baroni et al., 2009), covering about 500M words. The corpus has been part of speech tagged and lemmatized with Stanford Part-Of-Speech Tagger (Toutanova and Manning, 2000; Toutanova et al., 2003), and parsed with MaltParser (Nivre et al., 2006) trained on sections 2-21 of the Wall Street Journal section of the Penn Treebank extended with about 4000 questions from the QuestionBank[4], so that dependency triples could be extracted. The sentences of the English lexical substitution task have been tagged, lemmatized and parsed in the same way. The model for French has been trained on the French version of Wikipedia (± 100M words), parsed with the FRMG parser (Villemonte de La Clergerie, 2010) for French.

For English, we built different models for each part of speech (nouns, verbs, adjectives and adverbs), which yields four models in total. For each model, the matrices needed for our interleaved NMF factorization are extracted from the corpus. The noun model, for example, was built using 5K nouns, 80K dependency relations, and 2K context words[5] (excluding stop words) with highest frequency in the training set, which yields matrices of 5K nouns × 80K dependency relations, 5K nouns × 2K context words, and 80K dependency relations × 2K context words. The models for the three other parts of speech were constructed in a similar vein. For French, we only constructed a model for nouns, as our lexical substitution task for French is limited to this part of speech.

The interleaved NMF model was carried out using $K = 600$ (the number of factorized dimensions in the model), and applying 100 iterations.[6] The interleaved NMF algorithm was implemented in Matlab; the preprocessing scripts and scripts for vector computation in context were written in Python. Cosine was used as a similarity measure.

4.3 Measures

Up till now, most researchers have interpreted the lexical substitution task as a ranking problem, in which the possible substitutes are given beforehand and the goal is to rank the substitutes according to their suitability in a particular context, so that sound substitutes are given a higher rank than their non-suitable counterparts. This means that all possible substitutes for a given target word (extracted from the gold standard) are lumped together, and the system then has to produce a ranking for the complete set of substitutes.

We also adopt this approach in our evaluation framework, but we complement it with the original evaluation measures of the lexical substitution task, in which the system is not given a list of possible substitutes beforehand, but has to come up with the suitable candidates itself. This is a much harder task, but we believe that such an approach is more compelling in assessing the system's ability to induce a proper meaning representation for word usage in context. We coin the former approach *paraphrase ranking*, and the latter one *paraphrase induction*. In the next paragraphs, we will describe the actual evaluation measures that have been used for both approaches.

Paraphrase ranking Following Dinu and Lapata (2010), we compare the ranking produced by our model with the gold standard ranking using Kendall's τ_b (which is adjusted for ties). For reasons of comparison, we also compute general average precision (GAP, Kishida (2005)), which was used by Erk and Padó (2010) and Thater et al. (2010) to evaluate their rankings. Differences between models are tested for significance using stratified shuffling (Yeh, 2000), using a standard number of 10000 iterations.

We compare the results for paraphrase ranking to two different baselines. The first baseline is a random one, in which the gold standard is compared to an arbitrary ranking. The second baseline is a dependency-based vector space model that does not take the context of the particular instance into account (and thus returns the same ranking for each instance of the target word). This is a fairly competitive baseline, as noted by other researchers (Erk and Padó, 2008; Thater et al., 2009; Dinu and Lapata, 2010).

[4] http://maltparser.org/mco/english_parser/engmalt.html

[5] We used a fairly large, paragraph-like window of four sentences.

[6] We experimented with different values (in the range 300–1500) for K, but the models did not seem to improve much beyond $K = 600$; hence, we stuck with 600 factors, due to speed and memory advantages of a lower number of factors.

Paraphrase induction To evaluate the system's ability to come up with suitable substitutes from scratch, we use the measures designed to evaluate systems that took part in the original English lexical substitution task (McCarthy and Navigli, 2007). Two different measures were used, which were coined *best* and *out-of-ten (oot)*. The strict *best* measure allows the system to give as many candidate substitutes as it considers appropriate, but the credit for each correct substitute is divided by the total number of guesses. Recall is then calculated as the average annotator response frequency of substitutes found by the system over all items T.

$$R_{best} = \frac{\sum_{s \in M \cap G} f(s)}{|M| \cdot \sum_{s \in G} f(s)} \tag{11}$$

where M is the system's candidate list[7], G is the gold-standard data, and $f(s)$ is the annotator response frequency of the candidate.

The out-of-ten measure is more liberal; it allows the system to give up to ten substitutes, and the credit for each correct substitute is not divided by the total number of guesses. The more liberal measure was introduced to account for the fact that the lexical substitution task's gold standard is susceptible to a considerate amount of variation, and there is only a limited number of annotators.

$$P_{10} = \frac{\sum_{s \in M \cap G} f(s)}{\sum_{s \in G} f(s)} \tag{12}$$

where M is the system's list of 10 candidates, and G and $f(s)$ are the same as above. Because we only use the best guess with R_{best}, the two measures are exactly the same except for the number of candidates M.

4.4 Results

4.4.1 English

Table 1 presents the paraphrase ranking results of our approach, comparing them to the two baselines and to a number of previous approaches to meaning computation in context.

The first two models represent our baselines. The first baseline is the random baseline, where the candidate substitutes are ranked randomly (τ_b close to zero indicates that there is no correlation). The second baseline is a standard dependency-based vector space model, which yields the same ranking for all instances of a target word. Note that the second baseline is a rather competitive one.

model	τ_b	GAP
random	-0.61	29.98
vector$_{dep}$	16.57	45.08
EP09	–	32.2 $^{\blacktriangledown}$
EP10	–	39.9 $^{\blacktriangledown}$
TFP	–	45.94$^{\blacktriangledown}$
DL	16.56	41.68
NMF$_{context}$	20.64**	47.60**
NMF$_{dep}$	22.49**	48.97**
NMF$_{c+d}$	**22.59****	**49.02****

Table 1: Kendall's τ_b and GAP paraphrase ranking scores for the English lexical substitution task. Scores marked with '$^{\blacktriangledown}$' are copied from the authors' respective papers. Scores marked with '**' are statistically significant with $p < 0.01$ compared to the second baseline.

The next four models represent previous approaches to meaning computation in context. EP09 is Erk and Pado's (2009) selectional preference approach; EP10 is Erk and Pado's (2010) exemplar-based approach; TFP stands for Thater et al.'s (2010) approach; and DL is Dinu and Lapata's (2010) latent modeling approach. The results are reproduced from their respective papers, except for Dinu and Lapata's approach, which we reimplemented ourselves.[8] Note that the reproduced results (EP09, EP10 and TFP) are not entirely comparable, because the authors only use a subset of the lexical substitution task.

The last three models are instantiations of our approach: NMF$_{context}$ is a model that uses window-based context features, NMF$_{dep}$ is a model that uses dependency-based context features, and NMF$_{c+d}$ is a model that uses a linear combination of window-based and dependency-based context features, giving equal weight to both.

The three instantiations of our approach reach better results than all previous approaches. Moreover, our approach is the only one able to significantly

[7] In our evaluations, we calculate *best* using the system's best guess only, so the candidate list contains only one item.

[8] The original paper reports a slightly lower τ_b of 16.01 for their best scoring model.

beat our second (competitive) baseline of a standard dependency-based vector model. Comparing our three instantiations, the model that combines window-based context and dependency-based context scores best, closely followed by the dependency-based model. The model that only uses window-based context gets the lowest score of the three, but is still fairly competitive compared to the previous approaches. The differences between the models are statistically significant ($p < 0.01$), except for the difference between NMF$_{dep}$ and NMF$_{c+d}$.

model	n	v	a	r
vector$_{dep}$	15.85	11.68	16.71	25.29
NMF$_{context}$	20.58	16.24	21.00	27.22
NMF$_{dep}$	21.96	17.33	**24.57**	28.16
NMF$_{c+d}$	**22.68**	**17.47**	23.84	**28.66**

Table 2: Kendall's τ_b paraphrase ranking scores for the English lexical substitution task across different parts of speech

Table 2 shows the performance of the three model instantiations on paraphrase ranking across different parts of speech. The results largely confirm tendencies reported by other researchers (cfr. Dinu and Lapata (2010)), viz. that verbs are the most difficult, followed by nouns and adjectives. These parts of speech also benefit the most from the use of a contextualized model. Adverbs are easier, but there is less to be gained from using contextualized models.

model	R_{best}	P_{10}
vector$_{dep}$	8.78	30.21
DL	1.06	7.59
KU	20.65	46.15
IRST2	20.33	68.90
NMF$_{context}$	8.81	**30.49**
NMF$_{dep}$	7.73	26.92
NMF$_{c+d}$	**8.96**	29.26

Table 3: R_{best} and P_{10} paraphrase induction scores for the English lexical substitution task

Table 3 shows the performance of the different models on the paraphrase induction task. Note once again that our baseline vector$_{dep}$ – a simple dependency-based vector space model – is a highly competitive one. NMF$_{context}$ and NMF$_{c+d}$ are able to reach marginally better results, but the differences are not statistically significant. However, all of our models are able to reach much better results than Dinu and Lapata's approach. The results indicate that our approach, after vector adaptation in context, is still able to provide accurate similarity calculations across the complete word space. While other algorithms are able to rank candidate substitutes at the expense of accurate similarity calculations, our approach is able to do both. This is one of the important advantages of our approach.

For reasons of comparison, we also included the scores of the best performing models that participated in the SEMEVAL 2007 lexical substitution task (KU (Yuret, 2007) and IRST2 (Giuliano et al., 2007), which got the best scores for R_{best} and P_{10}, respectively). These models reach better scores compared to our models. Note, however, that all participants of the SEMEVAL 2007 lexical substitution task relied on a predefined sense inventory (i.e. WordNet, or a machine readable thesaurus). Our system, on the other hand, induces paraphrases in a fully unsupervised way. To our knowledge, this is the first time a fully unsupervised system is tested on the paraphrase induction task.

model	n	v	a	r
vector$_{dep}$	31.66	23.53	29.91	38.43
NMF$_{context}$	**33.73****	25.21*	28.58	36.45
NMF$_{dep}$	31.40	25.97**	20.56	31.48
NMF$_{c+d}$	33.37*	**25.99****	24.20	35.81

Table 4: P_{10} paraphrase induction scores for the English lexical substitution task across different parts of speech. Scores marked with '$\star\star$' and '$\star$' are statistically significant with respectively $p < 0.01$ and $p < 0.05$ compared to the baseline.

Table 4 presents the results for paraphrase induction (*oot*) across the different parts of speech. The results indicate that paraphrase induction works best for nouns and verbs, with statistically significant improvements over the baseline. The differences among the models themselves are not significant. Adjectives and adverbs yield lower scores, indicating that their contextualization yields less precise vectors for meaning computation. Note, however, that the NMF$_{context}$ model is still quite apt for meaning computation, yielding results that are only slightly lower than the dependency-based vector space model.

4.4.2 French

This section presents the results on the French lexical substitution task. Table 5 presents the results for paraphrase ranking, while table 6 shows the models' performance on the paraphrase induction task.

model	Kendall's τ_b	GAP
vector$_{dep}$	7.79	36.46
DL	17.99	41.73
NMF$_{context}$	**18.63**	**44.96**
NMF$_{dep}$	17.15	44.66
NMF$_{c+d}$	18.40	43.14

Table 5: Kendall's τ_b and GAP paraphrase ranking scores for the French lexical substitution task

The results for paraphrase ranking in French (table 5) show similar tendencies as the results for English: all of our models are able to improve significantly over the dependency-based vector space baseline. Note, however, thar our models generally score a bit lower compared to the English results. This drop in performance is not present for Dinu and Lapata's model. The difference might be due to the difference in corpora size: for the method to operate at full power, we need to make a good estimate of the cooccurrences of three modes (words, window-based context words and dependency-based features), and thus our methods requires a significant amount of data. Nevertheless, our approach still yields the best results, with NMF$_{context}$ as the best scoring model.

Finally, the results for paraphrase induction in French (table 6) interestingly show a significant and large improvement over the baseline. The improvements indicate once again that the models are able to carry out precise similarity computations over the whole word space, while at the same time providing an adequately adapted contextualized meaning vector. Dinu and Lapata's model, which performs similarity calculations in the latent space, is not able to provide accurate word vectors, and thus perform worse at the paraphrase induction task.

model	R_{best}	P_{10}
vector$_{dep}$	6.38	24.43
DL	0.50	5.34
NMF$_{context}$	**10.71**	31.42
NMF$_{dep}$	9.65	28.52
NMF$_{c+d}$	10.64	**35.32**

Table 6: R_{best} and P_{10} paraphrase induction scores for the French lexical substitution task

5 Conclusion

In this paper, we presented a novel method for the modeling of word meaning in context. We make use of a factorization model based on non-negative matrix factorization, in which words, together with their window-based context words and their dependency relations, are linked to latent dimensions. The factorization model allows us to determine which particular dimensions are important for a target word in a particular context. A key feature of the algorithm is that we adapt the original dependency-based feature vector of the target word through the latent semantic space. By doing so, our model is able to make accurate similarity calculations for word meaning in context across the whole word space. Our evaluation shows that the approach presented here is able to improve upon the state-of-the art performance on paraphrase ranking. Moreover, our approach scores well for both paraphrase ranking and paraphrase induction, whereas previous approaches only seem capable of improving performance on the former task at the expense of the latter.

During our research, a number of topics surfaced that we consider worth exploring in the future. First of all, we would like to further investigate the optimal configuration for combining window-based and dependency-based contexts. At the moment, the performance of the combined model does not yield a uniform picture. The results might improve further if window-based context and dependency-based context are combined in an optimal way. Secondly, we would like to subject our approach to further evaluation, in particular on a number of different evaluation tasks, such as semantic compositionality. And thirdly, we would like to transfer the general idea of the approach presented in this paper to a tensorbased framework (which is able to capture the multi-way co-occurrences of words, together with their window-based and dependency-based context features, in a natural way) and investigate whether such a framework proves beneficial for the modeling of word meaning in context.

Acknowledgements

The work reported in this paper was funded by the Isaac Newton Trust (Cambridge, UK), the EU FP7 project 'PANACEA', the EPSRC grant EP/G051070/1 and the Royal Society (UK).

References

Eneko Agirre, David Martínez, Oier López de Lacalle, and Aitor Soroa. 2006. Two graph-based algorithms for state-of-the-art wsd. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP) Conference*, pages 585–593, Sydney, Australia.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Chris Ding, Tao Li, and Wei Peng. 2008. On the equivalence between non-negative matrix factorization and probabilistic latent semantic indexing. *Computational Statistics & Data Analysis*, 52(8):3913–3927.

Georgiana Dinu and Mirella Lapata. 2010. Measuring distributional similarity in context. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1162–1172, Cambridge, MA, October.

Katrin Erk and Diana McCarthy. 2009. Graded word sense assignment. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 440–449, Suntec, Singapore.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 897–906, Waikiki, Hawaii, USA.

Katrin Erk and Sebastian Padó. 2009. Paraphrase assessment in structured vector space: Exploring parameters and datasets. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 57–65, Athens, Greece.

Katrin Erk and Sebastian Padó. 2010. Exemplar-based models for word meaning in context. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 92–97, Uppsala, Sweden.

Katrin Erk, Diana McCarthy, and Nicholas Gaylord. 2009. Investigations on word senses and word usages. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 10–18.

Katrin Erk. 2010. What is word meaning, really? (and how can distributional models help us describe it?). In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, pages 17–26.

Eric Gaussier and Cyril Goutte. 2005. Relation between PLSA and NMF and implications. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 601–602, Salvador, Brazil.

Claudio Giuliano, Alfio Gliozzo, and Carlo Strapparava. 2007. Fbk-irst: Lexical substitution task exploiting domain and syntagmatic coherence. In *Proceedings of the Fourth International Workshop on Semantic Evaluations*, pages 145–148.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: the 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 57–60, New York, New York, USA.

Nancy Ide and Yorick Wilks. 2006. Making Sense About Sense. In *Word Sense Disambiguation: Algorithms And Applications*, chapter 3. Springer, Dordrecht.

Kazuaki Kishida. 2005. Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments. Technical report, National Institute of Informatics.

Thomas Landauer and Susan Dumais. 1997. A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychology Review*, 104:211–240.

Thomas Landauer, Peter Foltz, and Darrell Laham. 1998. An Introduction to Latent Semantic Analysis. *Discourse Processes*, 25:295–284.

Daniel D. Lee and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, pages 556–562.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL98), Volume 2*, pages 768–774, Montreal, Quebec, Canada.

Diana McCarthy and Roberto Navigli. 2007. SemEval-2007 task 10: English lexical substitution task. In

Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007), pages 48–53.

Diana McCarthy and Roberto Navigli. 2009. The English lexical substitution task. *Language resources and evaluation*, 43(2):139–159.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. *proceedings of ACL-08: HLT*, pages 236–244.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC-2006*, pages 2216–2219.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 613–619, Edmonton, Alberta, Canada.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Stefan Thater, Georgiana Dinu, and Manfred Pinkal. 2009. Ranking paraphrases in context. In *Proceedings of the 2009 Workshop on Applied Textual Inference*, pages 44–47, Suntec, Singapore.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 948–957, Uppsala, Sweden.

Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*, pages 63–70.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL 2003*, pages 252–259.

Tim Van de Cruys. 2008. Using three way data for word sense discrimination. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 929–936, Manchester.

Eric Villemonte de La Clergerie. 2010. Building factorized TAGs with meta-grammars. In *Proceedings of the 10th International Conference on Tree Adjoining Grammars and Related Formalisms (TAG+10)*, pages 111–118, New Haven, Connecticut, USA.

Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics*, pages 947–953, Saarbrücken, Germany.

Deniz Yuret. 2007. Ku: Word sense disambiguation by substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations*, pages 207–213.