Modeling Sentences in the Latent Space

Weiwei Guo
Department of Computer Science,
Columbia University,
weiwei@cs.columbia.edu

Mona Diab
Center for Computational Learning Systems,
Columbia University,
mdiab@ccls.columbia.edu

Abstract

Sentence Similarity is the process of computing a similarity score between two sentences. Previous sentence similarity work finds that latent semantics approaches to the problem do not perform well due to insufficient information in single sentences. In this paper, we show that by carefully handling words that are **not** in the sentences (missing words), we can train a reliable latent variable model on sentences. In the process, we propose a new evaluation framework for sentence similarity: Concept Definition Retrieval. The new framework allows for large scale tuning and testing of Sentence Similarity models. Experiments on the new task and previous data sets show significant improvement of our model over baselines and other traditional latent variable models. Our results indicate comparable and even better performance than current state of the art systems addressing the problem of sentence similarity.

1 Introduction

Identifying the degree of semantic similarity [SS] between two sentences is at the core of many NLP applications that focus on sentence level semantics such as Machine Translation (Kauchak and Barzilay, 2006), Summarization (Zhou et al., 2006), Text Coherence Detection (Lapata and Barzilay, 2005), etc.To date, almost all Sentence Similarity [SS] approaches work in the high-dimensional word space and rely mainly on word similarity. There are two main (not unrelated) disadvantages to word similarity based approaches: 1. lexical ambiguity as the pairwise word similarity ignores the semantic interaction between the word and its sentential context; 2. word co-occurrence information is not sufficiently exploited.

Latent variable models, such as Latent Semantic Analysis [LSA] (Landauer et al., 1998), Probabilistic Latent Semantic Analysis [PLSA] (Hofmann, 1999), Latent Dirichlet Allocation [LDA] (Blei et al., 2003) can solve the two issues naturally by modeling the semantics of words and sentences simultaneously in the low-dimensional latent space. However, attempts at addressing SS using LSA perform significantly below high dimensional word similarity based models (Mihalcea et al., 2006; O'Shea et al., 2008).

We believe that the latent semantics approaches applied to date to the SS problem have not yielded positive results due to the deficient modeling of the sparsity in the semantic space. SS operates in a very limited contextual setting where the sentences are typically very short to derive robust latent semantics. Apart from the SS setting, robust modeling of the latent semantics of short sentences/texts is becoming a pressing need due to the pervasive presence of more bursty data sets such as Twitter feeds and SMS where short contexts are an inherent characteristic of the data.

In this paper, we propose to model the missing words (words that are not in the sentence), a feature that is typically overlooked in the text modeling literature, to address the sparseness issue for the SS task. We define the missing words of a sentence as the whole vocabulary in a corpus minus the observed words in the sentence. Our intuition is since observed words in a sentence are too few to tell us what the sentence is about, missing words can be used to tell us what the sentence is **not** about. We assume that the semantic space of both the observed

and missing words make up the complete semantics profile of a sentence.

After analyzing the way traditional latent variable models (LSA, PLSA/LDA) handle missing words, we decide to model sentences using a weighted matrix factorization approach (Srebro and Jaakkola, 2003), which allows us to treat observed words and missing words differently. We handle missing words using a weighting scheme that distinguishes missing words from observed words yielding robust latent vectors for sentences.

Since we use a feature that is already implied by the text itself, our approach is very general (similar to LSA/LDA) in that it can be applied to any format of short texts. In contrast, existing work on modeling short texts focuses on exploiting additional data, e.g., Ramage et al. (2010) model tweets using their metadata (author, hashtag, etc.).

Moreover in this paper, we introduce a new evaluation framework for SS: Concept Definition Retrieval (CDR). Compared to existing data sets, the CDR data set allows for large scale tuning and testing of SS modules without further human annotation.

2 Limitations of Topic Models and LSA for Modeling Sentences

Usually latent variable models aim to find a latent semantic profile for a sentence that is most relevant to the observed words. By explicitly modeling missing words, we set another criterion to the latent semantics profile: it should **not** be related to the missing words in the sentence. However, missing words are not as informative as observed words, hence the need for a model that does a good job of modeling missing words at the right level of **emphasis/impact** is central to completing the semantic picture for a sentence.

LSA and PLSA/LDA work on a word-sentence co-occurrence matrix. Given a corpus, the row entries of the matrix are the unique M words in the corpus, and the N columns are the sentence ids. The yielded $M \times N$ co-occurrence matrix X comprises the TF-IDF values in each X_{ij} cell, namely that TF-IDF value of word w_i in sentence s_j. For ease of exposition, we will illustrate the problem using a special case of the SS framework where the sentences are concept definitions in a dictionary such as WordNet (Fellbaum, 1998) (WN). Therefore, the sentence corresponding to the concept definition of $bank\#n\#1$ is a sparse vector in X containing the following observed words where $X_{ij} \neq 0$:
the 0.1, $financial$ 5.5, $institution$ 4, $that$ 0.2, $accept$ 2.1, $deposit$ 3, and 0.1, $channel$ 6, the 0.1, $money$ 5, $into$ 0.3, $lend$ 3.5, $activity$ 3

All the other words (*girl, car,..., check, loan, business,...*) in matrix X that do not occur in the concept definition are considered missing words for the concept entry $bank\#n\#1$, thereby their $X_{ij} = 0$.

Topic models (PLSA/LDA) do not explicitly model missing words. PLSA assumes each document has a distribution over K topics $P(z_k|d_j)$, and each topic has a distribution over all vocabularies $P(w_i|z_k)$. Therefore, PLSA finds a topic distribution for each concept definition that maximizes the log likelihood of the corpus X (LDA has a similar form):

$$\sum_i \sum_j X_{ij} \log \sum_k P(z_k|d_j) P(w_i|z_k) \qquad (1)$$

In this formulation, missing words do not contribute to the estimation of sentence semantics, i.e., excluding missing words ($X_{ij} = 0$) in equation 1 does not make a difference.

However, empirical results show that given a small number of observed words, usually topic models can only find one topic (most evident topic) for a sentence, e.g., the concept definitions of $bank\#n\#1$ and $stock\#n\#1$ are assigned the financial topic only without any further discernability. This results in many sentences are assigned exactly the same semantics profile as long as they are pertaining/mentioned within the same domain/topic. The reason is topic models try to learn a 100-dimension latent vector (assume dimension $K = 100$) from very few data points (10 observed words on average). It would be desirable if topic models can exploit missing words (a lot more data than observed words) to render more nuanced latent semantics, so that pairs of sentences in the same domain can be differentiable.

On the other hand, LSA explicitly models missing words but not at the right level of emphasis. LSA finds another matrix $\hat{X}$ (latent vectors) with rank K to approximate X using Singular Vector Decomposition ($X \approx \hat{X} = U_K \Sigma_K V_K^\top$), such that the Frobe-

	financial	sport	institution	R_o	R_m	$R_o - R_m$	$R_o - 0.01R_m$
v_1	1	0	0	***20***	***600***	-580	14
v_2	0.2	0.3	0.2	5	100	***-95***	4
v_3	0.6	0	0.1	18	300	-282	***15***

Table 1: Three possible latent vectors hypotheses for the definition of $bank\#n\#1$

nius norm of difference between the two matrices is minimized:

$$\sqrt{\sum_i \sum_j \left(\hat{X}_{ij} - X_{ij}\right)^2} \tag{2}$$

In effect, LSA allows missing and observed words to equally impact the objective function. Given the inherent short length of the sentences, LSA (equation 2) allows for much more potential influence from missing words rather than observed words (99.9% cells are 0 in X). Hence the contribution of the observed words is significantly diminished. Moreover, the true semantics of the concept definitions is actually related to some missing words, but such true semantics will not be favored by the objective function, since equation 2 allows for too strong an impact by $\hat{X}_{ij} = 0$ for **any** missing word. Therefore the LSA model, in the context of short texts, is allowing missing words to have a significant "uncontrolled" impact on the model.

2.1 An Example

The three latent semantics profiles in table 1 illustrate our analysis for topic models and LSA. Assume there are three dimensions: financial, sports, institution. We use R_o^v to denote the sum of relatedness between latent vector v and all observed words; similarly, R_m^v is the sum of relatedness between the vector v and all missing words. The first latent vector (generated by topic models) is chosen by maximizing $R_{obs} = 600$. It suggests $bank\#n\#1$ is only related to the $financial$ dimension. The second latent vector (found by LSA) has the maximum value of $R_{obs} - R_{miss} = -95$, but obviously the latent vector is not related to $bank\#n\#1$ at all. This is because LSA treats observed words and missing words equally the same, and due to the large number of missing words, the information of observed words is lost: $R_{obs} - R_{miss} \approx -R_{miss}$. The third vector is the ideal semantics profile, since it is also related to the $institution$ dimension. It has a slightly smaller R_{obs} in comparison to the first vector, yet it has a substantially smaller R_{miss}.

In order to favor the ideal vector over other vectors, we simply need to adjust the objective function by assigning a smaller weight to R_{miss} such as: $R_{obs} - 0.01 \times R_{miss}$. Accordingly, we use weighted matrix factorization (Srebro and Jaakkola, 2003) to model missing words.

3 The Proposed Approach

3.1 Weighted Matrix Factorization

The weighted matrix factorization [WMF] approach is very similar to SVD, except that it allows for direct control on each matrix cell X_{ij}. The model factorizes the original matrix X into two matrices such that $X \approx P^\top Q$, where P is a $K \times M$ matrix, and Q is a $K \times N$ matrix (figure 1).

The model parameters (vectors in P and Q) are optimized by minimizing the objective function:

$$\sum_i \sum_j W_{ij} \left(P_{\cdot,i} \cdot Q_{\cdot,j} - X_{ij}\right)^2 + \lambda ||P||_2^2 + \lambda ||Q||_2^2 \tag{3}$$

where λ is a free regularization factor, and the weight matrix W defines a weight for each cell in X.

Accordingly, $P_{\cdot,i}$ is a K-dimension latent semantics vector profile for word w_i; similarly, $Q_{\cdot,j}$ is the K-dimension vector profile that represents the sentence s_j. Operations on these K-dimensional vectors have very intuitive semantic meanings:

(1) the inner product of $P_{\cdot,i}$ and $Q_{\cdot,j}$ is used to approximate semantic relatedness of word w_i and sentence s_j: $P_{\cdot,i} \cdot Q_{\cdot,j} \approx X_{ij}$, as the shaded parts in Figure 1;

(2) equation 3 explicitly requires a sentence should not be related to its missing words by forcing $P_{\cdot,i} \cdot Q_{\cdot,j} = 0$ for missing words $X_{ij} = 0$.

(3) we can compute the similarity of two sentences s_j and $s_{j'}$ using the cosine similarity between $Q_{\cdot,j}$, $Q_{\cdot,j'}$.

The latent vectors in P and Q are first randomly initialized, then can be computed iteratively by the following equations (derivation is omitted due to limited space, which can be found in (Srebro and Jaakkola, 2003)):

$$\begin{aligned} P_{\cdot,i} &= \left(Q\tilde{W}^{(i)}Q^\top + \lambda I\right)^{-1} Q\tilde{W}^{(i)}X_{i,\cdot}^\top \\ Q_{\cdot,j} &= \left(P\tilde{W}^{(j)}P^\top + \lambda I\right)^{-1} P\tilde{W}^{(i)}X_{\cdot,j} \end{aligned} \tag{4}$$

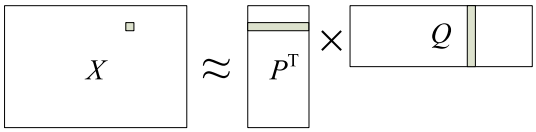

Figure 1: Matrix Factorization

where $\tilde{W}^{(i)} = \text{diag}(W_{\cdot,i})$ is an $M \times M$ diagonal matrix containing ith row of weight matrix W. Similarly, $\tilde{W}^{(j)} = \text{diag}(W_{\cdot,j})$ is an $N \times N$ diagonal matrix containing jth column of W.

3.2 Modeling Missing Words

It is straightforward to implement the idea in Section 2.1 (choosing a latent vector that maximizes $R_{obs} - 0.01 \times R_{miss}$) in the WMF framework, by assigning a small weight for all the missing words and minimizing equation 3:

$$W_{i,j} = \begin{cases} 1, & \text{if } X_{ij} \neq 0 \\ w_m, & \text{if } X_{ij} = 0 \end{cases} \quad (5)$$

We refer to our model as Weighted Textual Matrix Factorization [WTMF]. [1]

This solution is quite elegant: 1. it explicitly tells the model that in general all missing words should not be related to the sentence; 2. meanwhile latent semantics are mainly generalized based on observed words, and the model is not penalized too much (w_m is very small) when it is very confident that the sentence is highly related to a small subset of missing words based on their latent semantics profiles ($bank\#n\#1$ definition sentence is related to its missing words *check loan*).

We adopt the same approach (assigning a small weight for some cells in WMF) proposed for recommender systems [RS] (Steck, 2010). In RS, an incomplete rating matrix R is proposed, where rows are users and columns are items. Typically, a user rates only some of the items, hence, the RS system needs to predict the missing ratings. Steck (2010) guesses a value for all the missing cells, and sets a small weight for those cells.

Compared to (Steck, 2010), we are facing a different problem and targeting a different goal. We have a full matrix X where missing words have a 0 value, while the missing ratings in RS are unavailable – the values are unknown, hence R is not complete. In the RS setting, they are interested in predicting individual ratings, while we are interested in the sentence semantics. More importantly, they do not have the sparsity issue (each user has rated over 100 items in the movie lens data[2]) and robust predictions can be made based on the observed ratings alone.

4 Evaluation for SS

We need to show the impact of our proposed model WTMF on the SS task. However we are faced with a problem, the lack of a suitable large evaluation set from which we can derive robust observations. The two data sets we know of for SS are: 1. human-rated sentence pair similarity data set (Li et al., 2006) [LI06]; 2. the Microsoft Research Paraphrase Corpus (Dolan et al., 2004) [MSR04]. The LI06 data set consists of 65 pairs of noun definitions selected from the Collin Cobuild Dictionary. A subset of 30 pairs is further selected by LI06 to render the similarity scores evenly distributed. While this is the ideal data set for SS, the small size makes it impossible for tuning SS algorithms or deriving significant performance conclusions.

On the other hand, the MSR04 data set comprises a much larger set of sentence pairs: 4,076 training and 1,725 test pairs. The ratings on the pairs are binary labels: similar/not similar. This is not a problem per se, however the issue is that it is very strict in its assignment of a positive label, for example the following sentence pair as cited in (Islam and Inkpen, 2008) is rated not semantically similar:
Ballmer has been vocal in the past warning that Linux is a threat to Microsoft.
In the memo, Ballmer reiterated the open-source threat to Microsoft.

We believe that the ratings on a data set for SS should accommodate variable degrees of similarity with various ratings, however such a large scale set does not exist yet. Therefore for purposes of evaluating our proposed approach we devise a new framework inspired by the LI06 data set in that it comprises concept definitions but on a large scale.

4.1 Concept Definition Retrieval

We define a new framework for evaluating SS and project it as a Concept Definition Retrieval (CDR) task where the data points are dictionary definitions. The intuition is that two definitions in different dic-

[1] An efficient way to compute equation 4 is proposed in (Steck, 2010).

[2] http://www.grouplens.org/node/73, with 1M data set being the most widely used.

tionaries referring to the same concept should be assigned large similarity. In this setting, we design the CDR task in a search engine style. The SS algorithm has access to all the definitions in WordNet (WN). Given an OntoNotes (ON) definition (Hovy et al., 2006), the SS algorithm should rank the equivalent WN definition as high as possible based on sentence similarity.

The manual mapping already exists for ON to WN. One ON definition can be mapped to several WN definitions. After preprocessing we obtain 13669 ON definitions mapped to 19655 WN definitions. The data set has the advantage of being very large and it doesn't require further human scrutiny.

After the SS model learns the co-occurrence of words from WN definitions, in the testing phase, given an ON definition d, the SS algorithm needs to identify the equivalent WN definitions by computing the similarity values between all WN definitions and the ON definition d, then sorting the values in decreasing order. Clearly, it is very difficult to rank the one correct definition as highest out of all WN definitions (110,000 in total), hence we use ATOP_d, *area under the* $\text{TOPK}_d(k)$ *recall curve for an ON definition* d, to measure the performance. Basically, it is the ranking of the correct WN definition among all WN definitions. The higher a model is able to rank the correct WN definition, the better its performance.

Let N_d be the number of aligned WN definitions for the ON definition d, and N_d^k be the number of aligned WN definitions in the top-k list. Then with a normalized k $\in$ [0,1], $\text{TOPK}_d(k)$ and ATOP_d is defined as:

$$\begin{aligned} \text{TOPK}_d(k) &= N_d^k / N_d \\ \text{ATOP}_d &= \int_0^1 \text{TOPK}_d(k) \mathrm{d}k \end{aligned} \tag{6}$$

ATOP_d computes the normalized rank (in the range of $[0, 1]$) of aligned WN definitions among all WN definitions, with value 0.5 being the random case, and 1 being ranked as most similar.

5 Experiments and Results

We evaluate WTMF on three data sets: 1. CDR data set using ATOP metric; 2. Human-rated Sentence Similarity data set [LI06] using Pearson and Spearman Correlation; 3. MSR Paraphrase corpus [MSR04] using accuracy.

The performance of WTMF on CDR is compared with (a) an Information Retrieval model (IR) that is based on surface word matching, (b) an n-gram model (N-gram) that captures phrase overlaps by returning the number of overlapping ngrams as the similarity score of two sentences, (c) LSA that uses svds() function in Matlab, and (d) LDA that uses Gibbs Sampling for inference (Griffiths and Steyvers, 2004). WTMF is also compared with all existing reported SS results on LI06 and MSR04 data sets, as well as LDA that is trained on the same data as WTMF. The similarity of two sentences is computed by cosine similarity (except N-gram). More details on each task will be explained in the subsections.

To eliminate randomness in statistical models (WTMF and LDA), all the reported results are averaged over 10 runs. We run 20 iterations for WTMF. And we run 5000 iterations for LDA; each LDA model is averaged over the last 10 Gibbs Sampling iterations to get more robust predictions.

The latent vector of a sentence is computed by: (1) using equation 4 in WTMF, or (2) summing up the latent vectors of all the constituent words weighted by X_{ij} in LSA and LDA, similar to the work reported in (Mihalcea et al., 2006). For LDA the latent vector of a word is computed by $P(z|w)$. It is worth noting that we could directly use the estimated topic distribution θ_j to represent a sentence, however, as discussed the topic distribution has only non-zero values on one or two topics, leading to a low ATOP value around 0.8.

5.1 Corpus

The corpus we use comprises three dictionaries WN, ON, Wiktionary [Wik],[3] Brown corpus. For all dictionaries, we only keep the definitions without examples, and discard the mapping between sense ids and definitions. All definitions are simply treated as individual documents. We crawl Wik and remove the entries that are not tagged as noun, verb, adjective, or adverb, resulting in $220,000$ entries. For the Brown corpus, each sentence is treated as a document in order to create more coherent co-occurrence values. All data is tokenized, pos-tagged[4], and lem-

[3] http://en.wiktionary.org/wiki/Wiktionary:Main_Page

[4] http://nlp.stanford.edu/software/tagger.shtml

Models	Parameters	Dev	Test
1. IR	-	0.8578	0.8515
2. N-gram	-	0.8238	0.8171
3. LSA	-	0.8218	0.8143
4a. LDA	$\alpha = 0.1, \beta = 0.01$	0.9466 ± 0.0020	0.9427 ± 0.0006
4b. LDA	$\alpha = 0.05, \beta = 0.05$	0.9506 ± 0.0017	0.9470 ± 0.0005
5. WTMF	$w_m = 1, \lambda = 0$	0.8273 ± 0.0028	0.8273 ± 0.0014
6. WTMF	$w_m = 0, \lambda = 20$	0.8745 ± 0.0058	0.8645 ± 0.0031
7a. WTMF	$w_m = 0.01, \lambda = 20$	0.9555 ± 0.0015	0.9511 ± 0.0003
7b. WTMF	$w_m = 0.0005, \lambda = 20$	0.9610 ± 0.0011	0.9558 ± 0.0004

Table 2: ATOP Values of Models ($K = 100$ for LSA/LDA/WTMF)

matized[5]. The importance of words in a sentence is estimated by the TF-IDF schema.

All the latent variable models (LSA, LDA, WTMF) are built on the same set of corpus: WN+Wik+Brown ($393,666$ sentences and $4,262,026$ words). Words that appear only once are removed. The test data is never used during training phrase.

5.2 Concept Definition Retrieval

Among the 13669 ON definitions, 1000 definitions are randomly selected as a development set (dev) for picking best parameters in the models, and the rest is used as a test set (test). The performance of each model is evaluated by the average ATOP$_d$ value over the 12669 definitions (test). We use the subscript set in ATOP$_{set}$ to denote the average of ATOP$_d$ of a set of ON definitions, where $d \in \{set\}$. If all the words in an ON definition are not covered in the training data (WN+Wik+Br), then ATOP$_d$ for this instance is set to 0.5.

To compute ATOP$_d$ for an ON definition efficiently, we use the rank of the aligned WN definition among a random sample (size=1000) of WN definitions, to approximate its rank among all WN definitions. In practice, the difference between using 1000 samples and all data is tiny for ATOP$_{test}$ (± 0.0001), due to the large number of data points in CDR.

We mainly compare the performance of IR, N-gram, LSA, LDA, and WTMF models. Generally results are reported based on the last iteration. However, we observe that for model 6 in table 2, the best performance occurs at the first few iterations. Hence for that model we use the ATOP$_{dev}$ to indicate when to stop.

[5]http://wn-similarity.sourceforge.net, WordNet::QueryData

5.2.1 Results

Table 2 summarizes the ATOP values on the dev and test sets. All parameters are tuned based on the dev set. In LDA, we choose an optimal combination of α and β from $\{0.01, 0.05, 0.1, 0.5\}$.In WTMF, we choose the best parameters of weight w_m for missing words and λ for regularization. We fix the dimension $K = 100$. Later in section 5.2.2, we will see that a larger value of K can further improve the performance.

WTMF that models missing words using a small weight (model 7b with $w_m = 0.0005$) outperforms the second best model LDA by a large margin. This is because LDA only uses 10 observed words to infer a 100 dimension vector for a sentence, while WTMF takes advantage of much more missing words to learn more robust latent semantics vectors.

The IR model that works in word space achieves better ATOP scores than N-gram, although the idea of N-gram is commonly used in detecting paraphrases as well as machine translation. Applying TF-IDF for N-gram is better, but still the ATOP$_{test}$ is not higher: 0.8467. The reason is words are enough to capture semantics for SS, while n-grams/phrases are used for a more fine-grained level of semantics.

We also present model 5 and 6 (both are WTMF), to show the impact of: 1. modeling missing words with equal weights as observed words ($w_m = 1$) (LSA manner), and 2. not modeling missing words at all ($w_m = 0$) (LDA manner) in the context of WTMF model. As expected, both model 5 and model 6 generate much worse results.

Both LDA and model 6 ignore missing words, with better ATOP$_{test}$ scores achieved by LDA. This may be due to the different inference algorithms. Model 5 and LSA are comparable, where missing words are used with a large weight. Both of them yield low results. This confirms our assumption

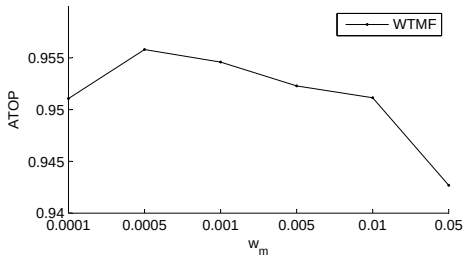

Figure 2: missing words weight w_m in WTMF

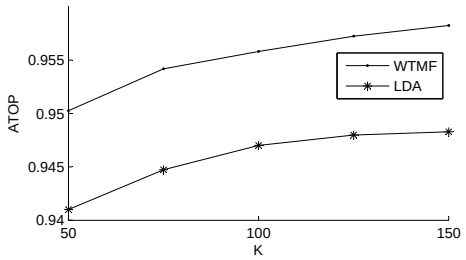

Figure 3: dimension K in WTMF and LDA

that allowing for equal impact of both observed and missing words is not the correct characterization of the semantic space.

5.2.2 Analysis

In these latent variable models, there are several essential parameters: weight of missing words w_m, and dimension K. Figure 2 and 3 analyze the impact of these parameters on ATOP$_{test}$.

Figure 2 shows the influence of w_m on ATOP$_{test}$ values. The peak ATOP$_{test}$ is around $w_m = 0.0005$, while other values of w_m (except $w_m = 0.05$) also yield high ATOP values (better than LDA).

We also measure the influence of the dimension $K = \{50, 75, 100, 125, 150\}$ on LDA and WTMF in Figure 3, where parameters for WTMF are $w_m = 0.0005, \lambda = 20$, and for LDA are $\alpha = 0.05, \beta = 0.05$. We can see WTMF consistently outperforms LDA by an ATOP value of 0.01 in each dimension. Although a larger K yields a better result, we still use a 100 due to computational complexity.

5.3 LI06: Human-rated Sentence Similarity

We also assess WTMF and LDA model on LI06 data set. We still use $K = 100$. As we can see in Figure 2, choosing the appropriate parameter w_m could boost the performance significantly. Since we do not have any tuning data for this task, we present Pearson's correlation r for different values of w_m in Table 3. In addition, to demonstrate that w_m does not overfit the 30 data points, we also evaluate on

	30 pairs		**35 pairs**	
$\mathbf{w_m}$	r	ρ	r	ρ
0.0005	0.8247	0.8440	0.4200	0.6006
0.001	0.8470	0.8636	0.4308	0.5985
0.005	0.8876	0.8966	**0.4638**	**0.5809**
0.01	**0.8984**	**0.9091**	0.4564	0.5450
0.05	0.8804	0.8812	0.4087	0.4766

Table 3: Different w_m of WTMF on LI06 ($K = 100$)

the other 35 pairs in LI06. Same as in (Tsatsaronis et al., 2010), we also include Spearman's rank order correlation ρ, which is correlation of ranks of similarity values . Note that r and ρ are much lower for 35 pairs set, since most of the sentence pairs have a very low similarity (the average similarity value is 0.065 in 35 pairs set and 0.367 in 30 pairs set) and SS models need to identify the tiny difference among them, thereby rendering this set much harder to predict.

Using $w_m = 0.01$ gives the best results on 30 pairs while on 35 pairs the peak values of r and ρ happens when $w_m = 0.005$. In general, the correlations in 30 pairs and in 35 pairs are consistent, which indicates $w_m = 0.01$ or $w_m = 0.005$ does not overfit the 30 pairs set.

Compared to CDR, LI06 data set has a strong preference for a larger w_m. This could be caused by different goals of the two tasks: CDR is evaluated by the rank of the most similar ones among all candidates, while the LI06 data set treats similar pairs and dissimilar pairs as equally important. Using a smaller w_m means the similarity score is computed mainly from semantics of the observed words. This benefits CDR, since it gives more accurate similarity scores for those similar pairs, but not so accurate for dissimilar pairs. In fact, from Figure 2 and Table 2 we see that $w_m = 0.01$ also produces a very high ATOP$_{test}$ value in CDR.

Table 4 shows the results of all current SS models with respect to the LI06 data set (30 pairs set). We cite their best performance for all reported results.

Once the correct $w_m = 0.01$ is chosen, WTMF results in the best Pearson's r and best Spearman's ρ ($w_m = 0.005$ yields the second best r and ρ). Same as in CDR task, WTMF outperforms LDA by a large margin in both r and ρ. It indicates that the latent vectors induced by WTMF are able to not only identify same/similar sentences, but also identify the "correct" degree of dissimilar sentences.

Model	r	ρ
STASIS (Li et al., 2006)	0.8162	0.8126
(Liu et al., 2007)	0.841	0.8538
(Feng et al., 2008)	0.756	0.608
STS (Islam and Inkpen, 2008)	0.853	0.838
LSA (O'Shea et al., 2008)	0.8384	0.8714
Omiotis (Tsatsaronis et al., 2010)	0.856	0.8905
WSD-STS (Ho et al., 2010)	0.864	0.8341
SPD-STS (Ho et al., 2010)	0.895	0.9034
LDA ($\alpha = 0.05, \beta = 0.05$)	0.8422	0.8663
WTMF ($w_m = 0.005, \lambda = 20$)	0.8876	0.8966
WTMF ($w_m = 0.01, \lambda = 20$)	**0.8984**	**0.9091**

Table 4: Pearson's correlation r and Spearman's correlation ρ on LI06 30 pairs

Model	Accuracy
Random	51.3
LSA (Mihalcea et al., 2006)	68.4
full model (Mihalcea et al., 2006)	70.3
STS (Islam and Inkpen, 2008)	72.6
Omiotis (Tsatsaronis et al., 2010)	69.97
LDA ($\alpha = 0.05, \beta = 0.05$)	68.6
WTMF ($w_m = 0.01, \lambda = 20$)	71.51

Table 5: Performance on MSR04 test set

5.4 MSR04: MSR Paraphrase Corpus

Finally, we briefly discuss results of applying WTMF on MSR04 data. We use the same parameter setting used for the LI06 evaluation setting since both sets are human-rated sentence pairs ($\lambda = 20, w_m = 0.01, K = 100$). We use the training set of MSR04 data to select a threshold of sentence similarity for the binary label. Table 5 summarizes the accuracy of other SS models noted in the literature and evaluated on MSR04 test set.

Compared to previous SS work and LDA, WTMF has the second best accuracy. It suggests that WTMF is quite competitive in the paraphrase recognition task.

It is worth noting that the best system on MSR04, STS (Islam and Inkpen, 2008), has much lower correlations on LI06 data set. The second best system among previous work on LI06 uses Spearman correlation, Omiotis (Tsatsaronis et al., 2010), and it yields a much worse accuracy on MSR04. The other works do not evaluate on both data sets.

6 Related Work

Almost all current SS methods work in the high-dimensional word space, and rely heavily on word/sense similarity measures, which is knowledge based (Li et al., 2006; Feng et al., 2008; Ho et al., 2010; Tsatsaronis et al., 2010), corpus-based (Islam and Inkpen, 2008) or hybrid (Mihalcea et al., 2006). Almost all of them are evaluated on LI06 data set. It is interesting to see that most works find word similarity measures, especially knowledge based ones, to be the most effective component, while other features do not work well (such as word order or syntactic information). Mihalcea et al. (2006) use LSA as a baseline, and O'Shea et al. (2008) train LSA on regular length documents. Both results are considerably lower than word similarity based methods. Hence, our work is the first to successfully approach SS in the latent space.

Although there has been work modeling latent semantics for short texts (tweets) in LDA, the focus has been on exploiting additional features in Twitter, hence restricted to Twitter data. Ramage et al. (2010) use tweet metadata (author, hashtag) as some supervised information to model tweets. Jin et al. (2011) use long similar documents (the article that is referred by a url in tweets) to help understand the tweet. In contrast, our approach relies solely on the information in the texts by modeling local missing words, and does not need any additional data, which renders our approach much more widely applicable.

7 Conclusions

We explicitly model missing words to alleviate the sparsity problem in modeling short texts. We also propose a new evaluation framework for sentence similarity that allows large scale tuning and testing. Experiment results on three data sets show that our model WTMF significantly outperforms existing methods. For future work, we would like to compare the text modeling performance of WTMF with LSA and LDA on regular length documents.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

This research was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the U.S. Army Research Lab. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI or the U.S. Government.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3.

William Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics*.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Jin Feng, Yi-Ming Zhou, and Trevor Martin. 2008. Sentence similarity based on relevance. In *Proceedings of IPMU*.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101.

Chukfong Ho, Masrah Azrifah Azmi Murad, Rabiah Abdul Kadir, and Shyamala C. Doraisamy. 2010. Word sense disambiguation-based sentence similarity. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*.

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 North American Chapter of the ACL*.

Aminul Islam and Diana Inkpen. 2008. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data*, 2.

Ou Jin, Nathan N. Liu, Kai Zhao, Yong Yu, and Qiang Yang. 2011. Transferring topical knowledge from auxiliary long texts for short text clustering. In *Proceedings of the 20th ACM international conference on Information and knowledge management*.

David Kauchak and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*.

Thomas K Landauer, Peter W. Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes*, 25.

Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*.

Yuhua Li, Davi d McLean, Zuhair A. Bandar, James D. O Shea, and Keeley Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transaction on Knowledge and Data Engineering*, 18.

Xiao-Ying Liu, Yi-Ming Zhou, and Ruo-Shi Zheng. 2007. Sentence similarity based on dynamic time warping. In *The International Conference on Semantic Computing*.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Articial Intelligence*.

James O'Shea, Zuhair Bandar, Keeley Crockett, and David McLean. 2008. A comparative study of two short text semantic similarity measures. In *Proceedings of the Agent and Multi-Agent Systems: Technologies and Applications, Second KES International Symposium (KES-AMSTA)*.

Daniel Ramage, Susan Dumais, and Dan Liebling. 2010. Characterizing microblogs with topic models. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*.

Nathan Srebro and Tommi Jaakkola. 2003. Weighted low-rank approximations. In *Proceedings of the Twentieth International Conference on Machine Learning*.

Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

George Tsatsaronis, Iraklis Varlamis, and Michalis Vazirgiannis. 2010. Text relatedness based on a word thesaurus. *Journal of Articial Intelligence Research*, 37.

Liang Zhou, Chin-Yew Lin, Dragos Stefan Munteanu, and Eduard Hovy. 2006. Paraeval: Using paraphrases to evaluate summaries automatically. In *Proceedings of Human Language Tech-nology Conference of the North American Chapter of the ACL,*.