Encoding Distributional Semantics into Triple-Based Knowledge Ranking for Document Enrichment

Muyu Zhang[1]*, **Bing Qin**[1], **Mao Zheng**[1], **Graeme Hirst**[2], **and Ting Liu**[1]
[1]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, Harbin, China
[2]Department of Computer Science, University of Toronto, Toronto, ON, Canada
{myzhang,qinb,mzheng,tliu}@ir.hit.edu.cn
gh@cs.toronto.edu

Abstract

Document enrichment focuses on retrieving relevant knowledge from external resources, which is essential because text is generally replete with gaps. Since conventional work primarily relies on special resources, we instead use triples of *Subject, Predicate, Object* as knowledge and incorporate distributional semantics to rank them. Our model first extracts these triples automatically from raw text and converts them into real-valued vectors based on the word semantics captured by Latent Dirichlet Allocation. We then represent these triples, together with the source document that is to be enriched, as a graph of triples, and adopt a global iterative algorithm to propagate relevance weight from source document to these triples so as to select the most relevant ones. Evaluated as a ranking problem, our model significantly outperforms multiple strong baselines. Moreover, we conduct a task-based evaluation by incorporating these triples as additional features into document classification and enhances the performance by 3.02%.

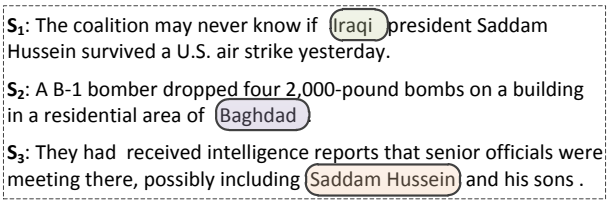
1 Introduction

Document enrichment is the task of acquiring relevant background knowledge from external resources for a given document. This task is essential because, during the writing of text, some basic but well-known information is usually omitted by the author to make the document more concise. For example, *Baghdad is the capital of Iraq* is omitted in Figure 1a. A human will fill these gaps automatically with the background knowledge in his mind. However, the machine lacks both the necessary background knowledge and the ability to select. The task of document enrichment is proposed to tackle this problem, and has been proved helpful in many NLP tasks such as web search (Pantel and Fuxman, 2011), coreference resolution (Bryl et al., 2010), document cluster (Hu et al., 2009) and entity disambiguation (Sen, 2012).

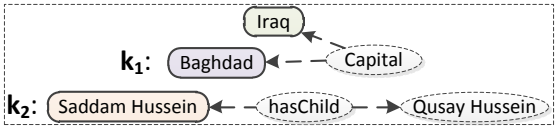
We can classify previous work into two classes according to the resources they rely on. The first line of work uses *Wikipedia*, the largest on-line encyclopedia, as a resource and introduces the content of Wikipedia pages as external knowledge (Cucerzan, 2007; Kataria et al., 2011; He et al., 2013). Most research in this area relies on the text similarity (Zheng et al., 2010; Hoffart et al., 2011) and structure information (Kulkarni et al., 2009; Sen, 2012; He et al., 2013) between the mention and the Wikipedia page. Despite the apparent success of these methods, most Wikipedia pages contain too much information, most of which is not relevant enough to the source document, and this causes a noise problem. Another line of work tries to improve the accuracy by introducing ontologies (Fodeh et al., 2011; Kumar and Salim, 2012) and structured knowledge bases such as WordNet (Nastase et al., 2010), which provide semantic information about words such as synonym (Sun et al., 2011) and antonym (Sansonnet and Bouchet, 2010). However, these methods primarily rely on special resources constructed with supervision or even manually, which are difficult to expand and in turn limit their applications in practice.

In contrast, we wish to seek the benefits of both coverage and accuracy from a better representation of background knowledge: triples of *Subject, Predicate, Object* (SPO). According to Hoffart et al. (2013), these triples, such as *LeonardCohen, wasBornIn, Montreal*, can be extracted automatically from Wikipedia and other sources, which is compatible with the RDF data model (Staab and Studer, 2009). Moreover, by extracting these

* This work was partly done while the first author was visiting University of Toronto.



(a) Source document: air strike aiming at Saddam in Baghdad



(b) Two omitted relevant pieces of background knowledge

Figure 1: An example of document enrichment: A source document about a U.S. air strike omitting two important pieces of background knowledge which are acquired by our framework.

triples from multiple sources, we also get better coverage. Therefore, one can expect that this representation is helpful for better document enrichment by incorporating both accuracy and coverage. In fact, there is already evidence that this representation is helpful. Zhang et al. (2014) proposed a triple-based document enrichment framework which uses triples of SPO as background knowledge. They first proposed a search engine–based method to evaluate the relatedness between every pair of triples, and then an iterative propagation algorithm was introduced to select the most relevant triples to a given source document (see Section 2), which achieved a good performance.

However, to evaluate the semantic relatedness between two triples, Zhang et al. (2014) primarily relied on the text of triples and used search engines, which makes their method difficult to re-implement and in turn limits its application in practice. Moreover, they did not carry out any task-based evaluation, which makes it uncertain whether their method will be helpful in real applications. Therefore, we instead use topic models, especially *Latent Dirichlet Allocation* (LDA), to encode distributional semantics of words and convert every triple into a real-valued vector, which is then used to evaluate the relatedness between a pair of triples. We then incorporate these triples into the given source document and represent them together as a graph of triples. Then a modified iterative propagation is carried out over the entire graph to select the most relevant triples of background knowledge to the given source document.

To evaluate our model, we conduct two series of experiments: (1) evaluation as a ranking problem, and (2) task-based evaluation. We first treat this task as a ranking problem which inputs one document and outputs the top N most-relevant triples of background knowledge. Second, we carry out a task-based evaluation by incorporating these relevant triples acquired by our model into the original model of document classification as additional features. We then perform a direct comparison between the classification models with and without these triples, to determine whether they are helpful or not. On the first series of experiments, we achieve a *MAP* of 0.6494 and a *P@N* of 0.5597 in the best situation, which outperforms the strongest baseline by 5.87% and 17.21%. In the task-based evaluation, the enriched model derived from the triples of background knowledge performs better by 3.02%, which demonstrates the effectiveness of our framework in real NLP applications.

2 Background

The most closely related work in this area is our own (Zhang et al., 2014), which used the triples of *SPO* as background knowledge. In that work, we first proposed a *triple graph* to represent the source document and then used a search engine–based iterative algorithm to rank all the triples. We describe this work in detail below.

Triple graph Zhang et al. (2014) proposed the *triple graph* as a document representation, where the triples of *SPO* serve as nodes, and the edges between nodes indicate their semantic relatedness. There are two kinds of nodes in the triple graph: (1) source document nodes (*sd-nodes*), which are triples extracted from source documents, and (2) background knowledge nodes (*bk-nodes*), which are triples extracted from external sources. Both of them are extracted automatically with *Reverb*, a well-known *Open Information Extraction* system (Etzioni et al., 2011). There are also two kinds of edges: (1) an edge between a pair of sd-nodes, and (2) an edge between one sd-node and another bk-node, both of which are unidirectional. In the original representation, there are no edges between two bk-nodes because they treat the bk-nodes as recipients of relevance weight only. In this paper, we modify this setup and connect every pair of bk-nodes with an edge, so the bk-nodes serve as intermediate nodes during the iterative propagation process and contribute to the final performance too as shown in our experiments (see Section 5.1).

Relevance evaluation To compute the weight of a edge, Zhang et al. (2014) evaluate the semantic relatedness between two nodes with a search engine–based method. They first convert every node, which is a triple of *SPO*, into a query by combining the text of *Subject* and *Object* together. Then for every pair of nodes t_i and t_j, they construct three queries: p, q, and $p \cap q$, which correspond to the queries of t_i, t_j, and $t_j \cap t_j$, the combination of t_i and t_j. All these queries are put into a search engine to get $H(p)$, $H(q)$, and $H(p \cap q)$, the numbers of returned pages for query p, p, and $p \cap q$. Then the *WebJaccard Coefficient* (Bollegala et al., 2007) is used to evaluate $r(i,j)$, the relatedness between t_i and t_j, according to Formula 1.

$$r(i,j) = WebJaccard(p,q) = \begin{cases} 0 & \text{if } H(p \cap q) \leq C \\ \dfrac{H(p \cap q)}{H(p) + H(q) - H(p \cap q)} & \text{otherwise.} \end{cases} \quad (1)$$

Using $r(i,j)$, Zhang et al. (2014) further define $p(i,j)$, the probability of t_i and t_j propagating to each other, as shown in Formula 2. Here N is the set of all nodes, and $\delta(i,j)$ denotes whether an edge exists between two nodes or not.

$$p(i,j) = \frac{r(i,j) \times \delta(i,j)}{\sum_{n \in N} r(n,j) \times \delta(n,j)} \quad (2)$$

Iterative propagation Considering that the source document D is represented as a graph of sd-nodes, so the relevance of background knowledge t_b to D is naturally converted into that of t_b to the graph of sd-nodes. Zhang et al. (2014) evaluate this relevance by propagating relevance weight from sd-nodes to t_b iteratively. After convergence, the relevance weight of t_b will be treated as the final relevance to D. There are in total $n \times n$ pairs of nodes, and their $p(i,j)$ are stored in a matrix P. Zhang et al. (2014) use $\vec{W} = (w_1, w_2, \ldots, w_n)$ to denote the relevance weights of nodes, where w_i indicates the relevance of t_i to D. At the beginning, each w_i of bk-nodes is initialized to 0, and each that of sd-nodes is initialized to its importance to D. Then $\vec{W}$ is updated to $\vec{W}'$ after every iteration according to Formula 3. They keep updating the weights of both sd-nodes and bk-nodes until convergence and do not distinguish them explicitly.

$$\begin{aligned} \vec{W}' &= \vec{W} \times P \\ &= \vec{W} \times \begin{bmatrix} p(1,1) & p(1,2) & \ldots & p(1,n) \\ p(2,1) & p(2,2) & \ldots & p(2,n) \\ \ldots & \ldots & \ldots & \ldots \\ p(n,1) & p(n,2) & \ldots & p(n,n) \end{bmatrix} \end{aligned} \quad (3)$$

3 Methodology

The key idea behind this work is that every document is composed of several units of information, which can be extracted into triples automatically. For every unit of background knowledge b, the more units that are relevant to b and the more relevant they are, the more relevant b will be to the source document. Based on this intuition, we first present both source document information and background knowledge together as a document-level triple graph as illustrated in Section 2. Then we use LDA to capture the distributional semantics of a triple by representing it as a vector of distributional probabilities over k topics and evaluate the relatedness between two triples with cosine-similarity. Finally, we propose a modified iterative process to propagate the relevance score from the source document information to the background knowledge and select the top n relevant ones.

3.1 Encoding distributional semantics

LDA LDA is a popular generative probabilistic model, which was first introduced by Blei et al. (2003). LDA views every document as a mixture over underlying topics, and each topic as a distribution over words. Both the document-topic and the topic-word distributions are assumed to have a *Dirichlet prior*. Given a set of documents and a number of topics, the model returns θ_d, the topic distribution for each document d, and ϕ_z, the word distribution for every topic z.

LDA assumes the following generative process for each document in a corpus D:

1. Choose $N \sim Poisson(\xi)$.
2. Choose $\theta \sim Dir(\alpha)$.
   (a) Choose a topic $z_n \sim Multinomial(\theta)$.
   (b) Choose a word w_n from $p(w_n|z_n,\beta)$ conditioned on the topic z_n.

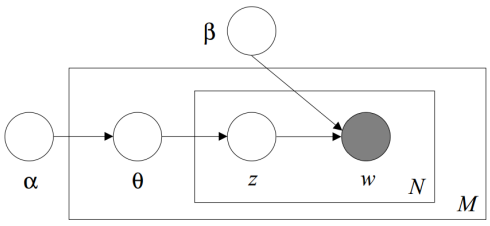


Figure 2: Graphical representation of LDA. The boxes represents replicates, where the inner box represents the repeated choice of N topics and words within a document, while the outer one represents the repeated generation of M documents.

Here the dimensionality k of the Dirichlet distribution (and thus the dimensionality of the topic variable z) is assumed to be known and fixed; θ is a k-dimensional Dirichlet random variable, where the parameter α is a k-vector with components $\alpha_i > 0$; and the β indicates the word probabilities over topics, which is a matrix with $\beta_{ij} = p(w^j = 1|z^i = 1)$. Figure 2 shows the representation of LDA as a probabilistic graphical model with three levels. There are two corpus-level parameters α and β , which are assumed to be sampled once in the process of generating a corpus; one document-level variable θ_d, which is sampled once per document; and two word-level variables z_{dn} and w_{dn}, which are sampled once for each word in each document.

We employ the publicly available implementation of LDA, JGibbLDA2[1] (Phan et al., 2008), which has two main execution methods: parameter estimation (model building) and inference for new data (classification of a new document).

Relevance evaluation Given a set of documents and the number of topics k, LDA will return ϕ_z, the word distribution over the topic z. So for every word w_n, we get k distributional probabilities over k topics. We use $p_{w_n z_i}$ to denote the probability that w_n appears in the i^{th} topic z_i, where $i \leq k$, $z_i \in Z$, the set of k topics. Then we combine these k possibilities together as a real-valued vector $\vec{v}_{w_n}$ to represent w_n as shown in Formula 4.

$$\vec{v}_{w_n} = (p_{w_n z_1}, p_{w_n z_2}, \ldots, p_{w_n z_k}) \tag{4}$$

After getting the vectors of words, we employ an intuitive method to compute the vector of a triple t, by accumulating all the corresponding vectors of words appearing in t according to Formula 5. Considering that the elements of this newly generated vector indicate the distributional probabilities of t over k topics, we then normalize it according to Formula 6 so that its elements sum to 1. This gives us $\vec{v}_t$, the real-valued vector of triple t, which captures its distributional probabilities over k topics. Here t corresponds to a triple of background knowledge or of source document, p_{tz_i} indicates the possibility of t to appear in the i^{th} topic z_i, and $w_n \in t$ means that w_n appears in t.

$$p_{tz_i} = \sum_{w_n \in t} p_{w_n z_i} \tag{5}$$

$$\vec{v}_t = \frac{(p_{tz_1}, p_{tz_2}, \ldots, p_{tz_k})}{\sum_{i=1}^{k} p_{tz_i}} \tag{6}$$

Using the vectors of triples, we can easily compute the semantic relatedness between a pair of triples as their cosine-similarity according to Formula 7. Here A, B correspond to the real-valued vectors of two triples, $r(A,B)$ denotes their semantic relatedness, and k is the number of topics, which is also the length of A (or B). A high value of $r(A,B)$ usually indicates a close relatedness between A and B, and thus a higher probability of propagating to each other in the following modified iterative propagation illustrated in Section 3.2.

$$\begin{aligned} r(A,B) &= \cos(A,B) = \frac{AB}{\|A\|\|B\|} \\ &= \frac{\sum_{i=1}^{k} A_i B_i}{\sqrt{\sum_{i=1}^{k} (A_i)^2}\sqrt{\sum_{i=1}^{k} (B_i)^2}} \end{aligned} \tag{7}$$

3.2 Modified iterative propagation

In this part, we propose a modified iterative propagation based ranking model to select the most-relevant triples of background knowledge. There are three primary modifications to the original model of Zhang et al. (2014), all of which are shown more powerful in our experiments.

First of all, the original model (Zhang et al., 2014) does not reset the relevance weight of sd-nodes after every iteration. This results in a continued decrease of the relevance weight of sd-nodes, which weakens the effect of sd-nodes during the iterative propagation and in turn affects the final performance. To tackle this problem, we decrease the relevance weight of bk-nodes and increase that of sd-nodes according to a fixed ratio after every iteration, so as to ensure that the total weight of sd-nodes is always higher than that of bk-nodes. Note that although the relevance

[1] http://jgibblda.sourceforge.net/

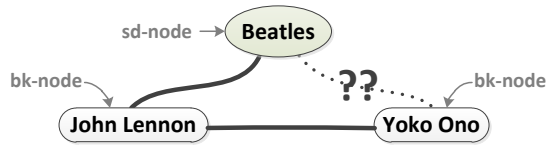


Figure 3: The edge between two bk-nodes helps in the better evaluation of relatedness between the bk-node *Yoko Ono* and the sd-node *Beatles*.

weights of bk-nodes are changed after the redistribution, the corresponding ranking of them is not changed because the redistribution is carried out over all nodes accordingly. In our experiments, we tried different ratios and finally chose 10:1, with sd-nodes corresponding to 10 and bk-nodes to 1, which achieved the best performance.

In addition, we also modify the triple graph, the representation of a document illustrated in Section 2, by connecting every pair of bk-nodes with an edge, which is not allowed in the original model. This modification was motivated by the intuition that the relatedness between bk-nodes also contributes to the better evaluation of relevance to the source document, because the bk-nodes can serve as the intermediate nodes during the iterative propagation over the entire graph. Figure 3 shows an example, where the bk-node *John Lennon* is close to both the sd-node *Beatles* and to another bk-node *Yoko Ono*, so the relatedness between two bk-nodes *John Lennon* and *Yoko Ono* helps in better evaluation of the relatedness between the bk-node *Yoko Ono* and the sd-node *Beatles*.

We also modify the definition of $p(i,j)$, the probability of two nodes t_i and t_j propagating to each other. Zhang et al. (2014) compute this probability according to Formula 2, which highlights the number of neighbors, but weakens the relatedness between nodes, due to the normalization. For instance, if a node t_x has only one neighbor t_y, no matter how low their relatedness is, their $p(x,y)$ will still be equal to 1 in the original model, while another node with two equally but closely related neighbors will only get a probability of 0.5 for each neighbor. We modify this setup by removing the normalization process and computing $p(i,j)$ as the relatedness between t_i and t_j directly, which is evaluated according to Formula 1 .

4 Encoding background knowledge into document classification

In this part, we demonstrate that the introduction of relevant knowledge could be helpful to real NLP applications. In particular, we choose the *document classification* task as a demonstration, which aims to classify documents into predefined categories automatically (Sebastiani, 2002). We choose this task for two reasons: (1) This task has witnessed a booming interest in the last 20 years, due to the increased availability of documents in digital form and the ensuing need to organize them, so it is important in both research and application. (2) The state-of-the-art performance of this task is achieved by a series of topic model–based methods, which rely on the same model as we do, but make use of source document information only. However, there is always some omitted information and relevant knowledge, which cannot be captured from the source document. Intuitively, the recovery of this information will be helpful. If we can improve the performance by introducing extra background knowledge into existing framework of document classification, we can inference naturally that the improvement benefits from the introduction of this knowledge.

Traditional methods primarily use topic models to represent a document as a topic vector. Then a *SVM* classifier takes this vector as input and outputs the class of the document. In this work, we propose a new framework for document classification to incorporate extra knowledge. Given a document to be classified, we select the top N most-relevant triples of background knowledge with our model introduced in Section 3, all of which are represented as vectors of $\vec{v}_t = (p_{tz_1}, p_{tz_2}, \ldots, p_{tz_k})$. Then we combine these N triples as a new vector $\vec{v}_t'$, which is then incorporated into the original framework of document classification. Another *SVM* classifier takes $\vec{v}_t'$, together with the original features extracted from the source document, as input and outputs the category of the source document. To combine N triples as one, we employ an intuitive method by computing the average of N corresponding vectors in every dimension.

One possible problem is how to decide N, the number of triples to be introduced. We first introduce a fixed amount of triples for every document. Moreover, we also select the triples according to their relevance weight to the source document (see Section 3.2) by setting a threshold of relevance

weight first and selecting the triples whose weights are higher than the threshold. We further discuss the impact of different thresholds in Section 5.2.

5 Experiments

To evaluate our model, we conduct two series of experiments: (1) We first treat this task as a ranking problem, which takes a document as input and outputs the ranked triples of background knowledge, and evaluate the ranking performance by computing the scores of *MAP* and *P@N*. (2) We also conduct a task-based evaluation, where document classification (see Section 4) is chosen as a demonstration, by enriching the background knowledge to the original framework as additional features and performing a direct comparison.

5.1 Evaluation as a ranking problem

Data preparation The data is composed of two parts: source documents and background knowledge. For source documents, we use a publicly available Chinese corpus which consists of 17,199 documents and 13,719,428 tokens extracted from Internet news[2] including 9 topics: *Finance, IT, Health, Sports, Travel, Education, Jobs, Art, Military*. We then randomly but equally select 600 articles as the set of source documents from 9 topics without data bias. We use all the other 16,599 documents of the same corpus as the source of background knowledge, and then introduce a well-known Chinese open source tool (Che et al., 2010) to extract the triples of background knowledge from the raw text automatically. So the background knowledge also distributes evenly across the same 9 topics. We use the same tool to extract the triples of source documents too.

Baseline systems As Zhang et al. (2014) argued, it is difficult to use the methods in traditional ranking tasks, such as information retrieval (Manning et al., 2008) and entity linking (Han et al., 2011; Sen, 2012), as baselines in this task, because our model takes triples as basic input and thus lacks some crucial information such as link structure. For better comparison, we implement three methods as baselines, which have been proved effective in relevance evaluation: (1) *Vector Space Model* (VSM), (2) *Word Embedding* (WE), and (3) *Latent Dirichlet Allocation* (LDA). Note that our model captures the distributional semantics of triples with LDA, while WE serves as a baseline only, where the word embeddings are acquired over the same corpus mentioned previously with the publicly available tool *word2vec*[3].

Here we use t_i, D, and w_i to denote a triple of background knowledge, a source document, and the relevance of t_i to D. For *VSM*, we represent both t_i and D with a *tf-idf* scheme first (Salton and McGill, 1986) and compute w_i as their cosine-similarity. For *WE*, we first convert both t_i and the triples extracted from D into real-valued vectors with WE and then compute w_i by accumulating all the cosine-similarities between t_i and every triple from D. For *LDA*, we represent t_i as a vector with our model introduced in Section 3.1 and get the vector of D directly with LDA. Then we evaluate their relevance of t_i to D by computing the cosine-similarity of two corresponding vectors.

Moreover, to determine whether our modified iterative propagation is helpful or not, we also compare our full model (*Ours*) against a simplified version without *iterative propagation* (*Ours-S*). In *Ours-S*, we represent both t_i and the triples extracted from D as real-valued vectors with our model introduced in Section 3.1. Then we compute w_i by accumulating all the cosine-similarities between t_i and the triples extracted from D. For all the baselines, we rank the triples of background knowledge according to w_i, their relevance to D.

Experimental setup Previous research relies on manual annotation to evaluate the ranking performance (Zhang et al., 2014), which costs a lot, and in which it is difficult to get high consistency. In this paper, we carry out an automatic evaluation. The corpus we used consists of 9 different classes, from which we extract triples of background knowledge. So correspondingly, there will be 9 sets of triples too. Then we randomly select 200 triples from every class and mix $200 \times 9 = 1800$ triples together as S, the set of triples of background knowledge. For every document D to be enriched, our model selects the top N most-relevant triples from S and returns them to D as enrichments. We treat a triple t_i selected by our model as positive only if t_i is extracted from the same class as D. We evaluate the performance of our model with two well-known criteria in ranking problem: *MAP* and *P@N* (Voorhees et al., 2005). Statistically significant differences of performance are determined using the two-tailed paired t-test

[2] http://www.sogou.com/labs/dl/c.html

[3] https://code.google.com/p/word2vec/

Model	MAP_5	P@5	MAP_10	P@10
VSM	0.4968	0.3435	0.4752	0.3841
WE	0.4356	0.3354	0.4624	0.3841
LDA	0.6134	0.4775	0.6071	0.5295
Ours-S	0.5325	0.3762	0.5012	0.4054
Ours	**0.6494**	**0.5597**	**0.6338**	**0.5502**

Table 1: The performance evaluated as a ranking task. Here *Ours* corresponds to our full model, while *Ours-S* is a simplified version of our model without *iterative propagation* (see Section 3.2).

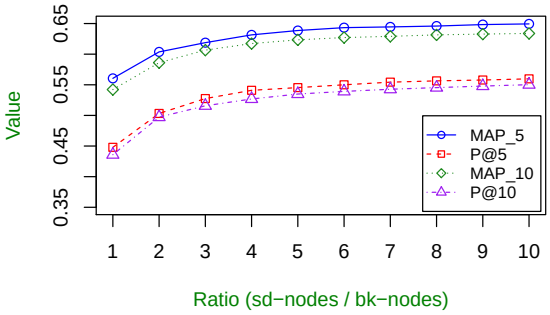


Figure 4: The performance of our model with different ratios between sd-nodes and bk-nodes.

computed at a 95% confidence level based on the average performance per source document.

Results The performance of multiple models is shown in Table 1. Overall, our full model *Ours* outperforms all the baseline systems significantly in every metric. When evaluating the top 10 triples with the highest relevance weight, our framework outperforms the best baseline *LDA* by 4.4% in *MAP* and by 3.91% in *P@N*. When evaluating the top 5 triples, our framework performs even better and significantly outperforms the best baseline by 5.87% in *MAP* and by 17.21% in *P@N*.

To analyze the results further, *Ours-S*, the simplified version of our model without *iterative propagation*, outperforms two strong baselines *VSM* and *WE*, which indicates the effectiveness of encoding distributional semantics. However, the performance of this simplified model is not as good as that of *LDA*, because *Ours-S* evaluates the relevance with simple accumulation, which fails to capture the relatedness between multiple triples from the source document. We tackle this problem by incorporating the modified iterative propagation over the entire triple graph into *Ours*, which achieves the best performance. One possible problem is why *WE* has a poor performance, the reason of which lies in the setup of our evaluation, where we label positive and negative instances according to the class information of triples and documents. This is better fit for topic model–based methods.

Discussion We further analyze the impact of the three modifications we made to the original model (see Section 3.2). We first focus on the impact of decreasing the relevance weight of bk-nodes and increasing that of sd-nodes after every iteration. As mentioned previously, we change their relevance weight according to a fixed ratio, which is important to the performance. Figure 4 shows the performance of models with different ratios. With any increase of the ratio, our model improves its performance in every metric, which shows the effectiveness of this setup. The performance remains stable from the value of 10:1, which is thus chosen as the final value in our experiments. We then turn to the other two modifications about the edges between bk-nodes and the setup of propagation probability. Table 2 shows the performance of our full model and the simplified models without these two modifications. With the edges between bk-nodes, our model improves the performance by 1.48% in *MAP_5* and by 1.82% in *P@5*. With the modified iterative propagation, we achieve a even greater improvement of 13.99% in *MAP_5* and 24.27% in *P@5*. All these improvements are statistically significant, which indicates the effectiveness of these modifications to the original model.

Model	MAP_5	P@5	MAP_10	P@10
Full	**0.6494**	**0.5597**	**0.6338**	**0.5502**
Full−bb	0.6399	0.5497	0.6254	0.5404
Full−p	0.5697	0.4504	0.5485	0.4409

Table 2: The performance of our full model (*Full*) and two simplified models without modifications: (1) without edges between bk-nodes (*Full−bb*), (2) without the newly proposed definition of propagation probability between nodes (*Full−p*).

5.2 Task-based evaluation

Data preparation To carry out the task-based evaluation, we use the same Chinese corpus as that in previous experiments, which consists of 17,199 documents extracted from Internet news in 9 topics. We also use the same tool (Che et al., 2010) to extract triples of both source document and background knowledge. For every document *D* to be classified, we first use our model to get the top N

Model	P	R	F
VSM+one-hot	0.8214	0.8146	0.8168
VSM+tf-idf	0.8381	0.8333	0.8336
LDA+SVM	0.8512	0.8422	0.8436
LDA+SVM+Ours-S	0.8584	0.8489	0.8501
LDA+SVM+Ours	**0.8748**	**0.8689**	**0.8691**

Table 3: The performance of document classification with (*LDA+SVM+Ours-S*, *LDA+SVM+Ours*) and without (*others*) background knowledge.

most-relevant triples to D, and then use them as extra features for the original model. We conduct a direct comparison between the models with and without background knowledge to evaluate the impact of introducing background knowledge.

Baseline systems We first illustrate two baselines without background knowledge based on VSM and LDA. For VSM, the test document D is represented as a bag of words, where the word distribution over candidate topics is trained on the same corpus mentioned previously. Then we evaluate the similarity between D and a candidate topic with cosine-similarity directly, where the topic with the highest similarity will be chosen as the final class. We use two setups: (1) *VSM-one-hot* represents a word as 1 if it appears in a document or topic, or 0 if not. (2) *VSM-tf-idf* represents a word as the value of tf-idf. For LDA, we re-implement the state-of-the-art system as another baseline, which represents D as a topic vector $\vec{v}_d$ in the parameter estimation step, and then introduces a SVM classifier to take $\vec{v}_d$ as input and decide the final class in the inference step.

We also evaluate the impact of knowledge quality by proposing two different models to introduce background knowledge: our full model introduced in Section 3 (*Ours*), and a simplified version of our model without iterative propagation (*Ours-S*). They have different performances on introducing background knowledge as shown in previous experiments (see Section 5.1). We then conduct a direct comparison between the document classification models with these conditions, whose differing performances demonstrates the impact of different qualities of background knowledge on this task.

Results Table 3 shows the results. We use P, R, F to evaluate the performance, which are computed as the micro-average over 9 topics. Both models with background knowledge (*LDA+SVM+Ours-S*, *LDA+SVM+Ours*) outperform systems without knowledge, which shows that the introduction of background knowledge helps in better classification of documents. The system with the simplified version of our model without iterative propagation (*LDA+SVM+Ours-S*) achieves a F-value of 0.8501, which outperforms the other baselines without knowledge too. Moreover, the system with our full model (*LDA+SVM+Ours*) achieves the best performance, a F-value of 0.8691, and outperforms the best baseline *LDA+SVM* significantly. This shows that introducing better quality of background knowledge is helpful to the better classification of documents. Statistical significance is also verified using the two-tailed paired t-test computed at a 95% confidence level based on the results of classification over the test set.

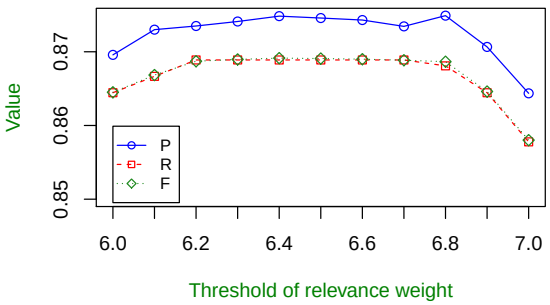


Figure 5: The performance of document classification models with different thresholds. The knowledge whose relevance weight to the source document exceeds the threshold will be introduced as background knowledge.

Discussion One important question here is how much background knowledge to include. As mentioned in Section 4, we have tried two different solutions: (1) introducing a fixed amount of background knowledge for every document, and (2) setting a threshold and selecting knowledge whose relevance weight exceeds the threshold. The results are shown in Table 4, where the systems

Model	P	R	F
Ours-S+Top5	0.8522	0.8444	0.8456
Ours-S+ThreD	**0.8584**	**0.8489**	**0.8501**
Ours+Top5	0.8769	0.8667	0.8677
Ours+ThreD	**0.8748**	**0.8689**	**0.8691**

Table 4: The performance of document classification with the full model (*Ours*) and the simplified model (*Ours-S*) to introduce knowledge.

with threshold outperform that with fixed amount, which shows that the threshold helps in better introduction of background knowledge.

We also evaluate the impact of different thresholds as shown in Figure 5. The performance keeps improving as the threshold increases up to 6.4 and becomes steady from 6.4 to 6.7, while it begins to decline sharply from 6.7. This is reasonable because at the beginning, as the threshold increases, we recall more background knowledge and provide more information. However, with the further increase of the threshold, we introduce more noise, which decreases the performance. In our experiments, we choose 6.4 as the final threshold.

6 Conclusion and Future Work

This study encodes distributional semantics into the triple-based background knowledge ranking model (Zhang et al., 2014) for better document enrichment. We first use LDA to represent every triple as a real-valued vector, which is used to evaluate the relatedness between triples, and then propose a modified iterative propagation model to rank all the triples of background knowledge. For evaluation, we conduct two series of experiments: (1) evaluation as ranking problem, and (2) task-based evaluation, especially for document classification. In the first set of experiments, our model outperforms multiple strong baselines based on VSM, LDA, and WE. In the second set of experiments, our full model with background knowledge outperforms the state-of-the-art systems significantly. Moreover, we also explore the impact of knowledge quality and show its importance.

In our future work, we wish to explore a better way to encode distributional semantics by proposing a modified LDA for better triples representation. In addition, we also want to explore the effect of introducing background knowledge in conjunction with other NLP tasks, especially with discourse parsing (Marcu, 2000; Pitler et al., 2009).

Acknowledgments

We would like to thank our colleagues for their great help. This work was partly supported by National Natural Science Foundation of China via grant 61133012, the National Natural Science Foundation of China Surface Project via grant 61273321, and the National 863 Leading Technology Research Project via grant 2015AA015407.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2007. Measuring semantic similarity between words using web search engines. *Proceedings of the 16th International Conference on World Wide Web*, 7:757–766.

Volha Bryl, Claudio Giuliano, Luciano Serafini, and Kateryna Tymoshenko. 2010. Using background knowledge to support coreference resolution. In *Proceedings of the 2010 Conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, volume 10, pages 759–764.

Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. Ltp: A Chinese language technology platform. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, pages 13–16. Association for Computational Linguistics.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, volume 7, pages 708–716.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open information extraction: The second generation. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume One*, pages 3–10. AAAI Press.

Samah Fodeh, Bill Punch, and Pang-Ning Tan. 2011. On ontology-driven document clustering using core semantic features. *Knowledge and Information Systems*, 28(2):395–421.

Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013. Learning entity representation for entity disambiguation. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, August.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. Yago2: a spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61.

Xiaohua Hu, Xiaodan Zhang, Caimei Lu, Eun K Park, and Xiaohua Zhou. 2009. Exploiting Wikipedia as external knowledge for document clustering. In *Proceedings of the 15th International Conference on Knowledge Discovery and Data Mining*, pages 389–396. ACM.

Saurabh S Kataria, Krishnan S Kumar, Rajeev R Rastogi, Prithviraj Sen, and Srinivasan H Sengamedu. 2011. Entity disambiguation with hierarchical topic models. In *Proceedings of the 17th International Conference on Knowledge Discovery and Data Mining*, pages 1037–1045. ACM.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of Wikipedia entities in web text. In *Proceedings of the 15th International Conference on Knowledge Discovery and Data Mining*, pages 457–466. ACM.

Yogan Jaya Kumar and Naomie Salim. 2012. Automatic multi document summarization approaches. *Journal of Computer Science*, 8(1).

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge.

Daniel Marcu. 2000. The rhetorical parsing of unrestricted texts: A surface-based approach. *Computational Linguistics*, 26(3):395–448.

Vivi Nastase, Michael Strube, Benjamin Börschinger, Cäcilia Zirn, and Anas Elghafari. 2010. Wikinet: A very large scale multi-lingual concept network. In *Proceeding of the 7th International Conference on Language Resources and Evaluation*.

Patrick Pantel and Ariel Fuxman. 2011. Jigs and lures: Associating web queries with structured entities. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 83–92. Association for Computational Linguistics.

Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. 2008. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web*, pages 91–100. ACM.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. 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: Volume 2-Volume 2*, pages 683–691. Association for Computational Linguistics.

Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

Jean-Paul Sansonnet and François Bouchet. 2010. Extraction of agent psychological behaviors from glosses of WordNet personality adjectives. In *Proc. of the 8th European Workshop on Multi-Agent Systems (EUMAS10)*.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March.

Prithviraj Sen. 2012. Collective context-aware topic models for entity disambiguation. In *Proceedings of the 21st International Conference on World Wide Web*, pages 729–738. ACM.

Steffen Staab and Rudi Studer. 2009. *Handbook on Ontologies*. Springer Publishing Company, Incorporated, 2nd edition.

Koun-Tem Sun, Yueh-Min Huang, and Ming-Chi Liu. 2011. A WordNet-based near-synonyms and similar-looking word learning system. *Educational Technology & Society*, 14(1):121–134.

Ellen M Voorhees, Donna K Harman, et al. 2005. *TREC: Experiment and evaluation in information retrieval*, volume 63. MIT press Cambridge.

Muyu Zhang, Bing Qin, Ting Liu, and Mao Zheng. 2014. Triple based background knowledge ranking for document enrichment. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 917–927, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

Zhicheng Zheng, Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010. Learning to link entities with knowledge base. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 483–491. Association for Computational Linguistics.