Deep Context: A Neural Language Model for Large-scale Networked Documents

Hao Wu
USC ISI
hwu732@usc.edu

Kristina Lerman
USC ISI
lerman@isi.edu

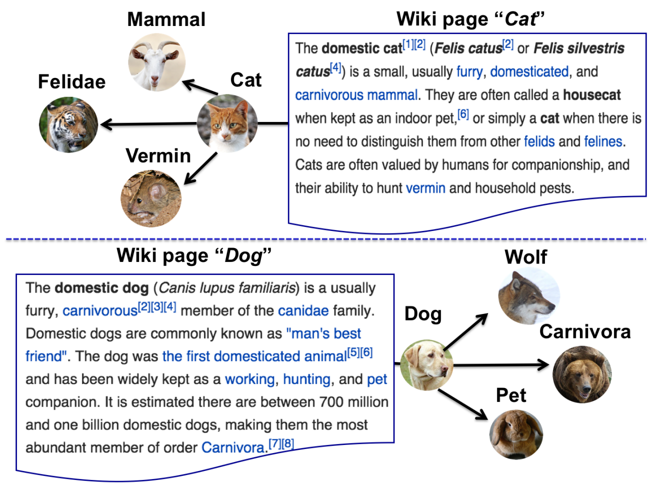
Abstract

We propose a scalable neural language model that leverages the links between documents to learn the deep context of documents. Our model, Deep Context Vector, takes advantage of distributed representations to exploit the word order in document sentences, as well as the semantic connections among linked documents in a document network. We evaluate our model on large-scale data collections that include Wikipedia pages, and scientific and legal citations networks. We demonstrate its effectiveness and efficiency on document classification and link prediction tasks.

1 Introduction

Text mining applications use quantitative representations of documents to analyze and compare them to one another. One popular approach to text modeling represents each document as a vector of its word frequencies. Due to its conceptual simplicity and computational efficiency, this approach is used widely in information retrieval, text summarization, and personalized recommendation. However, representing documents by their word frequencies has significant disadvantages that limit the utility of this representation. The principal of these is that word frequencies fail to capture word meaning. Individual words may be highly ambiguous: the same word can often mean different things, and different words frequently have the same meaning. To address this challenge, modern text analysis methods take advantage of language models (n-grams or topic modeling algorithms) which include the context of words in document representations, where a word's context is provided by the neighboring words, phrases in sentences and co-occurred words in same documents. Then they use quantitative methods to find statistical dependencies among words across documents. The intuition behind this approach is that surrounding words in a document, though themselves ambiguous, collectively help to pin down a given word's meaning.

However, words often have a deeper context that extends beyond nearby words, phrases, and sentences in the same document to other relevant documents and concepts. For example, the online encyclopedia Wikipedia is composed of a network of Web pages describing interconnected concepts



The **domestic cat**[1][2] (***Felis catus***[2] or ***Felis silvestris catus***[4]) is a small, usually furry, domesticated, and carnivorous mammal. They are often called a **housecat** when kept as an indoor pet,[6] or simply a **cat** when there is no need to distinguish them from other felids and felines. Cats are often valued by humans for companionship, and their ability to hunt vermin and household pests.

The **domestic dog** (*Canis lupus familiaris*) is a usually furry, carnivorous[2][3][4] member of the canidae family. Domestic dogs are commonly known as "man's best friend". The dog was the first domesticated animal[5][6] and has been widely kept as a working, hunting, and pet companion. It is estimated there are between 700 million and one billion domestic dogs, making them the most abundant member of order Carnivora.[7][8]

Figure 1: Wikipedia pages "cat" and "dog" with representative notions they refer to.

and entities. Figure 1 illustrates a small portion of Wikipedia related to the concept "cat". The text of the page describing "cat" references other pages describing concepts "felidae", "mammal" and "vermin". Similarly, the Wikipedia page describing the concept "dog" links to related pages describing "wolf", "pet", "carnivora". In order to get a complete picture of what "cat" and "dog" are, one has to read the descriptions in the linked pages. Similarly, in order to understand a scientific article, a reader must rely not only on the text of the article, but also on the background knowledge and supporting evidence that is described in other articles. Thus, the meaning the reader perceives is not simply derived from the words and sentences appearing in the article, but is the inferential result of the connections the article makes to other articles and the concepts expressed in them. These connections, and the text used by them to express concepts and themes of the documents, provide a deeper context for understanding the current document. Automatically learning these deep contexts from data will help us create better models of text documents that not only help to better solve existing tasks, such as finding documents similar to the given document, but also address novel text mining tasks that existing tools cannot solve.

In this paper, we address the problem of learning the deep contexts of documents with neural language models [Bengio *et al.*, 2003]. We focus on leveraging information in document text and the links that exist between documents in document networks. We describe a model that captures the generative process of document creation in which language and semantics are intricately linked across document networks. Authors read existing documents to draw inspiration for the vocabulary, grammar, and style and learn how to describe the concepts and ideas in their own works. Language influences propagate between documents through citations and cross-references, providing a deeper context for understanding the semantics of text. The question is how to capture the hierarchy of contexts of a word in a sentence, including the surrounding words, the sentence semantics, underlying theme of the article and influence from cited articles. To this end, we devise a new language model that accounts for these contexts in text documents.

2 Related Work

We begin with a review of two lines of related work.

2.1 Neural Language Models

Neural language models are based on the idea of distributed word representations [Bengio *et al.*, 2003]. Instead of "one-hot" representations for words in vocabulary, neural language models use continuous variables to represent words in vector space, in the hope of improving the generalization of classic n-gram language models. This idea of distributed word representations has been successfully applied to many tasks in natural language processing and data mining, such as modeling unigram words and phrases [Mikolov *et al.*, 2013], sentences and documents [Le and Mikolov, 2014], relational entities [Bordes *et al.*, 2013; Socher *et al.*, 2013], general text-based attributes [Kiros *et al.*, 2014], streaming documents [Djuric *et al.*, 2015] and semi-supervised learning of text embedding [Tang *et al.*, 2015a]. Representative applications of neural language models also include learning representations of nodes in networks [Perozzi *et al.*, 2014; Tang *et al.*, 2015b; Chang *et al.*, 2015; Grover and Leskovec, 2016; Wang *et al.*, 2016].

2.2 Semantic and Link Analysis

Modeling underlying structure of text documents and learning semantic representations is critical to various applications including information retrieval, text summarization and personalized recommendation. Approaches of learning latent semantics (notably "topic modeling" methods) include probabilistic Latent Semantic Analysis (PLSA [Hofmann, 1999]) and Latent Dirichlet Allocation (LDA [Blei *et al.*, 2003]) are proposed. However, most of these models do not account for data with the link structure where text documents lie on. Research work in link analysis [Hoff *et al.*, 2002; Kemp *et al.*, 2004; Airoldi *et al.*, 2009] attempt to model network structure in latent space. However, these approaches cannot be easily applied to node with text content. The goal of our model is closest to recent work in joint content and link analysis. Several models [Mccallum *et al.*, 2005; Dietz *et al.*, 2007; Nallapati *et al.*, 2008; Mei *et al.*, 2008; Chang and Blei, 2009] have been proposed based on topic modeling, and links between documents are explained by the similarity of topic mixtures. However, our model goes beyond the bag-of-words representations where these models bear in common, and exploit word order information in sentences.

3 Deep Context Neural Language Model

In this section, we describe our model in detail. To begin with, we review the log-bilinear language model [Mnih and Hinton, 2007], which is the foundation of our model.

3.1 The Log-Bilinear Model

We represent each word w using an F-dimensional real-valued feature vector, $\mathbf{v}_w \in \mathbb{R}^F$. The log-bilinear model (LBL [Mnih and Hinton, 2007; Mnih and Kavukcuoglu, 2013]) specifies the bilinear energy function of a sequence of context words $(w_1, \cdots, w_{n-1})$ and the predicted next word w_n:

$$E(w_{1:n-1}; w_n) = -\hat{\mathbf{v}}^\top \mathbf{v}_{w_n} \tag{1}$$

where $\hat{\mathbf{v}}$ is the predicted representation of next word, defined as

$$\hat{\mathbf{v}} = \sum_{i=1}^{n-1} \mathbf{c}_i \odot \mathbf{v}_{w_i} \tag{2}$$

where $\odot$ denotes element-wise multiplication, and $\mathbf{c}_i$ is the weight vector for the context word in position i. For symmetry, we use the same set of word representations for both the words being predicted and the context words.

The resulting probabilistic distribution of next word is given by a softmax function:

$$P(w_n|w_{1:n-1}) = \frac{1}{Z_c} \exp[-E(w_{1:n-1}; w_n)] \tag{3}$$

where $Z_c = \sum_{w_n=1}^{K} \exp[-E(w_{1:n-1}; w_n)]$ is the context dependent normalization factor, and K is the vocabulary size.

3.2 Model Architecture

Our model assumes that the generation of the next word in a word sequence depends not only on the preceding words, but also on the global context of the document and the other documents it references (and possibly the documents these references cite, and so forth). Figure 2 illustrates a general architecture, where we use the word sequence example "dogs are our best $\cdots$" in the document related to the concept "DOG". The concepts "PET" and "CAT" are also used to predict the next word "friends", as they are neighbors of "DOG" in the document network. More precisely, we firstly consider there is global semantic context for word sequences in each document d, which we also represent as a real-valued feature vector, $\mathbf{v}_d$. This idea is same as in [Le and Mikolov, 2014]. We also consider the semantic influence from the neighborhood of d in the document network (i.e., documents that can be reached from d in a few hops), and jointly model text in documents and their link structure. For simplicity, we learn word and document representations in the same F-dimensional vector space, and consider $\mathbf{v}_w \in \mathbb{R}^F, \mathbf{v}_d \in \mathbb{R}^F$.

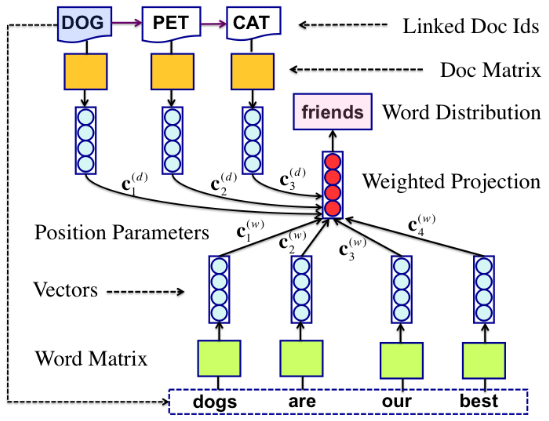


Figure 2: The general model architecture of Deep Context Vectors. We refer to it as **DCV-vLBL**.

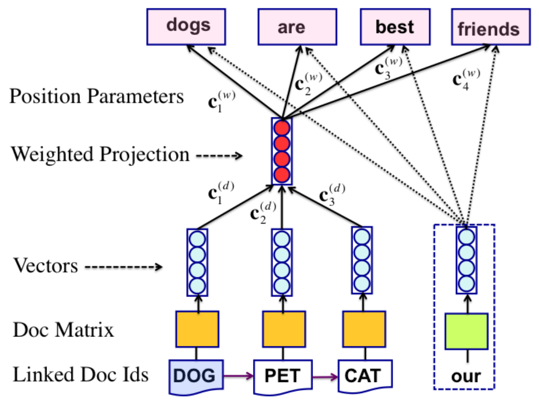


Figure 3: An alternative model architecture. We refer to it as **DCV-ivLBL** as it performs inverse language modeling.

Given a sequence of words $(w_1, \cdots, w_n)$ in a document d_1, we define the energy function as:

$$E(\mathcal{N}_{d_1}^m, w_{1:n-1}; w_n) = -\hat{\mathbf{v}}^\top \mathbf{v}_{w_n} \tag{4}$$

where $\mathcal{N}_{d_1}^m$ is the set of neighbors of d_1 that can be reached in $\leq m$ hops along the links in the document network. We take $\hat{\mathbf{v}}$ to be the predicted representation of the next word:

$$\hat{\mathbf{v}} = \left(\sum_{i=1}^{n-1} \mathbf{c}_i^{(w)} \odot \mathbf{v}_{w_i}\right) + \frac{1}{|\mathcal{N}_{d_1}^m|}\left(\sum_{p=1}^{|\mathcal{N}_{d_1}^m|} \mathbf{c}_p^{(d)} \odot \mathbf{v}_{d_p}\right), \tag{5}$$

where $d_p \in \mathcal{N}_{d_1}^m$, and $\mathbf{c}_p^{(d)} \in \mathbb{R}^F$ is the context parameters defining the weights of d_p. We may reuse $\mathbf{c}_p^{(d)}$ for any d_p that can be reached in the same number of hops. The probabilistic distribution of the next word is defined as

$$\begin{aligned} &P(w_n | w_{1:n-1}, \mathcal{N}_{d_1}^m) \\ =&\frac{1}{Z_c}\exp[-E(\mathcal{N}_{d_1}^m, w_{1:n-1}; w_n)] \end{aligned} \tag{6}$$

By learning next word's representation using features from its global document vector and the linked documents, our model captures the deep context for generating each word in a document. In this sense, we call the vectors learned by our model *Deep Context Vectors* (DCV). However, the model in Eq. (6) is computationally expensive due to potentially exponentially large number of documents within m hops, $\mathcal{N}_{d_1}^m$. To reduce computation, we introduce a practical algorithm that draws node sequences to characterize the neighborhoods.

Randomized Document Sequence

Given a sequence of words $(w_1, \cdots, w_n)$ within a document d_1, we draw an associated sequence of documents $(d_1, \cdots, d_{m+1})$ from $\mathcal{N}_{d_1}^m$ by following the directed links from document d_1. The document sequence can be sampled based on a probability distribution $P(d_{1:m+1}) = \prod_{j=1}^{m+1} P(d_j | d_{1:j-1})$. For simplicity, we adopt first order random walk scheme. In the first step, we start from the original document d_1. In each following step, we uniformly sample d_{j+1} from the set of documents that the current document d_j links to. Although higher order random walk is more powerful, we leave studying this direction as future work.

We hence redefine the energy function in Eq. (4) as:

$$E(d_{1:m+1}, w_{1:n-1}; w_n) = -\hat{\mathbf{v}}^\top \mathbf{v}_{w_n} \tag{7}$$

where $\hat{\mathbf{v}}$ is the predicted representation of the next word, and is learned from the feature vectors of preceding words as well as associated documents:

$$\hat{\mathbf{v}} = \left(\sum_{i=1}^{n-1} \mathbf{c}_i^{(w)} \odot \mathbf{v}_{w_i}\right) + \left(\sum_{j=1}^{m+1} \mathbf{c}_j^{(d)} \odot \mathbf{v}_{d_j}\right). \tag{8}$$

Here $\mathbf{c}_j^{(d)} \in \mathbb{R}^F$ is hop-dependent and specifying the weights of the feature vector of j-th document in the document sequence.

The conditional word distribution is given by

$$\begin{aligned} &P(w_n | w_{1:n-1}, d_{1:m+1}) \\ =&\frac{1}{Z_c}\exp[-E(d_{1:m+1}, w_{1:n-1}; w_n)] \end{aligned} \tag{9}$$

3.3 Learning with Negative Sampling

The word and document vectors are initialized with random values and trained by minimizing the negative log-likelihood of the data. The parameter updates are performed using stochastic gradient descent, and can be derived from Eq. (9):

$$\Delta\boldsymbol{\theta} = \epsilon \nabla_{\boldsymbol{\theta}} \log P(w_n | w_{1:n-1}, d_{1:m+1}) \tag{10}$$

where $\boldsymbol{\theta} = \{\mathbf{v}_w, \mathbf{v}_d, \mathbf{c}^{(w)}, \mathbf{c}^{(d)}\}$ is the set of parameters to be learned, and ϵ is the learning rate. The computation of gradients obtained in Eq. (10) involves the normalization term and is expensive since the complexity is proportional to the vocabulary size K. In this paper, we use a scalable optimization algorithm without calculation of the gradients of normalization, that is negative sampling [Mikolov *et al.*, 2013]. Negative sampling trains a logistic regression to distinguish between data samples of w_n from "noise". In our model, the objective is to maximize

$$\log \sigma(\hat{\mathbf{v}}^\top \mathbf{v}_{w_n}) + \sum_{i=1}^{k} \mathbb{E}_{w_i} \sim P_n(w)\left[\log \sigma(-\hat{\mathbf{v}}^\top \mathbf{v}_{w_i})\right] \tag{11}$$

where $\sigma(x)$ is the sigmoid function. $P_n(w)$ is the global unigram distribution of the training data acting as the noise distribution where we draw k negative samples.

3.4 Improving with Alternative Architectures

We explore different model architectures, in the hope of improving the performance with ensemble of feature vectors learned with them separately. We present an alternative architecture which uses similar scheme of Skip-gram [Mikolov *et al.*, 2013]. Figure 3 shows the architecture, which utilizes the current word to predict its word context (including preceding and following words), along with a weighted vector learned from the document sequence and also used to predict the word context. As it performs inverse language modeling, we refer to it as **DCV-ivLBL** and it's a counterpart of inverse vector LBL [Mnih and Kavukcuoglu, 2013]. The objective is to maximize the log-likelihood for a given word sequence $(w_{t-b}, \cdots, w_t, \cdots, w_{t+b})$

$$\sum_{-b \le i \le b, i \ne 0} \log P(w_{t+i}|w_t) + \sum_{-b \le i \le b, i \ne 0} \log P(w_{t+i}|d_{1:m+1}) \tag{12}$$

where b is the context size of words. The current word w_t and $d_{1:m+1}$ are used separately to predict the context words. The probability distributions is formulated as

$$P(w_{t+i}|w_t) = \frac{1}{Z_c^w} \exp[\mathbf{v}_{w_t} \top (\mathbf{c}_i \odot \mathbf{v}_{w_{t+i}})] \tag{13}$$

$$\begin{aligned} & P(w_{t+i}|d_{1:m+1}) \\ = & \frac{1}{Z_c^d} \exp\left[\left(\sum_{j=1}^{m+1} \mathbf{c}_j^{(d)} \odot \mathbf{v}_{d_j}\right) \top (\mathbf{c}_i^{(w)} \odot \mathbf{v}_{w_{t+i}}) \right] \end{aligned} \tag{14}$$

where Z_c^w and Z_c^d are the normalization term.

4 Experiment

In this section, we present evaluation results of our model. We begin with a description of our data sets.

4.1 Data Description

We select data sets from different domains including Wikipedia pages, scientific papers and legal opinions:

- Wikipedia: A dump of Wikipedia pages[1] in October 2015 is used in our experiments. Wikipedia data provides rich text with interlinked knowledge concepts. The document network is dense, with around 40 out-links per document on average.
- DBLP: We download the DBLP data set [Tang *et al.*, 2008][2], which contains a collection of papers with titles and citation links. This represents a type of documents with extremely short text, with only about 9 words per document.

[1]`http://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2`

[2]`http://arnetminer.org/lab-datasets/citation/DBLP_citation_2014_May.zip`

Table 1: Statistics of data sets.

Dataset	Wikipedia	DBLP	Legal
# docs	4,256,787	2,244,021	2,485,004
# links	166,212,300	4,354,534	4,682,528
# links/doc	39.5	1.9	1.8
# words	3,750,870,725	20,741,535	6,391,118,754
# words/doc	881.1	9.2	2,571.8
vocab size	3,888,263	248,377	2,593,557

- Legal: We collect a large digitized record of federal court opinions from the CourtListener[3] project in our study. The text in each case of legal opinion is mainly about the discourse of a legal case. The citation network is sparse and the lengths of documents are much larger than that of Wikipedia and DBLP, with more than 2 thousand words on average.

The statistics of the three datasets are shown in Table 1.

4.2 Methods for Comparison

We compare our model with the following baselines and the state-of-art approaches for learning document representations:

- BOW. We use bag-of-words representations as baseline, which is simply the frequency of unigrams in a document. We report performance on TFIDF weighted BoW.
- LDA [Blei *et al.*, 2003; Hoffman *et al.*, 2010]: Latent Dirichlet Allocation is used to learn document-specific topic distributions. By comparing our models with LDA, we may understand how important it is to modeling word order and the context of linked documents.
- Skip-gram [Mikolov *et al.*, 2013]: We learn word vectors using Skip-gram model. Each document is represented by averaging vectors of all words in it.
- PV [Le and Mikolov, 2014]: Paragraph Vector (PV) learns distributed representations for variable length pieces of text, such as sentences, paragraph and documents. By comparing our model with PV, we may see how informative the links are.
- DeepWalk [Perozzi *et al.*, 2014]: DeepWalk is used for learning distributed representations of nodes in a network. In order to understand how informative the word content are, we consider the word-word co-occurrence network G_{ww} and word-document network G_{wd}. In addition, we also use document-document network G_{dd} to understand the effects of links.
- LINE [Tang *et al.*, 2015b]: LINE models first-order and second-order proximity between nodes. LINE is optimized with edge-sampling and trained on word-word network G_{ww}, word-document network G_{wd} and document-document network G_{dd}.

We perform experiments on a single machine with 64 CPU cores at 2.3 GHz, and 256G memory. Asynchronous stochastic gradient descent algorithm is used with 40 threads to optimize our models. Models are trained for 20 epochs on each dataset.

[3]`https://www.courtlistener.com/`

4.3 Document Classification

In this experiment, we perform document classification and evaluate how discriminative the learned feature vectors are.

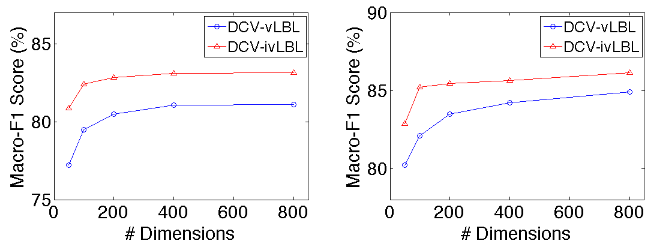
Experiment Setup

For each dataset, all text, links of all documents or the combination (listed in Table 1) are used for learning document representations in unsupervised manner. For evaluation, we sample a subset of documents with balanced category distribution for each dataset. (1) For Wikipedia dataset, we randomly select 10,000 samples from each category among the seven diverse ones "Arts", "History", "Human", "Mathematics", "Nature", "Technology", and "Sports"; (2) For DBLP papers, we select five research fields including "Artificial Intelligent", "Computer Graphics", "Computer Networks", "High Performance Computing", "Theory" as class labels. For each field, we randomly sample 10,000 papers; (3) In legal opinions, legal code are cited to support the judgement of a case. Each statute has a standard format, with title, code and section/subsection numbers. We use the statute title cited most by each case as its category label. We choose the eight most cited statute titles, namely "Title 8: Aliens and Nationality", "Title 11: Bankruptcy ", "Title 15: Commerce and Trade", "Title 18: Crimes and Criminal Procedure", "Title 21: Food and Drugs", "Title 28: Judiciary and Judicial Procedure", "Title 29: Labor" and "Title 42: The Public Health and Welfare". We randomly select 10,000 samples for each statue title.

The dimensionality of word and document vectors are fixed as 400 for all learning models. The number of negative sampling is fixed as 5 for Skip-gram, PV, LINE and DCV. We set the word context window size $n = 5$ in DCV-vLBL and $b = 5$ in DCV-ivLBL. The context window of document sequence m is fixed as 1 by which we only consider the immediate neighbors that the current document links to. Increasing m, DCV may yield performance gain, but with higher cost of computation. To construct word-word network G_{ww}, we consider co-occurrences of words with context window size 5. We also test model performance using concatenation of document vectors trained using PV-DBOW and PV-DM, as well as DCV-vLBL and DCV-ivLBL. For LINE, we use concatenation of the vectors trained on first and second-order proximity [Tang *et al.*, 2015b].

Results of Document Classification

We train one-vs-rest logistic regression on the resulting vector representations of documents.Table 2 shows the average Micro-F1 and Macro-F1 scores. The results are average over 5-fold cross-validation on the sampled data. The baseline BOW works well on all datasets, but with features in dimensionality of millions. Learning models with 400 dimensional vectors reduce the document feature space by at least 99.8% for any dataset. Our DCV consistently performs better than PV, LINE, DeepWalk and LDA. The performance gain of DCV is significant on DBLP where the documents (paper titles) are extremely short. The performance of PV degrades in modeling short text as the vectors learned will overfit the few words in each document. However, DCV does not suffer from this problem as the links provide discriminative information for learning better document representations. On Legal dataset, it's worth noticing that most of documents in Legal dataset are long documents with thousands of words. DCV is not easily tuned in this case, but still perform well. In most cases, models that use text features only can achieve good performance. The link information can slightly help document classification, but models using link information solely cannot perform well. We also present the running time to train vectors for each methods on Legal dataset. As we can see, DCV can efficiently train vectors on 2M documents with 6.3B words and 4M links in about 10 to 26 hours, which is quite scalable.



(a) Wikipedia (b) Legal

Figure 4: Performance w.r.t # of vector dimensions

Dimension of Vectors

We vary the dimensionality of vectors in DCV, and report the classification performance on Wikipedia and Legal datasets in Figure 4. With larger dimensionality of vectors, the vectors are trained at a higher computational cost. We can see the performance of DCV increases when dimensions of vectors increase to 200 or 400, which is suitable for both datasets. However, when dimensions increase to 800, the performance gain is marginal.

4.4 Link Prediction

In this experiment, we investigate how much information the learned vectors can provide for predicting unseen links between documents.

Experiment Setup

For Wikipedia and DBLP datasets, we hold out all out-links of the test data as described in the task of document classification (Section 4.3) for evaluation. For Legal dataset, we hold out all out-links of legal opinions of 62,018 supreme court decisions, each has about 4 links on average. For each test document, we rank other documents by Cosine similarity in vector space, instead of predict whether it will or not link to any other documents. In this experiment, we fix the number of hops m as 2 in sampling document sequence.

Results of Link Prediction

Table 3 shows the results averaged over all test documents. We use Precision at the cut-off 10 (P@10) and Mean Average Precision (MAP) for evaluation. As we can see, DCV consistently performs better than other methods, and the performance gain is significant on Wikipedia dataset which has relatively denser document-document network. DCV also performs well on DBLP dataset. This demonstrates the capacity

Table 2: Performance of document classification (%).

		Wikipedia		**DBLP**		**Legal**		
Modal	Method	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Time
text	BOW	81.54	81.56	72.25	72.24	86.12	86.18	–
	LDA [Blei *et al.*, 2003]	72.71	72.70	57.98	58.20	78.32	78.56	40h
	Skip-gram [Mikolov *et al.*, 2013]	77.20	77.25	67.34	67.39	80.13	80.28	8.5h
	PV-DM [Le and Mikolov, 2014]	79.48	79.52	64.30	64.27	82.23	82.44	9.8h
	PV-DBOW [Le and Mikolov, 2014]	80.31	80.26	68.73	68.75	81.32	81.43	21.5h
	PV-(DM+DBOW) [Le and Mikolov, 2014]	81.10	81.08	68.91	68.92	82.78	82.85	–
	DeepWalk($G_{ww} + G_{wd}$) [Perozzi *et al.*, 2014]	76.42	76.42	66.42	66.45	80.78	80.83	10.6h
	LINE($G_{ww} + G_{wd}$) [Tang *et al.*, 2015b]	81.20	81.20	66.10	66.15	81.64	81.78	20.3h
links	DeepWalk(G_{dd}) [Perozzi *et al.*, 2014]	70.33	70.39	56.17	54.84	56.87	55.52	0.8h
	LINE(G_{dd}) [Tang *et al.*, 2015b]	73.92	73.90	55.43	54.27	56.93	55.56	1.2h
text + links	DeepWalk($G_{ww} + G_{wd} + G_{dd}$) [Perozzi *et al.*, 2014]	77.32	77.27	72.48	72.62	82.63	82.72	11.3h
	LINE($G_{ww} + G_{wd} + G_{dd}$) [Tang *et al.*, 2015b]	81.74	81.73	72.55	72.60	82.54	82.59	20.5h
	DCV-vLBL	81.05	81.05	72.16	72.13	84.20	84.22	10.2h
	DCV-ivLBL	83.10	83.11	72.87	72.95	85.63	85.67	26.3h
	DCV-(vLBL + ivLBL)	**84.27**	**84.29**	**73.45**	**73.40**	**86.47**	**86.56**	–

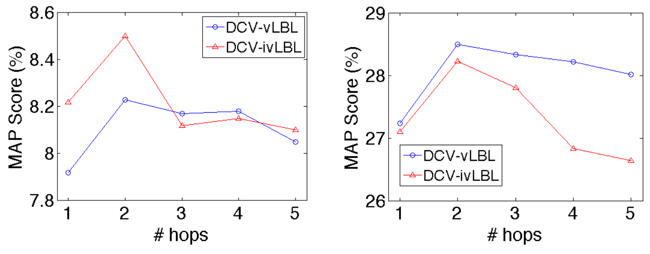
Table 3: Performance of link prediction (%).

		Wikipedia		**DBLP**		**Legal**	
Modal	Method	P@10	MAP	P@10	MAP	P@10	MAP
text	BOW	14.23	27.52	3.84	5.63	10.43	22.38
	LDA [Blei *et al.*, 2003]	7.15	13.57	1.27	2.31	6.03	14.13
	Skip-gram [Mikolov *et al.*, 2013]	13.12	25.23	3.54	5.49	10.56	23.10
	PV-(DM+DBOW) [Le and Mikolov, 2014]	15.21	28.15	3.58	5.21	11.54	24.57
	DeepWalk($G_{ww} + G_{wd}$) [Perozzi *et al.*, 2014]	13.74	26.13	3.35	5.09	9.34	19.26
	LINE($G_{ww} + G_{wd}$) [Tang *et al.*, 2015b]	14.02	27.48	3.32	5.07	10.56	21.78
links	DeepWalk(G_{dd}) [Perozzi *et al.*, 2014]	10.34	20.50	3.92	6.01	12.54	23.56
	LINE(G_{dd}) [Tang *et al.*, 2015b]	10.21	21.28	3.87	5.94	13.68	24.57
text + links	DeepWalk($G_{ww} + G_{wd} + G_{dd}$) [Perozzi *et al.*, 2014]	16.58	31.42	5.13	6.42	15.23	25.43
	LINE($G_{ww} + G_{wd} + G_{dd}$) [Tang *et al.*, 2015b]	16.78	31.95	5.07	6.35	16.23	26.19
	DCV-vLBL	18.32	32.68	6.72	8.23	17.89	28.50
	DCV-ivLBL	19.20	34.14	7.10	8.50	18.05	28.23
	DCV-(vLBL + ivLBL)	**19.63**	**35.61**	**7.12**	**8.59**	**18.54**	**28.77**

of DCV in modeling short text with link information. Exploiting both text and link information, DeepWalk, LINE and DCV generally perform better than using text or link solely. LDA performs worst in this task. This indicates the topic distributions learned from bag-of-words features do not provide much information for link prediction.

Number of Hops

We explore the effects of the number of hops, m. Figure 5 shows the link prediction results using DCV with different number of hops used. We observe for DBLP dataset, the performance of DCV does not change much with hops in the range of $1\sim 3$. Hence small number of hops are preferable due to its low computational cost. DCV performs better with the number of hops is set as 2 than other values for DBLP and Legal datasets. Sampling long document sequence has high computational cost and is not effective in real applications.



(a) DBLP (b) Legal

Figure 5: Performance w.r.t. number of hops

5 Conclusion and Future Work

In this paper, we proposed a neural language model called Deep Context Vectors (DCV) for linked documents in document networks. We described two model architectures and showed how to simultaneously learn word and document vectors in our framework. By modeling the deep contexts of words in linked documents, our model can learn better document representations in a low-dimensional vector space. Experimental results showed that our model outperforms other neural language models, such as Paragraph Vector [Le and Mikolov, 2014], that do not take into account the links between documents. Interestingly, the vectors learned by DCV, which have on the order of few hundreds of features, can perform better or comparably to the bag-of-words model, which uses millions of features.

In future work, we would like to use side information such as the relative *location* of each word and the hyperlinks or citation in the paragraph, to determine which linked documents should be modeled for each word.

Acknowledgements

This work was supported in part by the National Science Foundation under grant SMA-1360058 and in part by Leidos. We thank anonymous reviewers for their helpful comments.

References

[Airoldi *et al.*, 2009] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. In *NIPS*, pages 33–40, 2009.

[Bengio *et al.*, 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *JMLR*, 3:1137–1155, 2003.

[Blei *et al.*, 2003] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.

[Chang and Blei, 2009] Jonathan Chang and David M Blei. Relational topic models for document networks. In *AISTATS*, pages 81–88, 2009.

[Chang *et al.*, 2015] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 119–128. ACM, 2015.

[Dietz *et al.*, 2007] Laura Dietz, Steffen Bickel, and Tobias Scheffer. Unsupervised prediction of citation influences. In *Proceedings of ICML*, pages 233–240. ACM, 2007.

[Djuric *et al.*, 2015] Nemanja Djuric, Hao Wu, Vladan Radosavljevic, Mihajlo Grbovic, and Narayan Bhamidipati. Hierarchical neural language models for joint representation of streaming documents and their content. In *Proceedings of the 24th International Conference on World Wide Web*, pages 248–255, 2015.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of SIGKDD*, pages 855–864, 2016.

[Hoff *et al.*, 2002] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

[Hoffman *et al.*, 2010] Matthew Hoffman, Francis R Bach, and David M Blei. Online learning for latent dirichlet allocation. In *NIPS*, pages 856–864, 2010.

[Hofmann, 1999] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of SIGIR*, pages 50–57. ACM, 1999.

[Kemp *et al.*, 2004] Charles Kemp, Thomas L Griffiths, and Joshua B Tenenbaum. Discovering latent classes in relational data. 2004.

[Kiros *et al.*, 2014] Ryan Kiros, Richard S Zemel, and Ruslan Salakhutdinov. A multiplicative model for learning distributed text-based attribute representations. *arXiv preprint arXiv:1406.2710*, 2014.

[Le and Mikolov, 2014] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

[Mccallum *et al.*, 2005] Andrew Mccallum, Andrés Corrada-emmanuel, and Xuerui Wang. Topic and role discovery in social networks. In *In IJCAI*, 2005.

[Mei *et al.*, 2008] Qiaozhu Mei, Deng Cai, Duo Zhang, and ChengXiang Zhai. Topic modeling with network regularization. In *Proceedings of WWW*, pages 101–110. ACM, 2008.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[Mnih and Hinton, 2007] Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In *Proceedings of ICML*, pages 641–648. ACM, 2007.

[Mnih and Kavukcuoglu, 2013] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *NIPS*, pages 2265–2273, 2013.

[Nallapati *et al.*, 2008] Ramesh M Nallapati, Amr Ahmed, Eric P Xing, and William W Cohen. Joint latent topic models for text and citations. In *Proceedings of SIGKDD*, pages 542–550. ACM, 2008.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. *arXiv preprint arXiv:1403.6652*, 2014.

[Socher *et al.*, 2013] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, pages 926–934, 2013.

[Tang *et al.*, 2008] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of SIGKDD*, pages 990–998. ACM, 2008.

[Tang *et al.*, 2015a] Jian Tang, Meng Qu, and Qiaozhu Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of SIGKDD*, pages 1165–1174. ACM, 2015.

[Tang *et al.*, 2015b] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of WWW*, pages 1067–1077, 2015.

[Wang *et al.*, 2016] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1225–1234. ACM, 2016.