DocTag2Vec: An Embedding Based Multi-label Learning Approach for Document Tagging

Sheng Chen[♯*], **Akshay Soni**[†], **Aasish Pappu**[‡], **Yashar Mehdad**[§]
[♯]University of Minnesota-Twin Cities, Minneapolis, MN 55455, USA
[†]Yahoo Research, Sunnyvale, CA 94089, USA and [‡]New York, NY 10036, USA
[§]Airbnb, San Francisco, CA 94103, USA
chen2832@umn.edu {akshaysoni,aasishkp}@yahoo-inc.com
yashar.mehdad@airbnb.com

Abstract

Tagging news articles or blog posts with relevant tags from a collection of predefined ones is coined as document tagging in this work. Accurate tagging of articles can benefit several downstream applications such as recommendation and search. In this work, we propose a novel yet simple approach called DocTag2Vec to accomplish this task. We substantially extend Word2Vec and Doc2Vec – two popular models for learning distributed representation of words and documents. In DocTag2Vec, we simultaneously learn the representation of words, documents, and tags in a joint vector space during training, and employ the simple k-nearest neighbor search to predict tags for unseen documents. In contrast to previous multi-label learning methods, DocTag2Vec directly deals with raw text instead of provided feature vector, and in addition, enjoys advantages like the learning of tag representation, and the ability of handling newly created tags. To demonstrate the effectiveness of our approach, we conduct experiments on several datasets and show promising results against state-of-the-art methods.

1 Introduction

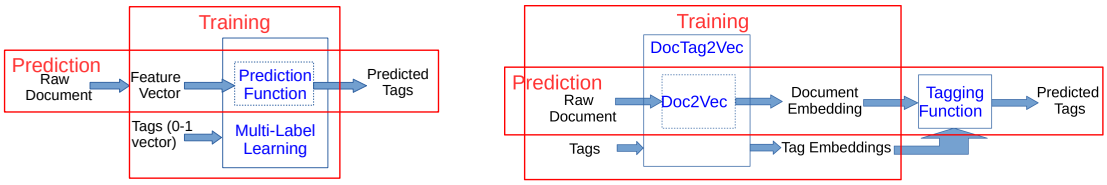
Every hour, several thousand blog posts are actively shared on social media; for example, blogging sites such as Tumblr[1] had more than 70 billion posts by January 2014 across different communities (Chang et al., 2014). In order to reach right audience or community, authors often assign keywords or "#tags" (hashtags) to these blog posts. Besides being topic-markers, it was shown that hashtags also serve as group identities (Bruns and Burgess, 2011), and as brand labels (Page, 2012). On Tumblr, authors are allowed to create their own tags or choose existing tags to label their blog. Creating or choosing tags for maximum outreach can be a tricky task and authors may not be able to assign all the relevant tags. To alleviate this problem, algorithm-driven document tagging has emerged as a potential solution in recent times. Automatically tagging these blogs has several downstream applications, e.g., blog search, cluster similar blogs, show topics associated with trending tags, and personalization of blog posts. For better user engagement, the personalization algorithm could match user interests with the tags associated with a blog post.

From machine learning perspective, document tagging is by nature a *multi-label learning* (MLL) problem, where the input space is certain feature space $\mathcal{X}$ of document and the output space is the power set $2^{\mathcal{Y}}$ of a finite set of tags $\mathcal{Y}$. Given training data $\mathcal{Z} \subset \mathcal{X} \times 2^{\mathcal{Y}}$, we want to learn a function $f : \mathcal{X} \mapsto 2^{\mathcal{Y}}$ that predicts tags for unseen documents. As shown in Figure 1a, during training a standard MLL algorithm (big blue box) one typically attempts to fit the prediction function (small blue box) into feature vectors of documents and the corresponding tags. Note that feature vectors are generated *separately* before training, and tags for each document are encoded as a $|\mathcal{Y}|$-dimensional binary vector with one representing the presence and zero otherwise. In prediction phase, the learned prediction function will output relevant tags for the input feature vector of an unseen document. Following such a paradigm, many generic algorithms have been developed for MLL (Weston et al., 2011; Prabhu and Varma,

[*]This work was done when the author was an intern at Yahoo.

[1]tumblr.com

Proceedings of the 2nd Workshop on Representation Learning for NLP, pages 111–120,
Vancouver, Canada, August 3, 2017.



(a) standard multi-label learning

(b) proposed method (DocTag2Vec)

Figure 1: Comparison of standard multi-label learning framework and the proposed method

2014; Bhatia et al., 2015). With a surge of text content created by users online, such as blog posts, Wikipedia entries, etc., the algorithms for document tagging has many challenges. Firstly, time sensitive news articles are generated on a daily basis, and it is important for an algorithm to assign tags before they loose freshness. Secondly, new tagged documents could be fed into the training system, thus incrementally adapting the system to new training data without re-training from scratch is also critical. Thirdly, we might face a very large set of candidate tags that can change dynamically, as new things are being invented.

In view of the aforementioned challenges, in this paper we propose a new and simple approach for document tagging: DocTag2Vec. Our approach is motivated by the line of works on learning distributed representation of words and documents, e.g., Word2Vec (Mikolov et al., 2013) and Doc2Vec (a.k.a. Paragraph Vector) (Le and Mikolov, 2014). Word2Vec and Doc2Vec aim at learning low-dimensional feature vectors (i.e., embeddings) for words and documents from large corpus in an *unsupervised* manner, such that similarity between words (or documents) can be reflected by some distance metric on their embeddings. The general assumption behind Word2Vec and Doc2Vec is that more frequent co-occurrence of two words inside a small neighborhood of document should imply higher semantic similarity between them (see Section 2.2 for details). The DocTag2Vec extends this idea to document and tag by positing that document and its associated tags should share high semantic similarity, which allows us to learn the embeddings of tags along with documents (see Section 2.3 for details). Our method has two striking differences compared with standard MLL frameworks: firstly, our method directly works with raw text and *does not* need feature vectors extracted in advance. Secondly, our DocTag2Vec produces tag embeddings, which carry semantic information that are generally not available from standard MLL framework. During training, DocTag2Vec directly takes the raw documents and tags as input and learns their embeddings using *stochastic gradient descent* (SGD). In terms of prediction, a new document will be first embedded using a Doc2Vec component inside the DocTag2Vec, and tags are then assigned by searching for the nearest tags embedded around the document. Overall the proposed approach has the following merits.

- The SGD training supports the incremental adjustment of DocTag2Vec to new data.
- The prediction uses the simple k-nearest neighbor search among tags instead of documents, whose running time does not scale up as training data increase.
- Since our method represent each individual tag using its own embedding vector, it it easy to dynamically incorporate new tags.
- The output tag embeddings can be used in other applications.

Related Work: Multi-label learning has found several applications in social media and web, like sentiment and topic analysis (Huang et al., 2013a), social text stream analysis (Ren et al., 2014), and online advertising (Agrawal et al., 2013). MLL has also been applied to diverse Natural Language Processing (NLP) tasks. However to the best of our knowledge we are the first to propose embedding based MLL approach to a NLP task. MLL has been applied to Word Sense Disambiguation (WSD) problem for polysemic adjectives (Boleda et al., 2007). (Huang et al., 2013b) proposed a joint model to predict sentiment and topic for tweets and (Surdeanu et al., 2012) proposed a multi-instance MLL based approach for relation extraction with distant supervision.

Recently learning embeddings of words and sentences from large unannotated corpus has gained immense popularity in many NLP tasks, such as Named Entity Recognition (Passos et al., 2014; Lample et al., 2016; Ma and Hovy, 2016), sentiment classification (Socher et al., 2011; Tang

et al., 2014; dos Santos and Gatti, 2014) and summarization (Kaageback et al., 2014; Rush et al., 2015; Li et al., 2015). Also, vector space modeling has been applied to search re-targeting (Grbovic et al., 2015a) and query rewriting (Grbovic et al., 2015b).

Given many potential applications, document tagging has been a very active research area. In information retrieval, it is often coined as content-based *tag recommendation* problem (Chirita et al., 2007), for which numbers of approaches were proposed, such as (Heymann et al., 2008), (Song et al., 2008b), (Song et al., 2008a) and (Song et al., 2011). Personalized tag recommendation is also studied in the literature (Symeonidis et al., 2008; Rendle et al., 2009). In machine learning community, a lot of general MLL algorithms have been developed, with application to document tagging, including compressed-sensing based approach (Hsu et al., 2009), WSABIE (Weston et al., 2011), ML-CSSP (Bi and Kwok, 2013), LEML (Yu et al., 2014), FastXML (Prabhu and Varma, 2014), SLEEC (Bhatia et al., 2015) to name a few.

Paper Organization: The rest of the paper is organized as follows. In Section 2, we first give a brief review of Word2Vec and Doc2Vec models, and then present training and prediction step respectively for our proposed extension, DocTag2Vec. In Section 3, we demonstrate the effectiveness of our DocTag2Vec approach through experiments on several datasets. In the end, Section 4 is dedicated to conclusions and future works.

2 Proposed Approach

In this section, we present details of DocTag2Vec. For the ease of exposition, we first introduce some mathematical notations followed by a brief review for two widely-used embedding models: Word2Vec and Doc2Vec.

2.1 Notation

We let V be the size of vocabulary (i.e., set of unique words), N be the number of documents in the training set, M be the size of tag set, and K be the dimension of the vector space of embedding. We denote the vocabulary as $\mathcal{W} = \{w_1, \ldots, w_V\}$, set of documents as $\mathcal{D} = \{d_1, \ldots, d_N\}$, and the set of tags as $\mathcal{T} = \{t_1, \ldots, t_M\}$. Each document $d \in \mathcal{D}$ is basically a sequence of n_d words represented by $(w_1^d, w_2^d, \ldots, w_{n_d}^d)$, and is associated with M_d tags $\mathcal{T}_d = \{t_1^d, \ldots, t_{M_d}^d\}$. Here the subscript d of n and M suggests that the number of word and tag is different from document to document. For convenience, we use the shorthand $w_i^d : w_j^d$, $i \leq j$, to denote the subsequence of words $w_i^d, w_{i+1}^d, \ldots, w_{j-1}^d, w_j^d$ in document d. Correspondingly, we denote $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_V] \in \mathbb{R}^{K \times V}$ as the matrix for word embeddings, $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_N] \in \mathbb{R}^{K \times N}$ as the matrix for document embeddings, and $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_M] \in \mathbb{R}^{K \times M}$ as the matrix for tag embeddings. Sometimes we may use the symbol d_i interchangeably with the embedding vector $\mathbf{d}_i$ to refer to the i-th document, and use $\mathbf{d}_d$ to denote the vector representation of document d. Similar conventions apply to word and tag embeddings. Besides we let $\sigma(\cdot)$ be the sigmoid function, i.e., $\sigma(a) = 1/(1 + \exp(-a))$.

2.2 Word2Vec and Doc2Vec

The proposed approach is inspired by the work of Word2Vec, an unsupervised model for learning embedding of words. Essentially, Word2Vec embeds all words in the training corpus into a low-dimensional vector space, so that the semantic similarities between words can be reflected by some distance metric (e.g., cosine distance) defined on their vector representations. The way to train Word2Vec model is to minimize the loss function associated with certain classifier with respect to both *feature vectors* (i.e., word embeddings) and *classifier parameters*, such that the nearby words are able to predict each other. For example, in *continuous bag-of-word* (CBOW) framework, Word2Vec specifically minimizes the following average negative log probability

$$\sum_{d \in \mathcal{D}} \sum_{i=1}^{n_d} -\log p(w_i^d \mid w_{i-c}^d : w_{i-1}^d, w_{i+1}^d : w_{i+c}^d) \,,$$

where c is the size of context window inside which words are defined as "nearby". To ensure the conditional probability above is legitimate, one usually needs to evaluate a partition function, which may lead to a computationally prohibitive model when the vocabulary is large. A popular choice to bypass such issue is to use hierarchical softmax (HS) (Morin and Bengio, 2005), which factorizes the conditional probability into products of some simple terms. The hierarchical softmax relies on the construction of a binary tree $\mathcal{B}$ with V leaf nodes, each of which corresponds to a particular word in the vocabulary $\mathcal{W}$. HS is parameterized by a

matrix $\mathbf{H} \in \mathbb{R}^{K\times(V-1)}$, whose columns are respectively mapped to a unique non-leaf node of $\mathcal{B}$. Additionally, we define $\text{Path}(w) = \{(i,j) \in \mathcal{B} \,|\, \text{edge } (i,j) \text{ is on the path from root to word } w\}$. Then the negative log probability is given as

$$\begin{aligned}
&-\log p(w_i^d \mid w_{i-c}^d : w_{i-1}^d, w_{i+1}^d : w_{i+c}^d) \\
&= -\log \prod_{(u,v)\in \text{Path}(w_i^d)} \sigma\left(\text{child}(v)\cdot\langle \mathbf{g}_i^d, \mathbf{h}_v\rangle\right) \\
&= -\sum_{(u,v)\in \text{Path}(w_i^d)} \log\sigma\left(\text{child}(v)\cdot\langle \mathbf{g}_i^d, \mathbf{h}_v\rangle\right) \,, \\
&\mathbf{g}_i^d = \sum_{\substack{-c\le j\le c\\ j\neq 0}} \mathbf{w}_{i+j}^d \,,
\end{aligned}$$

where $\text{child}(u,v)$ is equal to 1 if v is the left child of u and 0 otherwise. Figure 2a shows the model architecture of CBOW Word2Vec. Basically $\mathbf{g}_i^d$ is the input feature for HS classifier corresponding to projection layer in Figure 2a , which essentially summarizes the feature vectors of context words surrounding w_i^d, and other options like averaging of $\mathbf{w}_{i+j}^d$ can also be applied. This Word2Vec model can be directly extended to Distributed memory (DM) Doc2Vec model by conditioning the probability of w_i^d on d as well as $w_{i-c}^d, \ldots, w_{i+c}^d$, which yields

$$\begin{aligned}
&-\log p(w_i^d \mid w_{i-c}^d : w_{i-1}^d, w_{i+1}^d : w_{i+c}^d, d) \\
&= -\sum_{(u,v)\in \text{Path}(w_i^d)} \log\sigma\left(\text{child}(v)\cdot\langle \tilde{\mathbf{g}}_i^d, \mathbf{h}_v\rangle\right) \,,
\end{aligned} \tag{1}$$

$$\tilde{\mathbf{g}}_i^d = \mathbf{d}_d + \sum_{\substack{-c\le j\le c\\ j\neq 0}} \mathbf{w}_{i+j}^d \,. \tag{2}$$

The architecture of DM Doc2Vec model is illustrated in Figure 2b. Instead of optimizing some rigorously defined probability function, both Word2Vec and Doc2Vec can be trained using other objectives, e.g., negative sampling (NEG) (Mikolov et al., 2013).

2.3 Training for DocTag2Vec

Our approach, DocTag2Vec, extends the DM Doc2Vec model by adding another component for learning tag embeddings. In addition to predicting target word w_i^d using context $w_{i-c}^d, \ldots, w_{i+c}^d$, as shown in Figure 2c, DocTag2Vec also uses the document embedding to predict each associated tag, with hope that they could be closely embedded. The joint objective is given by

$$\begin{aligned}
\sum_{d\in\mathcal{D}}\sum_{i=1}^{n_d}\Big(&-\log p(w_i^d \mid w_{i-c}^d : w_{i-1}^d, w_{i+1}^d : w_{i+c}^d, d) \\
&- \alpha \sum_{t\in\mathcal{T}_d} \log p(t \mid d)\Big) \,,
\end{aligned}$$

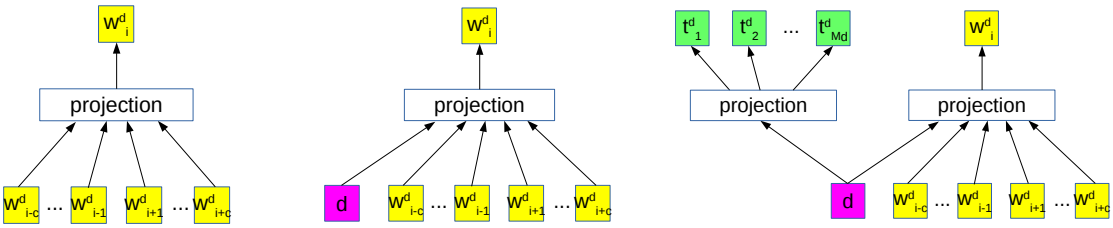
where α is a tuning parameter. As discussed for Word2Vec, the problem of evaluating costly partition function is also faced by the newly introduced probability $p(t|d)$. Different from the conditional probability of w_i^d, the probability $p(t|d)$ cannot be modeled using hierarchical softmax, as the columns of parameter matrix do not have one-to-one correspondence to tags (remember that we need to obtain a vector representation for each tag). Motivated by the idea of negative sampling used in Word2Vec, we come up with the following objective for learning tag embedding rather than stick to a proper probability function,

$$-\sum_{t\in\mathcal{T}_d} \log\sigma\left(\langle \mathbf{d}_t, \mathbf{t}_t\rangle\right) + r\cdot\mathbb{E}_{\mathbf{t}\sim p}\left[\log\sigma\left(-\langle \mathbf{d}_t, \mathbf{t}\rangle\right)\right] \,, \tag{3}$$

where p is a discrete distribution over all tag embeddings $\{\mathbf{t}_1, \ldots, \mathbf{t}_M\}$ and r is a integer-valued hyperparameter. The goal of such objective is to differentiate the tag t from the draws according to p, which is chosen as *uniform distribution* for simplicity in our practice. Now the final loss function for DocTag2Vec is the combination of (1) and (3),

$$\begin{aligned}
&\ell\left(\mathbf{W}, \mathbf{D}, \mathbf{T}, \mathbf{H}\right) = \sum_{d\in\mathcal{D}}\sum_{i=1}^{n_d} \\
&\Big(\underbrace{-\sum_{(u,v)\in \text{Path}(w_i^d)} \log\sigma\big(\text{child}(v)\cdot\langle \tilde{\mathbf{g}}_i^d, \mathbf{h}_v\rangle\big)}_{\text{DM Doc2Vec with hierarchical softmax}} - \\
&\underbrace{\alpha\sum_{t\in\mathcal{T}_d} \log\sigma\left(\langle \mathbf{d}_t, \mathbf{t}_t\rangle\right) + r\cdot\mathbb{E}\left[\log\sigma\left(-\langle \mathbf{d}_t, \mathbf{t}\rangle\right)\right]}_{\text{tag embedding with negative sampling}}\Big)
\end{aligned} \tag{4}$$

We minimize $\ell\left(\mathbf{W}, \mathbf{D}, \mathbf{T}, \mathbf{H}\right)$ using stochastic gradient descent (SGD). To avoid exact calculation of the expectation in negative sampling, at each iteration we sample r i.i.d. instances of $\mathbf{t}$ from distribution p, denoted by $\{\mathbf{t}_p^1, \mathbf{t}_p^2, \ldots, \mathbf{t}_p^r\}$, to stochastically approximate the expectation, i.e., $\sum_{j=1}^r \log\sigma(-\langle \mathbf{d}_t, \mathbf{t}_p^j\rangle) \approx r\cdot\mathbb{E}\left[\log\sigma\left(-\langle \mathbf{d}_t, \mathbf{t}\rangle\right)\right]$.



(a) CBOW Word2Vec (b) DM Doc2Vec (c) DocTag2Vec

Figure 2: Model architectures of different embedding approaches

2.4 Prediction for DocTag2Vec

Unlike Word2Vec and Doc2Vec, which only target on learning high-quality embeddings of words and documents, DocTag2Vec needs to make predictions of relevant tags for new documents. To this end, we first embed the new document via the Doc2Vec component within DocTag2Vec and then perform k-nearest neighbor (k-NN) search among tags. To be specific, given a new document d, we first optimize the objective (1) with respect to $\mathbf{d}_d$ by fixing $\mathbf{W}$ and $\mathbf{H}$. Note that this is the standard inference step for new document embedding in Doc2Vec. Once $\mathbf{d}_d$ is obtained, we search for the k-nearest tags to it based on cosine similarity. Hence the prediction function is given as

$$f_k(\mathbf{d}_d) = \left\{ i \,|u_i \text{ is in the largest } k \text{ entries of } \mathbf{u} = \overline{\mathbf{T}}^T \overline{\mathbf{d}}_d \right\}, \quad (5)$$

where $\overline{\mathbf{T}}$ is column-normalized version of $\mathbf{T}$. To boost the prediction performance of DocTag2Vec, we apply the *bootstrap aggregation* (a.k.a. bagging) technique to DocTag2Vec. Essentially we train b DocTag2Vec learners using different randomly sampled subset of training data, resulting in b different tag predictors $f^1_{k'}(\cdot), \ldots, f^b_{k'}(\cdot)$ along with their tag embedding matrices $\mathbf{T}^1, \ldots, \mathbf{T}^b$. In general, the number of nearest neighbors k' for individual learner can be different from k. In the end, we combine the predictions from different models by selecting from $\bigcup_{j=1}^b f^j_{k'}(\mathbf{d}_d)$ the k tags with the largest aggregated similarities with $\mathbf{d}_d$,

$$f^{bag}_k(\mathbf{d}_d) = \left\{ i \,|u_i \text{ is in the largest } k \text{ entries of } \mathbf{u}, \text{ where } u_i = \sum_{j=1}^{b} \mathbb{I}\{i \in f^j_{k'}(\mathbf{d}_d)\} \cdot \langle \overline{\mathbf{t}}^j_i, \overline{\mathbf{d}}_d \rangle \right\}.$$

3 Experiments

3.1 Datasets

In this subsection, we briefly describe the datasets included in our experiments. It is worth noting that DocTag2Vec method needs raw texts as input instead of extracted features. Therefore many benchmark datasets for evaluating multi-label learning algorithms are not suitable for our setting. For the experiment, we primarily focus on the diversity of the source of tags, which capture different aspects of documents. The statistics of all datasets are provided in Table 1.

Public datasets:

- **Wiki10**: The wiki10 dataset contains a subset of English Wikipedia documents, which are tagged collaboratively by users from the social bookmarking site *Delicious*[1]. We remove the two uninformative tags, "wikipedia" and "wiki", from the collected data.
- **WikiNER**: WikiNER has a larger set of English Wikipedia documents. The tags for each document are the named entities inside it, which is detected automatically by some named entity recognition (NER) algorithm.

Proprietary datasets

- **Relevance Modeling (RM)**: The RM dataset consists of two sets of financial news article in Chinese and Korean respectively. Each article is tagged with related ticker symbols of companies given by editorial judgement.
- **News Content Taxonomy (NCT)**: NCT dataset is a collection of news articles annotated by editors with topical tags from a taxonomy tree. The closer the tag is to the root, the more general the topic is. For such tags with hierarchical structure, we also evaluate our method separately for tags of general topics (depth=2) and specific topics (depth=3).

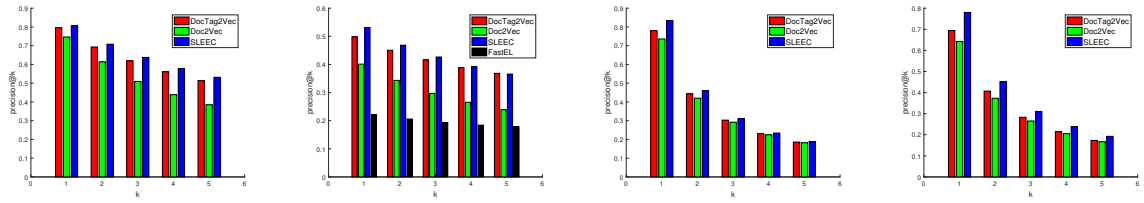
3.2 Baselines and Hyperparameter Setting

The baselines include one of the state-of-the-art multi-label learning algorithms called SLEEC

[1] https://del.icio.us/

Dataset	#training point	#testing point	#unique tags	Avg #tags per document
Wiki10	14095	6600	31177	17.27
WikiNER	89521	10000	67179	22.96
Relevance Modeling (Chinese)	4505	500	391	1.02
Relevance Modeling (Korean)	1292	500	261	1.07
NCT (all)	40305	9546	883	1.88
NCT (general)	39401	9389	294	1.76
NCT (specific)	17278	4509	412	1.41

Table 1: Statistics of datasets



(a) Wiki10 (b) WikiNER (c) RM (Chinese) (d) RM (Korean)

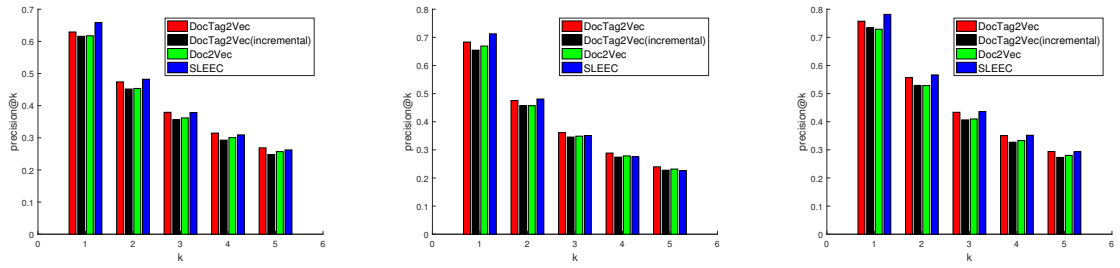
Figure 3: Precision on Wiki10, WikiNER and Relevance Modeling dataset

(Bhatia et al., 2015), a variant of DM Doc2Vec, and an unsupervised entity linking system, FastEL (Blanco et al., 2015), which is specific to WikiNER dataset. SLEEC is based on non-linear dimensionality reduction of binary tag vectors, and use a sophisticated objective function to learn the prediction function. For comparison, we use the TF-IDF representation of document as the input feature vector for SLEEC, as it yields better result than embedding based features like Doc2Vec feature. To extend DM Doc2Vec for tagging purpose, basically we replace the document d shown in Figure 2b with tags $t^d_1, \ldots, t^d_{M_d}$, and train the Doc2Vec to obtain the tag embeddings. During testing, we perform the same steps as DocTag2Vec to predict the tags, i.e., inferring the embedding of test document followed by k-NN search. FastEL is unsupervised appproach for entity linking of web-search queries that walks over a sequence of words in query and aims to maximize the likelihood of linking the text span to an entity in Wikipedia. FastEL model calculates the conditional probabilities of an entity given every substring of the input document, however avoid computing entit to entity joint dependencies, thus making the process efficient. We built FastEL model using query logs that spanned 12 months and Wikipedia anchor text extracted from Wikipedia dumps dated November 2015. We choose an entity linker baseline because it is a simple way of detecting topics/entities that are semantically associated with a document.

Regarding hyperparameter setting, both SLEEC and DocTag2Vec aggregate multiple learners to enhance the prediction accuracy, and we set the number of learners to be 15. For SLEEC, we tune the rest of hyperparameters using grid search. For SLEEC and DocTag2Vec, we set the number of epochs for SGD to be 20 and the window size c to be 8. To train each individual learner, we randomly sample 50% training data. In terms of the nearest neighbor search, we set $k' = 10$ for Wiki10 and WikiNER while keeping $k' = 5$ for others. For the rest of hyperparameters, we also apply grid search to find the best ones. For DocTag2Vec, we additionally need to set the number of negative tags r and the weight α in (4). Typically r ranges from 1 to 5, and $r = 1$ gives the best performance on RM and NCT datasets. Empirically good choice for α is between 0.5 and 5. For FastEL, we consider a sliding window of size 5 over the raw-text (no punctuations) of document to generate entity candidates. We limit the number of candidates per document to 50.

3.3 Results

We use $precision@k$ as the evaluation metric for the performance. Figure 3 shows the precision plot of different approaches against choices of k on Wiki10, WikiNER and RM dataset. On Wiki10, we see that the precision of our DocTag2Vec is close to the one delivered by SLEEC, while Doc2Vec performs much worse. We observe similar result on WikiNER except for the precision@1, but our precision catches up as k increases. For RM dataset, SLEEC outperforms our approach, and we conjecture that such gap is due to the small size of training data, from which DocTag2Vec is not able to learn good embeddings. It



(a) All tags (b) Specific tags (c) General tags

Figure 4: Precision on News Content Taxonomy dataset

	DocTag2Vec	DocTag2Vec (incremental)	Doc2Vec	SLEEC
NCT (all tags)	0.6702	0.6173	0.6389	0.6524
NCT (specific tags)	0.8111	0.7678	0.7810	0.7624
NCT (general tags)	0.7911	0.7328	0.7521	0.7810

Table 2: Overall Recall on News Content Taxonomy dataset

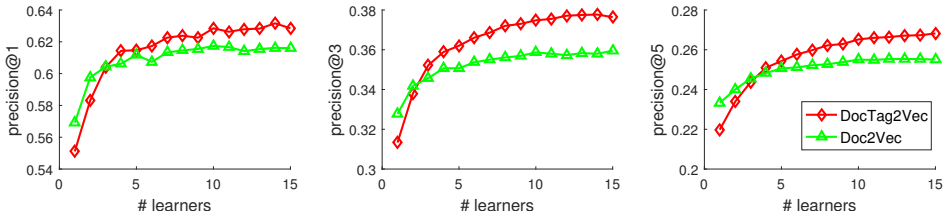


Figure 5: Precision vs. number of learners on NCT dataset

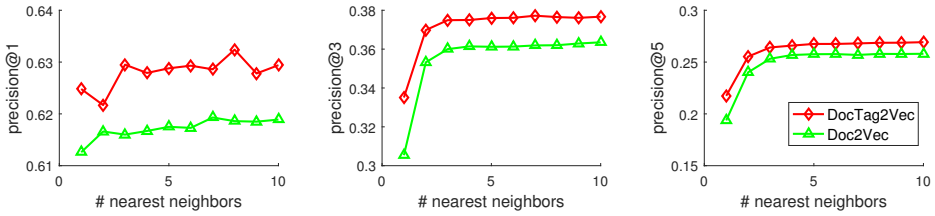


Figure 6: Precision vs. number of nearest neighbors on NCT dataset

News excerpt	Editorial tags	Prediction (top 3)	
		Predicted tags	similarity
The world is definitely getting warmer, according to the U.S. National Atmospheric and Oceanic Administration. For its annual "State of the Climate" report, NOAA for the first time gathered data on 37 climate indicators, such as air and sea temperatures, sea level, humidity, and snow cover in one place, and found that, taken together, the measurements show an "unmistakable upward trend" in temperature. Three hundred scientists analyzed the information and concluded it's "undeniable" that the planet has warmed since 1980, with the last decade taking the record for hottest ever recorded.	*/Nature & Environment/ Natural Phenomena*	*/Nature & Environment/ Environment/Climate Change*	1.99
		/Science/ Meteorology	0.64
		/Nature & Environment/Natural Phenomena/Weather	0.57
Business software maker Epicor Software Corp. said Thursday that its second-quarter loss narrowed as revenue climbed. For the April-June quarter, Epicor's loss totaled $1 million, or 2 cents per share, compared with a loss of $6.7 million, or 11 cents per share, in the year-ago quarter. When excluding one-time items, Epicor earned 13 cents per share, which is what analysts polled by Thomson Reuters expected. Revenue rose 9 percent to $109.2 million, beating analyst estimates for $105.2 million.	*/Business/Sectors & Industries/ Information Technology/Internet Software & Services* */Finance/Investment & Company Information*	*/Finance/Investment & Company Information/ Company Earnings*	2.25
		/Finance/Investment & Company Information	1.23
		/Finance/Investment & Company Information/ Stocks & Offerings	0.32
TicketLiquidator, the leading provider of the world's most extensive ticket inventory for hard-to-find, low priced tickets, today announced that tickets are available for the Orlando Magic vs. Cleveland Cavaliers game on Wednesday, November 11th at Orlando's Amway Arena. The much-anticipated matchup features LeBron James, who is now in the final year of his contract with the Cavaliers.	*/Sports & Recreation/ Baseball*	*/Sports & Recreation/Basketball*	4.07
		/Arts & Entertainment/Events/Tickets	3.42
		/Sports & Recreation/Baseball	0.96

Table 3: Examples of Better Prediction over Editorial Judgement

is to be noted that SLEEC requires proper features as input and does not work directly with raw documents; while DocTag2Vec learns vector representation of documents that are not only useful for multilabel learning but also as features for other tasks like sentiment analysis, hate speech detec-

tion, and content based recommendation. We have demonstrated improvements in all the above mentioned use-cases of DocTag2Vec vectors but the discussion on those is out of the scope of this paper.

For NCT dataset, we also train the DocTag2Vec incrementally, i.e., each time we only feed 100 documents to DocTag2Vec and let it run SGD, and we keep doing so until all training samples are presented. As shown in Figure 4, our DocTag2Vec outperform Doc2Vec baseline, and delivers competitive or even better precision in comparision with SLEEC. Also, the incremental training does not sacrifice too much precision, which makes DocTag2Vec even appealing. The overall recall of DocTag2Vec is also slightly better than SLEEC, as shown in Table 2. Figure 5 and 6 include the precision plot against the number of learners b and the number of nearest neighbors k' for individual learner, respectively. It is not difficult to see that after $b = 10$, adding more learners does not give significant improvement on precision. For nearest neighbor search, $k' = 5$ would suffice.

3.4 Case Study for NCT dataset

For NCT dataset, when we examine the prediction for individual articles, it turns out surprisingly that there are a significant number of cases where DocTag2Vec outputs better tags than those by editorial judgement. Among all these cases, we include a few in Table 3 showing the superiority of the tags given by DocTag2Vec sometimes. For the first article, we can see that the three predicted tags are all related to the topic, especially the one with highest similarity, */Nature & Environment/ Environment/Climate Change*, seems more pertinent compared with the editor's. Similarly, we predict */Finance/Investment & Company Information/Company Earnings* as the most relevant topic for the second article, which is more precise than its parent */Finance/Investment & Company Information*. Besides our approach can even find the wrong tags assigned by the editor. The last piece of news is apparently about NBA, which should have the tag */Sports & Recreation/Basketball* as predicted, while the editor annotates them with the incorrect one, */Sports & Recreation/Baseball*. On the other hand, by looking at the similarity scores associated with the predicted tags, we can see that higher score in general implies higher aboutness, which can also be used as a quantification of prediction confidence.

4 Conclusions and Future Work

In this paper, we present a simple method for document tagging based on the popular distributional representation learning models, Word2Vec and Doc2Vec. Compared with classical multi-label learning methods, our approach provides several benefits, such as allowing incremental update of model, handling the dynamical change of tag set, as well as producing feature representation for tags. The document tagging can benefit a number of applications on social media. If the text content over web is correctly tagged, articles or blog posts can be pushed to the right users who are likely to be interested. And such good personalization will potentially improve the users engagement. In future, we consider extending our approach in a few directions. Given that tagged documents are often costly to obtain, it would be interesting to extend our approach to a semi-supervised setting, where we can incorporate large amounts of unannotated documents to enhance our model. On the other hand, with the recent progress in graph embedding for social network (Yang et al., 2016; Grover and Leskovec, 2016), we may be able to improve the tag embedding by exploiting the representation of users and interactions between tags and users on social networks.

Acknowledgements

The authors would like to thank the anonymous reviewers for their encouraging and thoughtful comments.

References

R. Agrawal, A. Gupta, Y. Prabhu, and M. Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22Nd International Conference on World Wide Web*. WWW '13.

K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain. 2015. Sparse local embeddings for extreme multi-label classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*. pages 730–738.

W. Bi and J. Kwok. 2013. Efficient multi-label classification with many labels. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*. pages 405–413.

Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, pages 179–188.

Gemma Boleda, Sabine Schulte im Walde, and Toni Badia. 2007. Modelling polysemy in adjective classes by multi-label classification. In *EMNLP-CoNLL*. pages 171–180.

Axel Bruns and Jean E Burgess. 2011. The use of twitter hashtags in the formation of ad hoc publics. In *Proceedings of the 6th European Consortium for Political Research (ECPR) General Conference 2011*.

Yi Chang, Lei Tang, Yoshiyuki Inagaki, and Yan Liu. 2014. What is tumblr: A statistical overview and comparison. *ACM SIGKDD Explorations Newsletter* 16(1):21–29.

P.-A. Chirita, S. Costache, W. Nejdl, and S. Handschuh. 2007. P-tag: Large scale automatic generation of personalized annotation tags for the web. In *Proceedings of the 16th International Conference on World Wide Web*. pages 845–854.

Cícero Nogueira dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*. pages 69–78.

M. Grbovic, N. Djuric, V. Radosavljevic, and N. Bhamidipati. 2015a. Search retargeting using directed query embeddings. In *Proceedings of the 24th International Conference on World Wide Web*. pages 37–38.

M. Grbovic, N. Djuric, V. Radosavljevic, F. Silvestri, and N. Bhamidipati. 2015b. Context- and content-aware embeddings for query rewriting in sponsored search. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 383–392.

A. Grover and J. Leskovec. 2016. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16.

P. Heymann, D. Ramage, and H. Garcia-Molina. 2008. Social tag prediction. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 531–538.

D. J Hsu, S. M Kakade, J. Langford, and T. Zhang. 2009. Multi-label prediction via compressed sensing. In *Advances in Neural Information Processing Systems 22*, pages 772–780.

S. Huang, W. Peng, J. Li, and D. Lee. 2013a. Sentiment and topic analysis on social media: A multi-task multi-label classification approach. In *Proceedings of the 5th Annual ACM Web Science Conference*. WebSci '13.

Shu Huang, Wei Peng, Jingxuan Li, and Dongwon Lee. 2013b. Sentiment and topic analysis on social media: a multi-task multi-label classification approach. In *Proceedings of the 5th annual acm web science conference*. ACM, pages 172–181.

Mikael Kaageback, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC) EACL*. pages 31–39.

Guillaume Lample et al. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360* .

Q. V. Le and T. Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*. volume 14, pages 1188–1196.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057* .

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354* .

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *AISTATS05*. pages 246–252.

Ruth Page. 2012. The linguistics of self-branding and micro-celebrity in twitter: The role of hashtags. *Discourse & Communication* 6(2):181–201.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. *arXiv preprint arXiv:1404.5367* .

Y. Prabhu and M. Varma. 2014. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pages 263–272.

Z. Ren, M.-H. Peetz, S. Liang, W. van Dolen, and M. de Rijke. 2014. Hierarchical multi-label classification of social text streams. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*. SIGIR '14.

S. Rendle, L. Balby Marinho, A. Nanopoulos, and L. Schmidt-Thieme. 2009. Learning optimal ranking with tensor factorization for tag recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pages 727–736.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* .

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 151–161.

Y. Song, L. Zhang, and C. L. Giles. 2008a. A sparse gaussian processes classification framework for fast tag suggestions. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. pages 93–102.

Y. Song, L. Zhang, and C. L. Giles. 2011. Automatic tag recommendation algorithms for social recommender systems. *ACM Trans. Web* 5(1):4:1–4:31.

Y. Song, Z. Zhuang, H. Li, Q. Zhao, J. Li, W.-C. Lee, and C. L. Giles. 2008b. Real-time automatic tag recommendation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 515–522.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, pages 455–465.

P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. 2008. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM Conference on Recommender Systems*. pages 43–50.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL (1)*. pages 1555–1565.

J. Weston, S. Bengio, and N. Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*. pages 2764–2770.

Z. Yang, W. W. Cohen, and R. Salakhutdinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*.

H.-F. Yu, P. Jain, P. Kar, and I. S. Dhillon. 2014. Large-scale multi-label learning with missing labels. In *International Conference on Machine Learning (ICML)*. volume 32.