An Integral Tag Recommendation Model for Textual Content

Shijie Tang,[1] Yuan Yao,[1] Suwei Zhang,[1] Feng Xu,[1]
Tianxiao Gu,[2] Hanghang Tong,[3] Xiaohui Yan,[4] Jian Lu[1]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]University of California, Davis, USA, [3]Arizona State University, USA, [4]Poisson Lab, Huawei Technologies, China
{tsj, zsw}@smail.nju.edu.cn, {y.yao, xf, lj}@nju.edu.cn,
tianxiao.gu@gmail.com, hanghang.tong@asu.edu, yanxiaohui2@huawei.com

Abstract

Recommending suitable tags for online textual content is a key building block for better content organization and consumption. In this paper, we identify three pillars that impact the accuracy of tag recommendation: (1) *sequential text modeling* meaning that the intrinsic sequential ordering as well as different areas of text might have an important implication on the corresponding tag(s) , (2) *tag correlation* meaning that the tags for a certain piece of textual content are often semantically correlated with each other, and (3) *content-tag overlapping* meaning that the vocabularies of content and tags are overlapped. However, none of the existing methods consider all these three aspects, leading to a suboptimal tag recommendation. In this paper, we propose an integral model to encode all the three aspects in a coherent encoder-decoder framework. In particular, (1) the encoder models the semantics of the textual content via Recurrent Neural Networks with the attention mechanism, (2) the decoder tackles the tag correlation with a prediction path, and (3) a shared embedding layer and an indicator function across encoder-decoder address the content-tag overlapping. Experimental results on three real-world datasets demonstrate that the proposed method significantly outperforms the existing methods in terms of recommendation accuracy.

Introduction

Tagging has been widely viewed as a very successful practice of associating metadata with online content, because online content can be better organized and consumed with the help of tags (Belém, Almeida, and Gonçalves 2017). Consequently, recommending suitable tags for online content becomes an important task. From the perspective of content creators, tag recommendation allows them to select tags from a ranking list, which in turn improves not only the user experience but also the quality of the chosen tags. From the perspective of content consumers, tag recommendation helps provide better services for the consumers' search and retrieval requests. For example, tags have been shown to be effective in a variety of tasks including user interest discovery (Li, Guo, and Zhao 2008) and content recommendation (Guy et al. 2010).

Despite its importance and extensive effort, recommending suitable tags for online content remains a very challenging task. In this work, we tackle the tag recommendation task for textual content. Let us first identify the following three key pillars underlying this task (Please see Fig. 1 for an illustrative example).

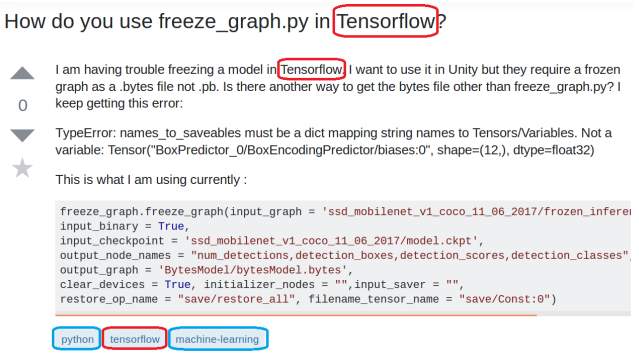


Figure 1: An illustrative example of content-tag overlapping and tag correlation. The word 'tensorflow' appears in both the content and tags (circled with red rectangles); the three tags (circled with blue rectangles) are semantically related to each other (i.e., both 'python' and 'machine-learning' are closely related to 'tensorflow').

- *(C1) Sequential Text Modeling*. Capturing the semantics of textual content is an essential step towards recommending better tags. Existing tag recommendation methods (e.g., (Krestel, Fankhauser, and Nejdl 2009; Ramage et al. 2009; Wu et al. 2016)) mainly adopt topic models to understand the textual content. However, topic models treat textual content as a bag of words, and thus ignore the sequential nature of sentences.
- *(C2) Tag Correlation*. Intuitively, the tags for a certain piece of content are often semantically correlated, which, if modelled appropriately, can be harnessed to improve the tag recommendation performance. For the example in Fig. 1, the three tags ('python', 'tensorflow', and 'machine-learning') are strongly correlated as they are all closely related to the tensorflow framework. Nonetheless, it has largely remained unknown on how to model tag correlation in the context of tag recommendation, with the only exception of (Wang et al. 2016).

- *(C3) Content-Tag Overlapping*. In general, the tag vocabulary is often overlapped with the content vocabulary (e.g., many tags may have appeared in the content). Take Fig. 1 as an example. The tag 'tensorflow' has appeared in both the title and the body of the content. However, such a content-tag overlapping phenomenon is largely ignored by existing work. For example, existing tag recommendation methods (Guo et al. 2004; Wang and Wang 2007) treating the problem as a supervised learning task would lead to the separation of the feature space (i.e., content) and the label space (i.e., tags).

Intuitively, each of the above three aspects plays a mutually complementary role in recommending tags for textual content. However, there lacks an integral method that captures all the three aspects within a coherent model. For example, Wu et al. (2016) explicitly considers the content-tag overlapping phenomenon; however, they ignore the sequential nature of sentences and the tag correlation phenomenon. Likewise, Wang et al. (2016) model the tag correlation phenomenon; however, they ignore the content-tag overlapping as the model is specially designed for image classification.

In this paper, we propose an integral model to encode all the three aspects in a coherent encoder-decoder framework. In detail, We first model textual content with a multi-layer RNN to capture its sequential nature in the encoder, and augment it with the attention mechanism to learn the text areas that are more closely related to each of the recommended tag. This naturally provides an intuitive interpretation on the recommendation results. Second, we model tag correlation with a prediction path in the decoder part. That is, when predicting the current tag, the input includes not only the encoder output but also the previous tags as a prior. This enables us to collectively predict all the tags for a given textual content. Third, based on the encoder-decoder structure, we model content-tag overlapping with a shared embedding layer and an indicator function. The shared embedding layer brings the textual content and tags in the same space by assigning the same embedding to the same word, and the indicator function learns the probability of directly copying a word from the content as the predicted tag.

In summary, the main contributions of this paper include:

- An integral model ITAG to recommend tags for textual content. The proposed ITAG simultaneously models the three important aspects (i.e., sequential text modeling, tag correlation, and content-tag overlapping).

- Extensive experimental evaluations on three real datasets showing the superior performance of the proposed ITAG. For example, ITAG can lead up to 23.1% relative improvement compared with its best competitors.

The rest of the paper is organized as follows. Section 2 provides the problem statement and summarizes the existing work in terms of the three pillars. Section 3 describes the proposed approach, and Section 4 presents the experimental evaluations. Section 5 reviews the related work, and Section 6 concludes.

Problem Statement

We consider the tag recommendation problem for textual content. The input of the training stage includes a collection of documents, each of which consists of a piece of textual content (e.g., several sentences containing a list of words) and several tags related to the content. In the testing stage, we aim to recommend one or more tags for a new document which only contains a piece of textual content (i.e., without any known tags).

To tackle the above tag recommendation problem, we identify the following three pillars that have a significant impact on the recommendation accuracy. The first pillar lies in textual content modeling. Instead of using TF-IDF or topic models to extract the text features, we resort to RNNs to model the sequential orders of words. We further introduce attention mechanism to learn different weights of the input words. This is crucial as it renders the interpretability of the proposed method by flagging the important text segment w.r.t. the recommended tags. The second pillar is the tag correlation phenomenon. This means that the tags for a certain piece of content are often semantically correlated, e.g., all the tags should be coherent with the meaning of the given text. This suggests that a *collective* recommendation might bring extra performance improvement. The third pillar is the content-tag overlapping, i.e., many words have appeared as both tags and content words, and the same word usually has the same meaning in both the content space and the tag space. Therefore, directly using a traditional multi-label learning framework to model the tag recommendation problem is suboptimal since the same word in content and tags is mapped to two different spaces. In this work, we aim to integrate the above three pillars into a unified model for textual tag recommendation.

We summarize and compare the existing work and our proposed method in terms of the three pillars in Table 1. As we can see, Tagspace (Weston, Chopra, and Adams 2014), Maxide (Xu, Jin, and Zhou 2013), and ConTagNet (Rawat and Kankanhalli 2016) consider none of the three pillars. The other methods consider only one of the three pillars. For example, Tag2Word (Wu et al. 2016) handles content-tag overlapping under the context of topic models, CNN-RNN (Wang et al. 2016) models the tag correlation for image classification, ABC (Gong and Zhang 2016) models the sequential nature of text with CNN, and TLSTM (Li et al. 2016b) and PBAM (Sun et al. 2018) adopt RNN to model the text. In contrast to these methods, the proposed ITAG takes all the three pillars into consideration.

The Proposed Approach

In this section, we present the proposed ITAG model. We start with the overall framework of the proposed model, and then describe each of its key components.

Overall Framework

The proposed ITAG is built upon the encoder-decoder framework (See Fig. 2 for an illustration). The encoder section takes a text sequence as the input, and outputs the vector representations for the text sequence. Specifically, we use

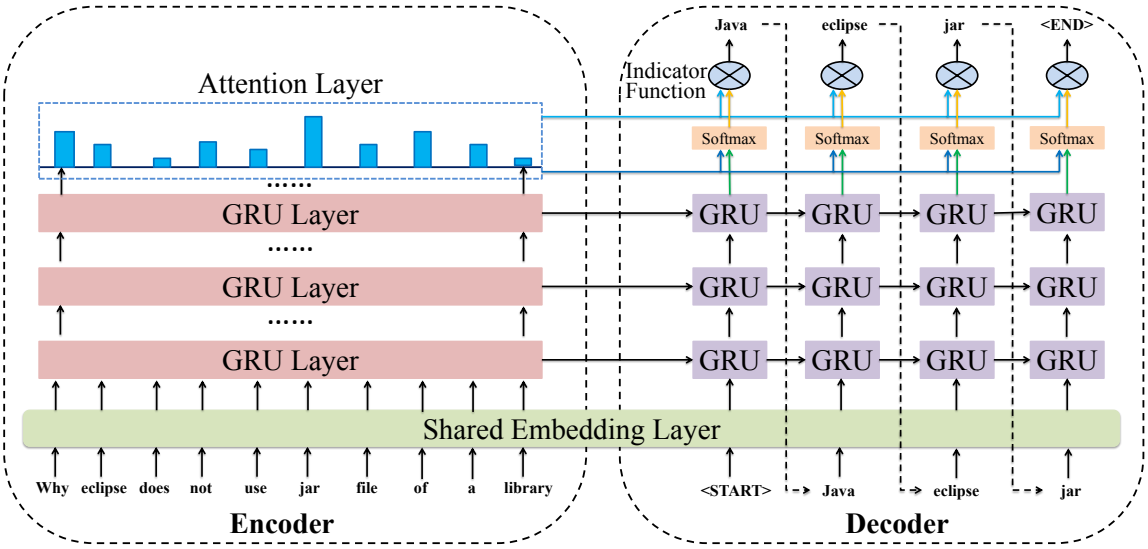


Figure 2: The overall framework of the proposed ITAG model.

a multi-layer RNN structure with attention mechanism to model the textual content. For the multiple RNN layers, we build the attention mechanism upon the top layer to bring the interpretability to the recommendation results.

In the decoder section, it takes as the input the vector representations from the encoder section, and outputs the tags for the corresponding input text. In particular, we connect several RNNs as a prediction path to characterize the tag correlation. That is, the output of the previous RNN serves as the additional input of the next RNN. For example, we utilize the first RNN with the output from the encoder to predict the first tag 'java'. After that, we utilize the second RNN with the output of the first RNN and the first tag 'java' to predict the second tag 'eclipse'.

Finally, we design a shared embedding layer and an indicator function that stretch across the encoder and the decoder to capture the content-tag overlapping. For example, the word 'eclipse' uses the same embedding in the encoder and the decoder, and we learn an indicator probability to determine whether to directly assign 'eclipse' as a tag.

The overall training objective function is shown below,

$$J = \frac{1}{|T|} \sum_{(t,z)\in T} \sum_{z_i \in z} -\log P(z_i|t), \tag{1}$$

where T is the training set, t and z are the textual content and its corresponding tag set of each input, z_i is the current tag under prediction, and $P(z_i|t)$ is the probability of choosing tag z_i for input text t.

Modeling Textual Content

A unique characteristic of RNNs lies in that it stores information circularly by iterating functions. As such, it is particularly suitable for processing sequential input (Li et al. 2016a; Sun et al. 2018). In our model, we utilize a multi-layer RNN structure (i.e., the grid-structure RNN) which includes l layers of RNNs. The bottom layer takes the text sequence as its input and generates a sequence of hidden states accordingly. Next, these hidden states are fed into the next layer which in turn outputs the hidden states for the next layer. Finally, the hidden states of the top layer serve as the final vector representations for the input text sequence.

As to the recurrent neuron for RNNs, we use the Gated Recurrent Unit (GRU) (Cho et al. 2014). Basically, GRU introduces the update gate to control the extent to which the state information of the previous moment is kept in the current state, and the reset gate to control the degree of ignoring the state information of the previous moment. We denote it as

$$h_i = GRU(x_i, h_{i-1}), \tag{2}$$

where h_i is the current hidden state (output vector), and x_i is the current input of the GRU neuron. We omit the detailed equations for GRU for brevity (see (Cho et al. 2014) for details).

We further introduce the attention mechanism (Bahdanau, Cho, and Bengio 2014) on the top layer of the multi-layer RNNs, which weights and aggregates each GRU state for

generating each tag. For example, given an input sequence of N words $(1, 2, ..., N)$ whose hidden states in the top layer are $(h_1, h_2, ..., h_N)$. For the i-th tag whose hidden state/feature is s_i, we calculate its new feature vector s'_i with attention as follows,

$$\begin{aligned} u_{ij} &= v_1^T \cdot tanh(W_1 \cdot h_j + W_2 \cdot s_i) \\ \alpha_{ij} &= softmax(u_{ij}) \\ s'_i &= \sum_{j=1}^{N} \alpha_{ij} \cdot h_j \end{aligned} \quad (3)$$

where vector v_1 and matrices W_1 and W_2 are parameters. In the above equations, the normalized α_{ij} determines how much attention should be given to the j-th hidden state of the encoder (i.e., h_j) for the new i-th hidden state of the decoder (i.e., s'_i). Finally, we can either concatenate s'_i with s_i or add them together to generate the overall feature vector for predicting the i-th tag. Our empirical evaluations indicate that there is little difference between these two choices in terms of recommendation accuracy. For the results we report in this paper, we add s'_i and s_i together for simplicity.

Modeling Tag Correlation

The vast majority of the existing approaches for tag recommendation typically model the problem as a multi-label classification problem. Although performing reasonably well, one major limitation lies in the fact that the tags for a certain piece of content are usually strongly correlated with each other. To address this limitation, we utilize RNNs to build a prediction path in the decoder section. That is, the output states of the previous RNN as well as the previous tags will be used in the current RNN to predict the current tag.

Take Fig. 2 as an example. At the beginning, the 'start' symbol[1] is fed into the first RNN. Along with the output vectors from the encoder section, we can predict the first tag as the output of this specific RNN. Next, we feed the output states of the first RNN as well as the first tag into the second RNN, and predict the second tag. We repeat such process until the 'end' symbol is predicted.

Inference Stage. In the training stage, we have the ground truth of previous tags. The situation is different in the inference stage. As mentioned above, we organize all the tags (e.g., $z_1, z_2, z_3, ..., z_M$) for a certain input sequence as a prediction path, i.e.,

$$\begin{aligned} \arg\max_{z_1,...,z_M} P(z_1, ..., z_M|I) = \arg\max_{z_1,...,z_M} P(z_1|I) \times \\ P(z_2|I, z_1) \times ... \times P(z_M|I, z_1, ..., z_{M-1}) \end{aligned} \quad (4)$$

where I is the output of the encoder section, and $P(z_i|I, z_1, ..., z_{i-1})$ indicates that the prediction of the current tag is dependent on I as well as the previous tags. However, the tags are predicted one by one, and the error of previous tag predictions will accumulate along the prediction path. Moreover, there is no polynomial-time algorithm for finding the optimal $P(z_i|I, z_1, ..., z_{i-1})$. To mitigate this problem, we employ the beam search algorithm to obtain multiple candidate prediction paths. After that, we can directly select the prediction path with the highest probability as the final result. However, a subtle issue of this strategy is that it would favor the shorter paths as they tend to have higher probabilities. To address this issue, we introduce a penalty into the probability computation by multiplying the probability with the length of the current prediction path.

[1]We use a 'start' symbol as the initial input for the first tag, and the prediction ends when an 'end' symbol is predicted.

Modeling Content-Tag Overlapping

In textual tag recommendation problem, the decoder vocabulary is usually overlapped with the encoder vocabulary. In addition, the same word in the encoder and the decoder should share the same meaning. We design a shared embedding layer and an indicator function that span across the encoder and the decoder in order to model such phenomenon.

Shared Embedding. The shared embedding layer maps the same word in the encoder section and that in the decoder section to the same embedding. Specifically, given a word k (the k-th word), we first represent it as a one-hot vector $e_k = [0, ...0, 1, 0, ..., 0]^T$ where the k-th position is 1 and the other positions are 0s; then, we multiply the one-hot vector with an embedding matrix E in the shared embedding layer, i.e.,

$$\omega_k = E \cdot e_k \quad (5)$$

where ω_k is the low-dimensional embedding for word k. In other words, the embedding of the k-th word corresponds to the k-th column of the embedding matrix E. In the above equation, E is a $d_x \times |V|$ matrix for both the content words and the tag words, where d_x is the embedding dimensionality for input words and $|V|$ is the total vocabulary size.[2]

Indicator Function. In addition to shared embedding, we design an indicator function to indicate the probability of directly copying a word from the content as the tag. For a given piece of input content, we denote this probability when predicting the i-th tag as p_i, which can be computed as follows,

$$\begin{aligned} o_i &= v_2^T \cdot tanh(W_3 \cdot s'_i) \\ p_i &= \sigma(o_i) \end{aligned} \quad (6)$$

where s'_i is the overall representation of the current content as defined in Eq. (3), v_2 and W_3 are parameters, and $\sigma(x) = 1/(1 + exp(-x))$.

After p_i is learned, we directly copy one word in the current text with probability p_i. The chosen word is decided by the attention weights α_{ij} from Eq. (3). By denoting $P(z_i)$ as the original probability of generating a tag from the decoder section, and $\alpha(z_i)$ as the attention weight of word z_i,[3] the new probability $P'(z_i)$ of generating the i-th tag z_i can be computed as follows.

$$P'(z_i) = (1 - p_i) \cdot P(z_i) + p_i \cdot \alpha(z_i) \quad (7)$$

Experimental Evaluations

In this section, we present the experimental setup and analyze the experimental results.

[2]V is the union of the content vocabulary and tag vocabulary.

[3]We check word z_i before the calculation of $P(z_i)$. If word z_i is not in tag vocabulary, we set $\alpha(z_i) = 0$.

Table 2: Statistics of the datasets.

Dataset	Math	AU	SO
# of posts	89240	259740	350000
vocabulary size	15000	20000	10000
# of tags	1305	2500	1003
# of overlapping words	269	1794	709

Datasets and Setup

Datasets In our experiments, we use three real datasets: Mathematics Stack Exchange (Math), Ask Ubuntu (AU), and Stack Overflow (SO). All the datasets are officially published and publicly available.[4] Each dataset contains several posts, and each post contains a piece of textual content (a title and a body) and the corresponding tags. We make the following pre-processing steps on the datasets. First, we remove all the punctuation marks, and then split the remaining strings into individual words. Here, we preserve the stopwords in the text. In this way, we can retain the full text order information which is often ignored by the bag-of-words model. Next, we delete some low-frequency words in the textual content to reduce noise. Specially, we compute the frequency of each word and keep the top frequent words for each dataset in the vocabulary. For the words out of the vocabulary, we define a special symbol 'unknown'. There are two types of unknown words: 1) words that appear in the training set but not in the vocabulary, and 2) words that do not appear in the training set but in the test set. Finally, as the low-frequency tags are seldom used, we count the frequencies of tags and keep the most frequently used ones. The remaining content words and tags form the final vocabulary. We randomly select a subset of the SO datasets, and the statistics of the datasets after pre-processing are summarized in Table 2. For each dataset, we randomly select 90% of the data as the training set, and use the rest 10% as the test set. In the training set, we randomly select 10% data as the validation set.

Compared Methods We compare our method with the following existing methods.[5]

- *TagSpace* (Weston, Chopra, and Adams 2014). TagSpace is perhaps the first tag recommendation method that adopts the CNNs to model text.
- *Maxide* (Xu, Jin, and Zhou 2013). Maxide is a traditional multi-label learning method for tag recommendation.
- *Tag2Word* (Wu et al. 2016). Tag2Word is a supervised topic modeling method for tag recommendation. It considers the content-tag overlapping.
- *ABC* (Gong and Zhang 2016). ABC adopts an attention-based CNN architecture to recommend tags for textual content. It models the text sequential order with both local attention and the global attention.
- *TLSTM* (Li et al. 2016b). TLSTM combines topic modeling with LSTMs to model the text for tag recommendation. It also considers text sequential order.

Evaluation Metrics In terms of evaluation metrics, we use recall, precision, and F1-score to measure the performance.

Parameters and Initializations For the multi-layer structure, we use three GRU layers in the encoder section (i.e., $l = 3$). For each layer, we set the maximum text length to 100. We truncate the input text if its length exceeds the maximum length, and we add zero-padding if the input length is less than the maximum length. For all the compared methods, we set the word embedding dimensionality to 100 (i.e., $d_x = 100$) and randomly initialize the embeddings if applicable. For the GRU/LSTM neuron, we set the dimensionality of its hidden state to 256 (i.e., $d_h = 256$). When training the model, we adopt the RMSProp optimizer with the batch size set to 64. We adopt dropout with rate 0.1 to prevent over-fitting, and early stopping to stop the training process when the loss is no longer decreasing.[6]

Results and Analysis

Effectiveness Comparisons For effectiveness, we compare the proposed ITAG with the existing competitors, and the results are shown in Table 3. In the table, we report top-1, top-3, and top-5 results.

We can observe from the table that the proposed ITAG generally outperforms the compared methods on the three datasets. For example, on the F1 metric, the relative improvements of ITAG are 9.2 - 23.1%, 11.9 - 22.4%, and 8.2 - 13.5% compared with its best competitors on the three datasets, respectively. Moreover, we conduct a paired t-test showing that the improvements are statistically significant (e.g., p-value < 0.001). The reasons for the performance improvement of ITAG are as follows. Maxide and TagSpace perform relatively poor as they do not pay special attention on the three pillars as summarized in Table 1. ITAG is better than Tag2Word as Tag2Word only considers the content-tag overlapping. For ABC and TLSTM, although they model the sequential nature of text, the two pillars of tag correlation and content-tag overlapping are ignored.

Performance Gain Analysis Next, in order to verify the effectiveness of the three pillars, we perform a performance gain analysis by removing some of them. To be specific, we first remove the modeling of content-tag overlapping and tag correlation. The resulting method uses only the encoder part to model text, and then builds a classifier via a softmax layer. We refer to this method as ITAG_1. Next, we add the tag correlation component into ITAG_1 to obtain ITAG_2. The results are shown in Table 4, where we report the top-3 results on the AU data for brevity.

As we can see from the table, ITAG_1 which simply models the text with RNN and attention performs relatively poor. Next, when tag correlation is introduced, ITAG_2 significantly

[4] https://archive.org/details/stackexchange

[5] We do not compare with the latest work (Sun et al. 2018) as it requires a hierarchical knowledge graph of tags as input, which is unavailable in our problem setting.

[6] The code of the proposed method is publicly available at https://github.com/SoftWiser-group/iTag.

Table 3: Effectiveness comparisons on the three datasets. The proposed ITAG generally outperforms the existing methods (The symbol * means that the improvement is significant with p-value < 0.001).

Data	Methods	@1			@3			@5		
		Recall	Precision	F1	Recall	Precision	F1	Recall	Precision	F1
Math	TagSpace	0.065	0.144	0.090	0.141	0.115	0.126	0.186	0.091	0.122
	Maxide	0.054	0.126	0.075	0.121	0.097	0.108	0.167	0.081	0.109
	Tag2Word	0.190	0.396	0.257	0.331	0.246	0.282	0.425	0.194	0.267
	ABC	0.317	0.853	0.463	0.656	0.587	0.620	**0.779**	0.419	0.545
	TLSTM	0.281	0.756	0.410	0.548	0.491	0.518	0.658	0.353	0.460
	ITAG	**0.421***	**0.876***	**0.570***	**0.755***	**0.737***	**0.746***	0.766	**0.486***	**0.595***
AU	TagSpace	0.030	0.084	0.044	0.079	0.072	0.075	0.119	0.064	0.084
	Maxide	0.054	0.126	0.075	0.121	0.097	0.108	0.167	0.081	0.109
	Tag2Word	0.191	0.441	0.266	0.352	0.287	0.316	0.447	0.225	0.299
	ABC	0.265	0.724	0.388	0.550	0.500	0.524	0.678	0.370	0.479
	TLSTM	0.278	0.759	0.407	0.579	0.527	0.552	0.712	0.389	0.503
	ITAG	**0.363***	**0.794***	**0.498***	**0.657***	**0.658***	**0.657***	**0.715**	**0.464***	**0.563***
SO	TagSpace	0.051	0.126	0.073	0.113	0.087	0.098	0.161	0.076	0.103
	Maxide	0.049	0.121	0.069	0.110	0.088	0.098	0.159	0.075	0.102
	Tag2Word	0.219	0.430	0.290	0.366	0.251	0.298	0.463	0.196	0.275
	ABC	0.286	0.687	0.403	0.523	0.419	0.465	0.621	0.299	0.403
	TLSTM	0.294	**0.706**	0.415	0.540	0.433	0.480	**0.643**	0.309	0.417
	ITAG	**0.337***	0.672	**0.449***	**0.551***	**0.540***	**0.545***	0.630	**0.356***	**0.455***

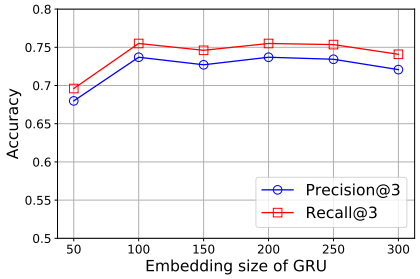
Table 4: Performance gain analysis of the proposed ITAG. All the three pillars contribute to the effectiveness of the proposed ITAG.

Models	Recall@3	Precision@3	F1@3
ITAG_1	0.073	0.068	0.071
ITAG_2	0.579	0.587	0.584
ITAG	0.657	0.658	0.657

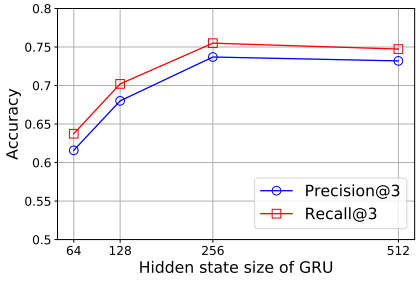
improves ITAG_1. Moreover, when the content-tag overlapping is modeled and incorporated into ITAG_2 (i.e., the ITAG method), further improvement is achieved. For example, the relative improvement on the F1@3 metric is 12.5%. We conclude that all the three pillars contribute to the effectiveness of the proposed model.

Parameter Sensitivity There are two major parameters in the proposed method, including word embedding size (d_x) and GRU hidden state size (d_h). The results are shown in Fig. 3, where we report the Recall@3 and Precision@3 results on the Math data for brevity. We can observe from Fig. 3(a) that the results are stable when d_x ls no less than 100. We have similar observations from Fig. 3(b), where the best d_h is round 256. Overall, the proposed method is robust with respect to these two parameters in a relatively large range. For the results we report in the paper, we fix these two parameters to 100 and 256, respectively.

Visualization Results Finally, we present some visualization results from the proposed method ITAG in Fig. 4, which could help to improve the interpretability of tag recommendation. The examples are all selected from the test set of the SO data, with deeper (yellow) color indicating higher attention weights. We can first see from the first two examples



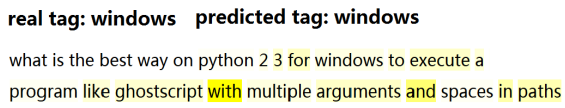
(a) The word embedding size d_x.



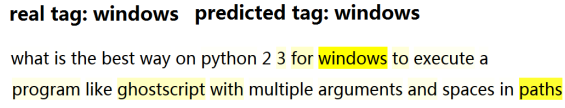
(b) The GRU hidden state size d_h.

Figure 3: Parameter sensitivity results. The proposed ITAG is robust with respect to the two parameters in a relatively wide range.

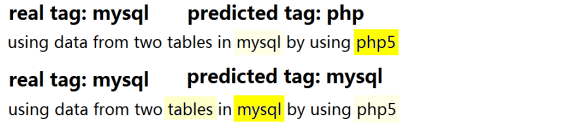
that, although both methods successfully predict the right tag 'windows', the attention learned when considering content-tag overlapping is more meaningful. That is, the attention of the model without content-tag overlapping is scattered



(a) Without modeling content-tag overlapping



(b) With modeling content-tag overlapping



(c) Predicting additional reasonable tags

Figure 4: Visualization examples of ITAG. The first two examples show that although both models can predict the right tag, modeling the content-tag overlapping helps to keep the model focused on the tag-related part of the input text. The third example shows that the proposed model can sometimes predict reasonable tags that are not in the ground-truth list.

around the text, whereas most of the attention with content-tag overlapping is concentrated on the word 'windows' in the text. Moreover, the common evaluation protocol with existing tag recommendation methods is to evaluate the effectiveness by accurately predicting the real tags in the datasets. However, the tags in the datasets are manually annotated by users, and as such they might be incomplete or even improper in some cases. The third example demonstrates that our model is able to identify some reasonable tags although these tags are not listed as ground-truth in the datasets. For example, although the ground-truth tag is 'mysql' in the test set, the predicted tag 'php' by our method is also a reasonable suggestion.

Related Work

In this section, we briefly review the related work, which is organized into content-based methods and collaborative filtering methods.

Content-based methods. Content-based methods aim to recommend suitable tags by directly modeling the content. The focus of this paper is on the textual content, and we divide existing methods into two steps: text representation and candidate tag generation. Text representation methods include TF-IDF, topic models (Ramage et al. 2009; Krestel, Fankhauser, and Nejdl 2009; Seitlinger et al. 2013; Wu et al. 2016; 2018), and neural networks (Weston, Chopra, and Adams 2014; Li et al. 2016a; Gong and Zhang 2016; Li et al. 2016b; Sun et al. 2018). A recent trend is to apply neural networks such as RNNs to learn the text representations. For example, Gong et al. (2016) add the attention mechanism to CNN to model the text; Li et al. (2016b) further combine RNN with topical distributions to learn text representations. Different from these methods, we further consider the content-tag overlapping and the tag correlation. As to candidate tag generation, existing methods mainly adopt classification (Li et al. 2016a; 2016b; Gong and Zhang 2016) and clustering-based methods (Mishne 2006; Sigurbjörnsson and Van Zwol 2008). Different from these methods, we adopt the sequence generation method for candidate tag generation so as to capture the tag correlation phenomenon.

Although the focus of this work is on the textual content, there are also research on recommending tags for other types of content. Among them, recommending tags for images are widely studied (Sigurbjörnsson and Van Zwol 2008; Liu et al. 2014; Rawat and Kankanhalli 2016; Zhang et al. 2017)). For example, Liu et al. (2014) combine visual features and textual information to recommend tags for photos; Rawat et al. (2016) propose a deep neural network to predict multiple tags for images. In addition to images, Zhao et al. (2010) recommend tags for songs based on tag semantic similarity and music content; Toderici et al. (2010) recommend tags for YouTube videos solely based on their audiovisual content; Font et al. (2014) propose a tag recommendation system in an online platform for audio clip sharing.

Collaborative filtering methods. In general, tag recommendation methods based on collaborative filtering take the existing tagging history as input, and aim to recommend tags for users in a personalized manner. Typically, existing collaborative filtering methods often model the tag recommendation problem as a k-nearest neighbor problem (Zheng and Li 2011), a tensor factorization problem (Symeonidis, Nanopoulos, and Manolopoulos 2008; Fang et al. 2015), or a graph link prediction problem (Feng and Wang 2012; Zhao, Guan, and Liu 2015). The combination of collaborative filtering method and content-based method is also studied (Wang, Chen, and Li 2013; Wang, Shi, and Yeung 2015).

Conclusions

In this paper, we have proposed an effective, unified tag recommendation model ITAG for textual content. The key idea of ITAG is to model three pillars (i.e., sequential text modeling, tag correlation, and content-tag overlapping) into a coherent encoder-decoder framework. Experimental results on three real-world datasets demonstrate that the proposed method significantly outperforms the existing methods in terms of recommendation accuracy, and our method is capable of producing interpretable results. Future directions include recommending tags for specific domains such as programming question answering sites, combining textual content with other types of content for tag recommendation, and extending the proposed method into the personalized setting.

Acknowledgments. This work is supported by the National Key Research and Development Program of China (No. 2016YFB1000802), the National Natural Science Foundation of China (No. 61690204, 61672274, 61702252), the Huawei Innovation Research Program (No. HO2018085291), and the Collaborative Innovation Center

of Novel Software Technology and Industrialization. Hanghang Tong is partially supported by NSF (IIS-1651203, IIS-1715385 and IIS-1743040), and DHS (2017-ST-061-QA0001). Xiaohui Yan is partially supported by the National Natural Science Foundation of China (No. 61502447).

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv*.

Belém, F. M.; Almeida, J. M.; and Gonçalves, M. A. 2017. A survey on tag recommendation methods. *Journal of the Association for Information Science and Technology* 68(4):830–844.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv*.

Fang, X.; Pan, R.; Cao, G.; He, X.; and Dai, W. 2015. Personalized tag recommendation through nonlinear tensor factorization using gaussian kernel. In *AAAI*.

Feng, W., and Wang, J. 2012. Incorporating heterogeneous information for personalized tag recommendation in social tagging systems. In *SIGKDD*, 1276–1284.

Font, F.; Serra, J.; and Serra, X. 2014. Class-based tag recommendation and user-based evaluation in online audio clip sharing. *Knowledge-Based Systems* 67:131–142.

Gong, Y., and Zhang, Q. 2016. Hashtag recommendation using attention-based convolutional neural network. In *IJCAI*.

Guo, G.; Wang, H.; Bell, D.; Bi, Y.; and Greer, K. 2004. An knn model-based approach and its application in text categorization. In *CICLing*, 559–570.

Guy, I.; Zwerdling, N.; Ronen, I.; Carmel, D.; and Uziel, E. 2010. Social media recommendation based on people and tags. In *SIGIR*, 194–201.

Krestel, R.; Fankhauser, P.; and Nejdl, W. 2009. Latent dirichlet allocation for tag recommendation. In *RecSys*, 61–68.

Li, J.; Xu, H.; He, X.; Deng, J.; and Sun, X. 2016a. Tweet modeling with lstm recurrent neural networks for hashtag recommendation. In *IJCNN*, 1570–1577.

Li, Y.; Liu, T.; Jiang, J.; and Zhang, L. 2016b. Hashtag recommendation with topical attention-based lstm. In *COLING*.

Li, X.; Guo, L.; and Zhao, Y. E. 2008. Tag-based social interest discovery. In *WWW*, 675–684.

Liu, J.; Li, Z.; Tang, J.; Jiang, Y.; and Lu, H. 2014. Personalized geo-specific tag recommendation for photos on social websites. *IEEE Transactions on Multimedia* 16(3):588–600.

Mishne, G. 2006. Autotag: a collaborative approach to automated tag assignment for weblog posts. In *WWW*, 953–954.

Ramage, D.; Hall, D.; Nallapati, R.; and Manning, C. D. 2009. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP*, 248–256.

Rawat, Y. S., and Kankanhalli, M. S. 2016. Contagnet: Exploiting user context for image tag recommendation. In *MM*, 1102–1106. ACM.

Seitlinger, P.; Kowald, D.; Trattner, C.; and Ley, T. 2013. Recommending tags with a model of human categorization. In *CIKM*, 2381–2386.

Sigurbjörnsson, B., and Van Zwol, R. 2008. Flickr tag recommendation based on collective knowledge. In *WWW*, 327–336.

Sun, B.; Zhu, Y.; Xiao, Y.; Xiao, R.; and Wei, Y. G. 2018. Automatic question tagging with deep neural networks. *IEEE Transactions on Learning Technologies*.

Symeonidis, P.; Nanopoulos, A.; and Manolopoulos, Y. 2008. Tag recommendations based on tensor dimensionality reduction. In *RecSys*, 43–50.

Toderici, G.; Aradhye, H.; Pasca, M.; Sbaiz, L.; and Yagnik, J. 2010. Finding meaning on youtube: Tag recommendation and category discovery. In *CVPR*, 3447–3454.

Wang, Y., and Wang, Z.-O. 2007. A fast knn algorithm for text categorization. In *International Conference on Machine Learning and Cybernetics*, volume 6, 3436–3441.

Wang, J.; Yang, Y.; Mao, J.; Huang, Z.; Huang, C.; and Xu, W. 2016. Cnn-rnn: A unified framework for multi-label image classification. In *CVPR*, 2285–2294.

Wang, H.; Chen, B.; and Li, W.-J. 2013. Collaborative topic regression with social regularization for tag recommendation. In *IJCAI*, 2719–2725.

Wang, H.; Shi, X.; and Yeung, D.-Y. 2015. Relational stacked denoising autoencoder for tag recommendation. In *AAAI*, 3052–3058.

Weston, J.; Chopra, S.; and Adams, K. 2014. #tagspace: Semantic embeddings from hashtags. In *EMNLP*, 1822–1827.

Wu, Y.; Yao, Y.; Xu, F.; Tong, H.; and Lu, J. 2016. Tag2word: Using tags to generate words for content based tag recommendation. In *CIKM*, 2287–2292.

Wu, Y.; Xi, S.; Yao, Y.; Xu, F.; Tong, H.; and Lu, J. 2018. Guiding supervised topic modeling for content based tag recommendation. *Neurocomputing* 314:479–489.

Xu, M.; Jin, R.; and Zhou, Z.-H. 2013. Speedup matrix completion with side information: Application to multi-label learning. In *NIPS*, 2301–2309.

Zhang, Q.; Wang, J.; Huang, H.; Huang, X.; and Gong, Y. 2017. Hashtag recommendation for multimodal microblog using co-attention network. In *IJCAI*, 3420–3426.

Zhao, Z.; Wang, X.; Xiang, Q.; Sarroff, A. M.; Li, Z.; and Wang, Y. 2010. Large-scale music tag recommendation with explicit multiple attributes. In *MM*, 401–410. ACM.

Zhao, W.; Guan, Z.; and Liu, Z. 2015. Ranking on heterogeneous manifolds for tag recommendation in social tagging services. *Neurocomputing* 148:521–534.

Zheng, N., and Li, Q. 2011. A recommender system based on tag and time information for social tagging systems. *Expert Systems with Applications* 38(4):4575–4587.