Large-Scale Question Tagging via Joint Question-Topic Embedding Learning

LIQIANG NIE, Shandong University, China
YONGQI LI, Shandong University, China
FULI FENG, National University of Singapore, Singapore
XUEMENG SONG, Shandong University, China
MENG WANG, Hefei University of Technology, China
YINGLONG WANG, Qilu University of Technology (Shandong Academy of Sciences), China

Recent years have witnessed a flourishing of community-driven question answering, like Yahoo! Answers and AnswerBag, where people can seek precise information. After 2010, some novel cQA systems, including Quora and Zhihu, are gaining momentum. Besides interactions, the latter enables users to label the questions with topic tags that highlight the key points conveyed in the questions. In this paper, we shed light on automatically annotating a newly posted question with topic tags which are pre-defined and pre-organized into a directed acyclic graph. To accomplish this task, we present an end-to-end deep interactive embedding model to jointly learn the embeddings of questions and topics by projecting them into the same space for similarity measure. In particular, we first learn the embeddings of questions and topic tags by two deep parallel models. Thereinto, we regularize the embeddings of topic tags via fully exploring their hierarchical structures, which is able to alleviate the problem of imbalanced topic distribution. Thereafter, we interact each question embedding with the topic tag matrix, i.e., all the topic tag embeddings. Following that, a sigmoid cross-entropy loss is appended to reward the positive question-topic pairs and penalize the negative ones. To justify our model, we have conducted extensive experiments on an unprecedented large-scale social QA dataset obtained from Zhihu.com, and the experimental results demonstrate that our model achieves superior performance to several state-of-the-art baselines.

CCS Concepts: • **Information systems** → **Document topic models**; *Learning to rank*.

Additional Key Words and Phrases: Question Tagging, Topic Hierarchy, CQA, Embedding Learning

ACM Reference Format:
LIQIANG NIE, YONGQI LI, FULI FENG, XUEMENG SONG, MENG WANG, and YINGLONG WANG. 2019. Large-Scale Question Tagging via Joint Question-Topic Embedding Learning. *ACM Transactions on Information Systems* 1, 1, Article 1 (January 2019), 24 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Authors' addresses: LIQIANG NIE, Shandong University, China, nieliqiang@gmail.com; YONGQI LI, Shandong University, China, liyongqi0@gmail.com; FULI FENG, National University of Singapore, Singapore, fulifeng93@gmail.com; XUEMENG SONG, Shandong University, China, sxmustc@gmail.com; MENG WANG, Hefei University of Technology, China, eric.mengwang@gmail.com; YINGLONG WANG, Qilu University of Technology (Shandong Academy of Sciences), China, wangyl@sdas.org.

1046-8188/2019/1-ART1 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The last three decades have witnessed the proliferation of community-based question answering, cQA for short. cQA sites, like Yahoo! Answers[1], AnswerBag[2], and Stack Overflow[3], are places to gain and share knowledge whereby users are encouraged to ask or answer questions and are able to connect with the contributors of unique insights and quality answers [4, 40, 42]. Users are thus empowered to learn from each other and to better understand the world. After the year of 2010, some novel sites in the cQA family have emerged, such as Zhihu[4] and Quora[5]. We hereafter refer to the novel cQA sites as collaborative cQA [7, 37], which refers to cQA systems that allow collaborative question answering. Compared to the conventional cQA sites, the collaborative ones have the following two prominent characteristics: 1) In the collaborative cQA sites, the spirit of crowdsourcing is highly encouraged, whereby any authenticated user is allowed to perform editing on any question, answer, and even topic, to refine the quality of community content. And 2) the collaborate cQA sites highlight more on social connections among users. For example, users in the collaborative cQA sites, like Quora and Zhihu, are enabled to follow each other, and even topics they are interested in, which is not allowed in conventional cQA sites, like Yahoo!Answers and AnswerBag[6]. These prominent features make them thriving. Considering Quora as an example, it had obtained over 300 million monthly unique visitors and archived 38 million plus distinct questions as of the May of 2019[7].

Owing to the proliferation of cQA sites, the amount of questions generated and answered by people is growing exponentially. So it becomes increasingly difficult and expensive for users to locate the questions they need and are interested in. Therefore, cQA sites organize questions by user-generated topics because tagging is a simple and efficient method to organize resources. To be more specific, each asker is strongly encouraged to select multiple topic tags from the suggested candidate list for labeling the newly posted question, which summarizes the question content in a coarse-grained but semantically meaningful level. One typical example of question tagging is demonstrated in Figure 1. Topic tags play pivotal roles in cQA sites, including but not limited to the following aspects: 1) Question routing. In addition to the unidirectional user-follow-user relations, in cQA sites, users can also follow the topics of interest. In the light of this, cQA sites can put the questions into the feeds of associated topic-followers to draw more attention from the potential answerers, and thus receive more quick and accurate answers. And 2) topic tags can be leveraged to benefit index, search, navigation and organization. Therefore, question tagging in cQA sites deserves our attention.

Despite its significance and value, question tagging in cQA is non-trivial due to the following facts: 1) Topic tags are not independent. Questions are organized by topic tags (e.g. Quora, Stack Overflow, and Zhihu) or topic categories (e.g. AnswerBag, Yahoo! Answer). For the cQA systems based on topic tags, the structures of topics tags are hierarchical, like Zhihu, or flat, like Quora and Stack Overflow. In this work, we focus on cQA systems with structural topic tags. For example, in Zhihu systems, topic tags are organized into a directed acyclic

[1] https://answers.yahoo.com/.
[2] https://www.answerbag.com/.
[3] https://stackoverflow.com/.
[4] https://www.zhihu.com/.
[5] https://www.quora.com/.
[6] https://socialcompare.com/en/comparison/compare-question-answer-sites-quora-vs-yahoo-answers-vs-stackoverflow-vs-ted-conversations.
[7] https://www.quora.com/How-many-users-does-Quora-have-in-2019.

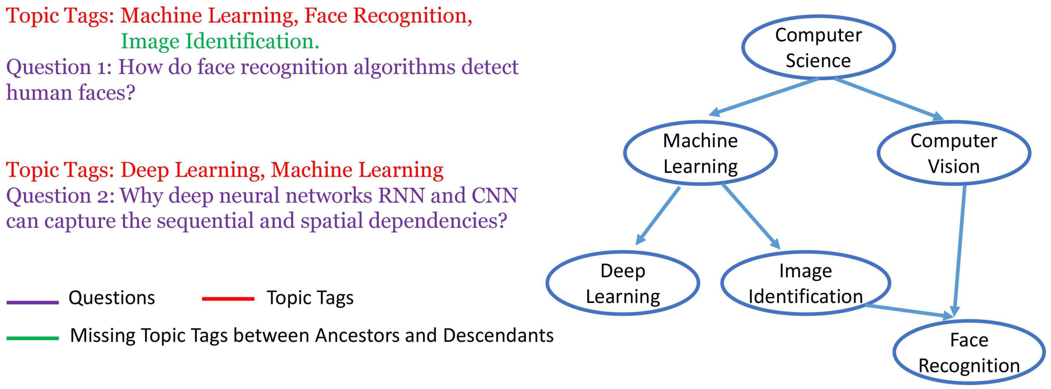

Topic Tags: Machine Learning, Face Recognition,
Image Identification.
Question 1: How do face recognition algorithms detect human faces?
Topic Tags: Deep Learning, Machine Learning
Question 2: Why deep neural networks RNN and CNN can capture the sequential and spatial dependencies?
Questions
Topic Tags
Missing Topic Tags between Ancestors and Descendants
Computer Science
Machine Learning
Computer Vision
Deep Learning
Image Identification
Face Recognition

Fig. 1. Exemplars of question tagging in cQA sites and the directed acyclic graph structure of topic tags.

graph (DAG) by experienced users and the hired experts, as shown in Figure 1. The DAG is more like a tree structure except that some nodes have multiple parents. From the root to leaf nodes, topic tags tend to be more specific. A question can be annotated by either the leaf or the internal nodes at the same time. How to encode the hierarchical structure of topic tags to fully explore the correlations among topic tags is the first challenge we are facing. It is worth mentioning that some cQA sites based on flat topic tags, such as Quora and Stack Overflow, do not organize topic tags into a hierarchical structure explicitly. However, their topic tag structures do exist in pairwise forms and their graph-based ontologies can be built. For example, tags "programming language" and "python" in Stack Overflow can be treated as the father-son pair, as python is one kind of programming languages. We will discuss how to model the tag hierarchy on those cQA sites in Section 6. 2) In the real-world settings, the cQA sites may archive dozens of thousands, even hundreds of thousands of topic tags. The number of questions tagged by each topic tag varies significantly. Statistics tell us that the leaf nodes account for around 70% of the DAG, while they label only about 25% of the questions. Whereas 30% of the DAG are the internal nodes used to label the rest questions. Considering such imbalanced distribution, it may be suboptimal to directly cast the tagging task as a classification problem. Instead of classification, in this paper, we treat the labeling task as a ranking problem by measuring the similarity between the given question and each topic tag candidates. In view of this, how to jointly learn the embeddings of questions and imbalanced topic tags within the same space is largely untapped. Meanwhile, the knowledge is supposed to be transfered from the more frequent internal nodes to the less frequent leaf ones for alleviating the imbalance based on the given DAG structure. And 3) the publicly accessible collaborative cQA dataset with explicit topic tag hierarchy are still in small scales. For example, the largest publicly available collaborative cQA dataset with topic tag hierarchy, thus far, has been released by the contest with less than 2,000 topic tags after desensitization[8], which is not representative enough for evaluation. How to justify the effectiveness of our model on a real-world and large-scale dataset is also a challenge we have to solve.

To address the aforementioned challenges, in this paper, we present a scheme for question tagging on cQA systems with structural topic tags, as illustrated in Figure 2. This scheme consists of offline learning and online tagging. As to the offline part, we propose an end-to-end

[8] https://zhuanlan.zhihu.com/p/26843044.

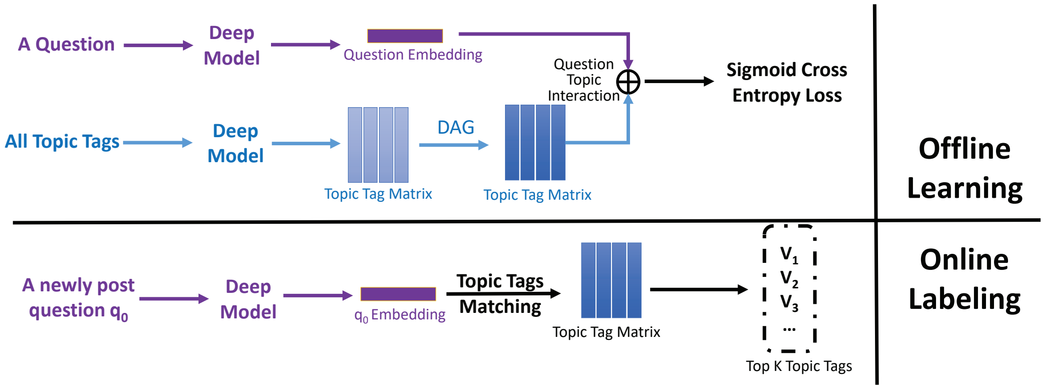


Fig. 2. Schematic illustration of our proposed scheme for question tagging in the cQA setting. In the offline part, we train a PROFIT model to learn the question and topic embedding. As to the online one, we recommend relevant topic candidates based on the matching score.

deeP inteRactive mOdel For questIon Tagging, dubbed as PROFIT, which is capable of simultaneously projecting the questions and imbalanced topic tags into the same space and learning their representations for similarity measurement. In particular, we first leverage a convolutional neural network (CNN) [18] and a multi-layer perception (MLP) model [51] to learn the embeddings of each question and all the imbalanced topic tags, respectively. Notably, the questions and topic tags are forced to share the same set of word embeddings that are pre-trained over a very large cQA set. To learn the discriminative embeddings, we encode the DAG structure of topic tags into our learning model by regularizing the hierarchical relations among topic tags. Considering the less frequent problem of the leaf nodes in the DAG, we leverage the weighted linear combination of the parent nodes to represent the child one, which is able to transfer the knowledge from top to down. We thereafter perform the interaction between the question embedding and the topic tag matrix, to obtain a similarity vector, whereby each element in the vector correspondingly represents the matching score between the question and the topic tag. We then normalize the similarity vector by a sigmoid layer which computes the sigmoid result for each element in the similarity matrix.

In the training phase, a cross-entropy loss is followed to reward the positive question-tag pairs and penalize the negative ones, which indeed implicitly guarantees the question and topic tag embeddings towards the same semantic space. In our work, we treat the (question$_i$, tag$_j$) as a positive pair if the tag$_j$ labels the question$_i$ before; otherwise, we view it as the negative. To train our proposed PROFIT model, we cooperate with the Zhihu company, China's biggest question-and-answer-style knowledge base, enabling us to run our model on an unprecedented large-scale cQA data collection. After the training, we can learn the embeddings of all the archived topic tags offline and a newly posted question online. Based upon this, we can recommend users the top topic tags according to the similarity between the embeddings of each topic tag and the newly posted question. This is in fact the online part. By conducting extensive experiments, our model is demonstrated to yield superior performance to several state-of-the-art baselines.

To sum up, the contributions in this work are threefold:

- As far as we know, this is the first work on question tagging in the cQA setting that leverages the DAG structure of topic tags by regularizing their hierarchical relations to transfer the knowledge from top to down discriminative embedding learning. This model somehow alleviates the problem of imbalanced topic tag distribution.
- We present a deep parallel scheme, which jointly learns the topic tag and question embeddings, and recommends topic candidates to the given question based on their embedding interaction. Unlike pure classification models, we actually cast the question tagging task into a ranking problem, which considers the topic semantics during the interaction.
- We comparatively justify our proposed PROFIT model over a large-scale and real-world dataset. In addition, we have released the codes and the involved parameters in this paper[9]. Particularly, we have released the new big cQA data with hierarchical topic tags, which will help boost the research.

The rest of this paper is structured as follows. In Section 2, we briefly review the related literature. In Section 3, we detail our proposed model, followed by experimental results and analyses in Section 4. We finally conclude the work and figure out the future directions in Section 5.

2 RELATED WORK

QA systems alleviate information overload by providing users simple and accurate answers. Besides, a great many researches have been conducted in QA systems, such as answer ranking [12, 16, 41] and question routing [28, 53]. Topic tags play a key role in question organization, routing and search, and many question tagging methods have been proposed [1, 20, 24]. On the basis of state-of-the-arts reviews, we mainly review related work about the hierarchical question classification, question tagging in cQA.

2.1 Hierarchical Question Classification

In the era of community-based question answering, representative cQA communities like Yahoo! Answers and Oshiete! Goo, organize their content by hierarchical ontologies. In particular, when posting a question, the user is encouraged to select a single category tag at the leaf of the given ontology to label the question. In such a context, several automatic question classification approaches have sprung up miscellaneously: big-bang and top-down approaches. The former often trains a single classifier and employs it to assign one leaf node of the category tree to the given question [3, 36]. By contrast, the latter constructs one classifier per level of the hierarchy in the training phase and classifies each given question from the higher-level to lower-ones until reaches a leaf category [29, 31].

Nevertheless, questions, more often than not, convey multiple topics. Thus, it gets more difficult to find a single appropriate category label to describe a given question. What is more, the leaf categories in the pre-defined ontology are insufficient to handle the ever-increasing questions with various topics. Taking Yahoo! Answer as an example, it has only 1,263 leaf-level nodes distributed over 26 top-level categories. This category vocabulary is extremely limited. Nishida and Fujimura in 2010 [26] partially alleviated these phenomena by annotating a given question with different abstractions, namely category, theme, and keywords. However, automatically generated theme and keywords are still far from satisfactory, as compared to the user-generated topic tags in the cQA sites.

[9] https://question-tagging.wixsite.com/anonymous.

2.2 Question Tagging in cQA

Stack Overflow is a popular Q&A site focusing on technical questions about software development, helping software engineers to strengthen their abilities in software development, maintenance and test processes. Topic tags are popular in Stack Overflow, used to search, describe, identify, and bookmark various software objects, as well as bridge the gap between social needs and technical development. In the year of 2013, Saha et al. [33] introduced a discriminative model to suggest tags for questions on Stack Overflow. This model consists of three main steps: converting questions into vectors with term frequency weighting scheme, training a discriminative model with SVM classifier, and suggesting tags with the top similarity. In the same year, Xia et al. [47] presented a composite framework, named TagCombine, to solve the tag recommendation problem from three different views with three components. One year later, Wang et al. in [45] introduced EnTagRec, an automatic tag recommender based on historical tag assignments, which improved TagCombine by 27.3% with respect to Recall@5. To further improve the quality of tags in Stack Overflow, Wang et al. [46] in 2018 proposed EnTagRec++, an advanced version of their prior work, i.e., EnTagRec. Beyond EnTagRec, EnTagRec++ does not only integrate the historical tag assignments to software objects, but also leverages the user information and an initial set of tags that a user may provide for tag recommendation. In the year of 2018, zhang et al. [49] introduced a multitasking-like convolutional neural network to learn semantic vectors for tag recommendation.

Besides Stack Overflow, there are some work on other cQA sites. Early in the last decade, pioneers conducted an in-depth analysis of the question labelling practices by contrasting the use of community-generated tags in the Live QnA service with the use of topic categories from a fixed taxonomy in the Yahoo! Answers service [22, 32]. They found that community tagging was related to higher levels of social interactions amongst users. The analysis of the most frequently used community tags reveals that active users may establish strong social ties around specific tags [11]. With the rise of collaborative cQA sites, a data-driven study over Quora was presented in 2013 [43], reporting that the user-follow-topic graph appeals to users in browsing and answering general questions. Nie et al. in 2014 [25] noticed the incompleteness of question tags in cQA sites and devised a novel scheme to automatically annotate questions, which was accomplished by finding appropriate tags from similar questions via an adaptive probabilistic hypergraph. Although existing studies have achieved compelling success in the question tagging on cQA sites, they failed to take into account the valuable hierarchical structures among topic tags. We have to mention that Zhihu.com organized a contest on question annotation in 2017[10] and attracted a notable amount of participants. To support this competition, Zhihu released 3 million questions and 1,999 structured topic tags after desensitization. After going through all the submitted solutions, we found that none of them took the DAG hierarchy into consideration during the learning process.

3 DATA COLLECTION

Zhihu is a Chinese socialized question-and-answer website where questions are collaboratively created, answered, edited and organized by its user community. Its website, zhihu.com, was launched on January 26, 2011. As of January of 2019, Zhihu has obtained 220 million registered users, 29 billion monthly page views, 26 million active users surfing the website on average 1 hour daily, as well as accumulated 30 million questions, 130 million answers, and

[10] https://biendata.com/competition/zhihu/.

35 million votes in total[11]. In Zhihu, users present their professionalism, find high-quality information to facilitate their decision making, and contact with people from whom they seek help, build collaboration or partnership. Because users in Zhihu can generate a tremendous number of content everyday, how to understand the created content and distribute them in a highly effective way is greatly desired in all cQA sites.

Routing content to appropriate users according to user-follow-topic relationship is natural, since the followed topic tags are able to better fulfill users' needs on the desired knowledge. In the light of this, automatic and accurate labelling topic tags for questions in cQA sites will play a key role in enhancing user experience and content distribution efficiency.

Cooperating with Zhihu company, we have obtained two datasets. One is a benchmark dataset manually pruned for a public contest (Dataset I), and the other is a real-world dataset without any preprocessing (Dataset II).

3.1 Dataset I

Zhihu algorithm team, together with IEEE Computer Association and IEEE China office, host Zhihu Machine Learning Challenge 2017, aiming to recommend topic tags to questions. Participants should design an automatic tagging model for untagged questions. We thereafter name this dataset as Dataset I.

Dataset I consists of $2,999,967$ questions labeled by $1,999$ distinct topic tags. Thereinto, 90.4% of the topic tags come with descriptions. On average, each topic tag has 3.73 words and labels $3,513$ questions. As mentioned before, most of the topic tags are coordinately organized into a hierarchical DAG by users and experts hired by Zhihu. In the given DAG, the $1,999$ topic tags are linked by $2,653$ edges and each edge represents the parent-child relation. It is worth emphasizing that, according to our statistics, 41.6% of question tags are internal nodes within the DAG structure, and the rest are leaf nodes. We list the statistical information of the given DAG structure in Dataset I in Table 1. In addition, we display the tag length distribution, frequency distribution, as well as the number of topic tags per question in Figure 3(a)-3(c).

On average, each question has been labeled by 2.34 topic tags. Stepping into the topic tags of the same question, we find that, 16.7% of the topic tags are siblings and 30.4% of them are ancestors and descendants. Among the ancestor-descendant topic tags of the same question, the average depth distance is 1.52. Before training our model, we first automatically label each question with extra topic tags on the path between two ancestor-descendant topic tags, as shown in Figure 1,whereinto the topic tag in green is the missing one and we complete it. Ultimately, each question has 2.57 topic tags after completion.

Considering user privacy and data security, the contest does not provide the original texts of the questions and topics, but uses numbered codes and numbered segmented words to represent text messages. Meanwhile, considering the vast use of Distributed Representation[19, 27], the contest provides embedding vectors at the level of character and word. These embedding vectors are obtained by conducting training with google word2vec and taking the advantage of the mega text corpora provided by Zhihu.

3.2 Dataset II

Considering that Dataset I is preprocessed for the contest usage, whereby the data distribution does not coincide with the real-world cQA and the original textual data are not available due to the privacy concerns, we hence co-work with Zhihu to build a large-scale and representative

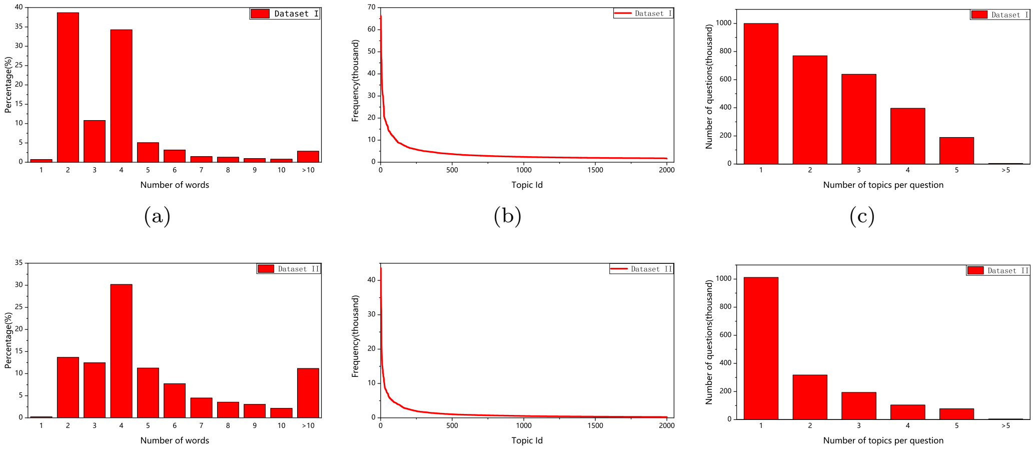
[11] https://www.infoq.cn/article/BNcC3WccELmPw6_LFFxP.

Table 1. Statistics of the given hierarchical DAG structure of topic tags in Dataset I.

Max Depth	Min Depth	Average Depth	# Internal Node	# Leaf Node	# Edge	# Average Children
15	1	5.16	831 (41.6%)	1,168 (58.4%)	2,653	3.2

Table 2. Statistics of the given hierarchical DAG structure of topic tags in Dataset II.

Max Depth	Min Depth	Average Depth	# Internal Node	# Leaf Node	# Edge	# Average Children
19	1	8.83	4,033 (30.5%)	9,225 (69.5%)	17,874	4.4



(a) (b) (c)

(d) (e) (f)

Fig. 3. Statistics of our Dataset I and II, obtained from Zhihu.com. (a,d) The length distribution of topic tags measured by the number of words. (b,e) The frequency distribution of topic tags measured by the number of questions labeled by the same topic tag. And (c,f) the number of questions labeled by different numbers of topic tags.

dataset. To be more specific, our dataset contains 1,707,023 textual questions and 10,843,647 corresponding answers. The questions are labeled with 13,258 distinct topic tags. Thereinto, 30.1% of the topic tags come with descriptions. The length and frequency distribution of topic tags are illustrated in Figure 3(d) and 3(e), respectively. On average, each topic tag has 5.73 words and labels 229 questions. As mentioned before, most of the topic tags are coordinately organized into a hierarchical DAG by users and experts hired by Zhihu. In the DAG, there are 13,258 topic tags linked by 17,874 edges and each edge represents the parent-child relation. The meta information of the DAG is shown in Table 2. It is worth emphasizing that, according to our statistics, 30.5% of question tags are internal nodes within the DAG, while 69.5% are leaf nodes.

In our dataset, the number distribution of topic tags for each question is illustrated in Figure 3(f). On average, each question has been labeled by 1.78 topic tags. Going deep into the topic tags of the same question, we find that, 17.2% of the topic tags are siblings and 34.7% are ancestors and descendants. Among the ancestor-descendant topic tags of the same question, the average depth distance is 1.68. Before training our model, we first automatically label each question with extra topic tags on the path between two ancestor-descendant topic tags, as shown in Figure 1. Ultimately, each question has 2.07 topic tags after completion.

It's worth mentioning that ordinary users are not allowed to create new topic tags[12]. In other words, the DAG is relatively stable. In the light of this, we only consider the topic tags in the DAG and target at recommending the relevant topic tags to the newly posted question.

4 OUR PROPOSED METHOD

In this section, we first define some notations and then detail our proposed PROFIT model.

4.1 Notation and Problem Formulation

For the ease of problem formulation, we first declare some notations. In particular, we use bold capital letters (e.g., $\mathbf{X}$) and bold lowercase letters (e.g., $\mathbf{x}$) to denote matrices and vectors, respectively. We employ non-bold letters (e.g., X) to represent scalars, and Greek letters (e.g., λ) as parameters. If not clarified, all vectors are in the column form.

Assume that we are given a set of N questions $\mathcal{Q} = \{q_1, q_2, ..., q_N\}$ labeled by M topic tags $\mathcal{T} = \{t_1, t_2, ..., t_M\}$. The topic tags are pre-organized into a tree-like DAG ontology $\mathcal{G}$, whereby the leaf and internal nodes in the DAG are topic tags and the edges to represent the parent-child relationship. Notably, a node in $\mathcal{G}$ may have more than one parent. The deeper level in the DAG a topic tag locates, the finer-grained concept it conveys; and vice versa. Therefore, the leaf nodes represent the most specific topic tags. Our research objective is to train a deep embedding model with $\mathcal{Q}$ and $\mathcal{G}$, towards identifying the relevant topic tags from $\mathcal{T}$ for a newly posted question q.

4.2 Deep Interactive Embedding Model

The offline part of our developed scheme for question tagging is the PROFIT model. Given the questions and their associated topic tags, the PROFIT model targets at learning their embeddings for further similarity matching. It is worth noting that the topic tags in the given DAG are relatively stable and hence the embeddings of all the topic tags can be further used during the online tagging. As demonstrated in Figure 4, the PROFIT model comprises three components, namely, question embedding layer, topic tag embedding layer, and question-topic interaction layer. We will detail them separately.

4.2.1 Question Embedding Layer. On the one hand, unlike the normal documents, in cQA, questions are typically short and most of the topic tags are usually phrases containing a few words. Statistically, on average, each question and each topic tag has 12.91 and 1.35 terms in Dataset I (13.12 and 2.08 terms in Dataset II), respectively. They thus do not provide sufficient contexts for similarity matching between questions and topic tags, especially for the keywords-based matching. On the ther hand, the traditional widely-used methods on text embedding mainly rely on N-gram models, like uni-gram, bi-gram and tri-gram. Despite their popularity, N-gram have two kinds of defects: 1) Uni-gram embedding destroys the word sequence which in turn destroys much of the semantic structure. Even though N-grams models partially consider the word order in short or local context, they have very little sense about the semantics of the words or more formal distances among the words. And 2) the dimensionality of N-gram representation is proportional to the dictionary size, leading to sparse and high dimensional embeddings.

Considering that the CNN model and its variants have been very successful in computer vision tasks [9, 10, 35], recommender systems [5, 48], with excellent performance in natural

[12] Only experienced users are allowed to create new topic tags. In particular, experienced users are the people who have more than five answers with no less than five votes.

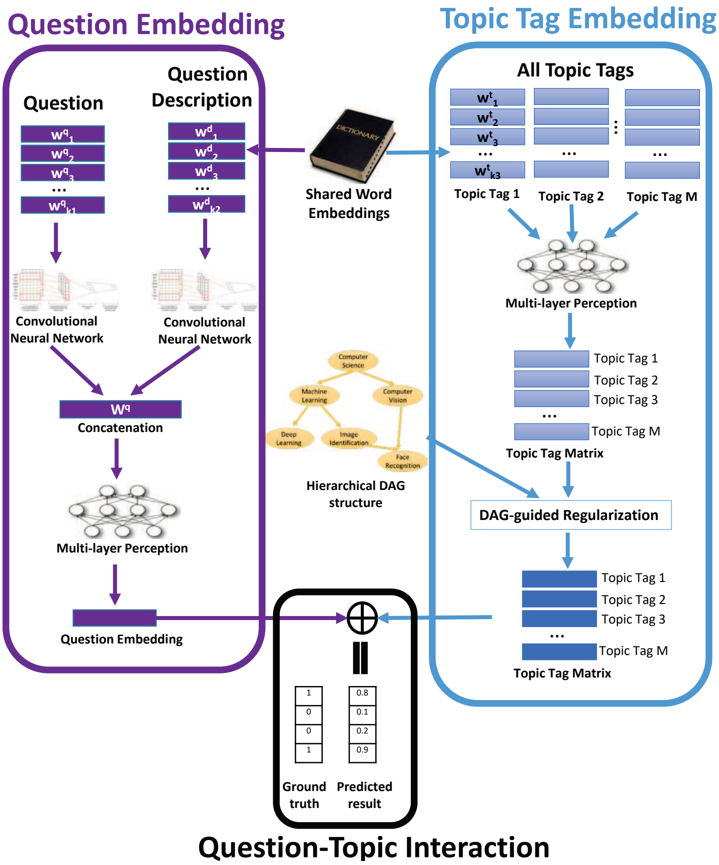


Fig. 4. Schematic illustration of our proposed PROFIT model. It consists of three components: question embedding, topic tag embedding and question-topic interaction.

language processing tasks [44, 50, 52], we leverage the CNN model to produce the fixed length vector representation of questions that preserve the semantic structure.In particular, we first build a very large vocabulary of phrases by segmenting all the sentences in our cQA set using the jieba tool[13]. In contrast with words, phrases are critical for capturing the lexical meanings. On the ground of word embedding techniques [2], we map the semantic meaning of phrases into a geometric space. This is accomplished by associating a numeric vector to every phrase in the dictionary, such that the distance (e.g. Euclidean distance or more commonly, the cosine distance) between any two vectors would capture part of the semantic relationship between the two associated phrases. The dictionary of phrases is shared between the question embedding learning and the topic tag embedding learning. We pre-train phrase embeddings instead of random initialization. Embeddings of phrases in a question concatenate together to form a matrix, representing the question. In our method, we use simple convolution layers with filters of multiple sizes on top of the learned embedding matrix. This extracts various high-level features and encodes semantic meaning of phrases by considering the intrinsic sequential information among phrases [15]. Our method in fact

[13] https://github.com/fxsjy/jieba.

addresses the word order problem of the Uni-gram model while avoiding dimensionality explosion of all the N-gram models.

In the cQA sites, apart from brief questions, users often describe their queries in detail, the so-called "question description", which actually encodes rich contexts and empowers the short questions. In our Dataset I, approximately 2,142,746 out of 2,999,965 questions have the associated descriptions (71.4%). In Dataset II, the ratio is around 67.4%. To make full use of the question descriptions, we devise a Siamese-style neural network whereby two disjoint CNNs are trained. These two CNNs share the same network structure, but not necessarily the identical parameters [38]. It is worth emphasizing that if a question has no description, we will by default treat the question itself as its description. After two CNNs, we concatenate the embeddings of the question and its description, followed by the multi-layer perception to project the ultimate question embedding into a 1,024-D space.

Because the questions and their descriptions are all of varying lengths, we pad short questions with dummy words and truncate very long questions. In this way, we can deal with questions with the same length. In our work, we set the threshold of question and question description lengths as 30 and 150, respectively[14]. It must be be noted that we do not add any max-pool or min-pool layer which is frequently utilized in visual computing to capture the spatial invariance, e.g., they detect a dog regardless of where in the given image the dog is located. Nevertheless, in the field of text understanding, the spatial location within the given question is of importance.

4.2.2 Topic Tag Embedding Layer. Analogous to the question embedding, we first form the embedding matrix for each topic tag. We then obtain a $M \times L \times D_p$ tensor, whereby M, D_p and L respectively denotes the number of topic tags, the dimension of phrase embedding, and the length threshold we set for each topic tag ($L = 5$ in this work). After the multi-layer perception, we reach a new $M \times 1024$ matrix, in which each row denotes a topic tag embedding.

As mentioned before, topic tags are not independent but hierarchically correlated. In our work, the inherent structural relatedness among topic tags is characterized via a DAG ontology $\mathcal{G}$, whereby the sibling and ancestor-descendent relationships are well-organized. Graph has been applied successfully in social networks [23] and recommender systems [21]. To learn a discriminant and robust embedding of each topic tag, we have to encode the ontology into our model with the expect that learned embeddings are capable of capturing the hierarchy among topic tags, such that the topic tags can reinforce each other to alleviate the problem of imbalanced topic tag distribution.

By observing and analyzing the given DAG carefully, we found that: 1) Nodes locating at a deeper layer of the DAG are less frequently used to label questions in the cQA sites and vice versa. For example, 70% of the DAG nodes in Dataset II are leaf nodes. They, however, only label 25% of the questions. Thereby, the internal nodes see more samples and contain richer information. Inspired by this observation, transferring knowledge from the parent nodes to the child ones is a natural way to alleviate the problem of imbalanced topic tag distribution. 2) As known, a child node in a DAG may have multiple parents. We find that a parent node usually captures only one aspect of its child node. For instance, considering "apple" as a child node, it has two parents "electronic product" and "fruit". And 3) the semantics expressed by different parents of the same child are often complementary rather than redundant or conflicting, and they together make up the overall semantic of

[14] According to our statistics, the average lengths of questions and question descriptions are 12.91 and 62.10 in Dataset I (13.12 and 56.07 in Dataset II), respectively.

the child node. In the light of this, we assume that the embedding of a child node can be approximated by a convex combination of the embeddings and parents of itself. It is a good way to consider a node itself for remembering the historical results during the updating. Based upon this assumption, we regularize the topic tag embedding learning,

$$\begin{cases} \mathbf{u_i} = \sum_{t_j \in \mathcal{C}(i)} \alpha_{ij} \mathbf{u_j}, \\ s.t \sum_{t_j \in \mathcal{C}(i)} \alpha_{ij} = 1, \alpha_{ij} \geq 0, \end{cases} \tag{1}$$

where $\mathbf{u_i} \in \mathbb{R}^D$ is a D-dimensional embedding vector of the node $t_i \in \mathcal{T}$, $\mathcal{C}(i)$ denotes a set of nodes containing node t_i and its parents, and $\alpha_{ij} \in \mathbb{R}^+$ refers to the attention weight on t_j when calculating t_i. The above formulation is transmissible, namely, beyond parents, a child node can indirectly benefit from its ancestors after a few updates. Inspired by the work in [6], we adopt the softmax function to estimate the attention weight α_{ij},

$$\begin{cases} \alpha_{ij} = \frac{\exp(f(\mathbf{u_i}, \mathbf{u_j}))}{\sum_{k \in \mathcal{C}(i)} \exp(f(\mathbf{u_i}, \mathbf{u_k}))}, \\ f(\mathbf{u_i}, \mathbf{u_j}) = \mathbf{z}^T \tanh(\mathbf{W}[\mathbf{u_i}, \mathbf{u_j}]^T + \mathbf{b}), \end{cases} \tag{2}$$

where $f(\mathbf{u_i}, \mathbf{u_j})$ measures how much $\mathbf{u_j}$ constitutes its child $\mathbf{u_i}$, which is calculated via multi-layer perception with a H-D hidden layer. The projecting matrix $\mathbf{W} \in \mathbb{R}^{H \times 2D}$, bias vector $\mathbf{b} \in \mathbb{R}^H$ and weight vector $\mathbf{z} \in \mathbb{R}^H$ are all parameters we need to learn.

4.2.3 Question-Topic Interaction Layer. Given the embedding of a question and the embeddings of all the topic tags, we perform interactions via dot production between the embedding vector and the embedding matrix. In this way, we obtain a score vector, whereby each element indicates the similarity degree between the given question and the corresponding topic tag. It is worth emphasizing that the element values may be out of the range of [0,1]. In the light of this, we leverage a sigmoid cross entropy loss to project the similarity score into [0,1],

$$\Phi = -\sum_i (\mathbf{y}_i * \log(s(\widehat{\mathbf{y}}_i)) + (1 - y) \log(1 - s(\widehat{\mathbf{y}}_i))), \tag{3}$$

where $s(\cdot)$ denotes the sigmoid function given the input vector. $\widehat{\mathbf{y}}_i \in \mathbb{R}^M$ and $\mathbf{y}_i \in \mathbb{R}^M$ are multi-hot vectors. Thereinto, $\mathbf{y}_i \in \mathbb{R}^M$ is the ground truth for the i-th question, whereby the j-th column is one if the j-th topic tag is used to label the i-th question; otherwise zero. $\widehat{\mathbf{y}}_i \in \mathbb{R}^M$ is the predicted result for our proposed PROFIT model.

4.2.4 Implementation Details. The aforementioned model is trained offline. After training, for a newly posted question, we can online calculate its embedding via the well-trained PROFIT model, and tag the newly posted question by measuring the embedding similarity between it and each topic tag, wherein the embedding of each topic tag is learned offline.

During the training phase, we feed our model batch by batch. Each batch comprises 128 question-description pairs, the associated topic tags, and one parents-child relation. After the word embedding layer, the batch of questions, question descriptions, and topic tags are denoted by $128 \times 30 \times D_p$, $128 \times 150 \times D_p$, and $M \times 5 \times D_p$ tensors, respectively. Thereinto, as aforementioned, 30, 150, and 5 respectively denotes the maximum length we set for each question, question description and topic tag. In addition, D_p refers to the embedding size, which is respectively set to be 128 and 256 for Dataset I and Dataset II. It is worth noting that we input all the M topic tags at one time and update the embeddings of topic tags batch by batch, whereby M stands for the number of topic tags in our DAG.

On the basis of the word embedding layer, we separately employ two one-layer CNNs to capture the high-level abstracts of the input question and its description. Each TextCNN is equipped with 256*5 filters in 5 distinct sizes [2, 3, 4, 5, 7]. Following the convolution, we use batch norm, relu and max-pooling. Thereafter, we obtain a $1,280$-D embedding vector for each question and its description, respectively. Next, we input the concatenation of the embeddings of a question and its description to a fully connected layer (size $2,560 \times 1,024$) followed by batch norm and Relu. We ultimately reach a $1,024$-D embedding vector to represent a question. Here we set the dropout rate as 0.5 for the fully connected layer, and we will detail why we choose this rate in the experimentations. Thus far, we have represented the batch of questions as a $128 \times 1,024$ matrix.

As to the topic embedding learning, we obtain a $M \times 5 \times D_p$ tensor after the word embedding layer. We then take the transpose of this tensor and reach a new $M \times D_p \times 5$ tensor. We input this tensor into a fully connected layer (size: 5×1), followed by a Relu activation function. We next lay another fully connected layer (size: $D_p \times 1024$; dropout rate: 0.5), followed by a batch norm. Up to now, we reach a $M \times 1024$ topic tag matrix. In fact, we only update a parents-child relation in each batch using Eqn.(1). To be more specific, we parse the given DAG structure and identify all the parents-child relations. We sequentially input all the relations one by one (i.e., batch by batch) to cover all the hierarchical relations.

Once we obtain the embeddings of 128 questions and all the M topic tags, we multiply each question embedding with the topic tag matrix to estimate the similarities between the given question and all the topic tags.

5 EXPERIMENTS

In this section, we conducted experiments to justify the effectiveness of our proposed PROFIT model and its components. Extensive experiments have been comparatively conducted over Dataset I and Dataset II.

5.1 Experimental Settings

Given a question and a set of topic tags, we targeted at generating an ordered list of tag candidates. This is in fact a ranking problem. To measure the performance of our model and the baselines, we adopted three metrics same as those in the Zhihu Machine Learning Challenge 2017: $precision_{rank}$, *recall* and $F\text{-}measure_{rank}$ and the standard metric $Precision@k$.

Traditionally, precision is defined as the fraction of the documents retrieved that are relevant to the user's need. Such definition, however, does not consider the position of the ranked documents. Inspired by the metric of Discounted cumulative gain (DCG) that is a measure of quality rating, we re-formulated the precision metric by placing stronger emphasis on retrieving the relevant documents and their positions. Formally, it is written as,

$$Precision_{rank}@k = \sum_{i=1}^{k} \frac{rel_i}{log(i+1)}, \tag{4}$$

where the relevance values of a topic tag in our work are binary, $rel_i \in \{0, 1\}$. If the i-th topic tag in the generated ranking list is used to label the given question, $rel_i = 1$; otherwise $rel_i = 0$. Noting that, $Precision_{rank}@k$ is possible to be larger than 1 according to our definition. This definition derives from the official website of Zhihu contest[15]. The reasons why we use these three 'non-standard' metrics are twofold: 1) These three metrics are more suitable for our task, as cQA sites usually recommend at most a fixed number of topic tags for each question (e.g., the number is 5 in Stack Overflow and Zhihu, namely, each question can be tagged with at most 5 topics). Accordingly, given a fixed k, it is more important to consider the ranking positions of the ground-truth topic tags. The traditional standard metrics, like precision and F-measure, are unable to capture the specific position information. And 2) existing studies that use the same dataset (i.e., Zhihu Contest) as ours, like [8], adopts the same evaluation metrics, i.e., $precision_{rank}$, $recall$, and $F\text{-}measure_{rank}$. Consequently, we used the same metrics to facilitate the comparisons.

The second metric in this work is $Recall@k$, which measures the fraction of the topic tags used to label the given question that is successfully ranked at the top k positions. The third metric is the $F\text{-}measure_{rank}$, which is the weighted harmonic mean of $precision_{rank}$ and $recall$, formulated as,

$$F\text{-}measure_{rank}@k = \frac{precision_{rank}@k * Recall@k}{precision_{rank}@k + Recall@k}. \tag{5}$$

This is also known as the F_1 measure, because $recall$ and $precision$ are evenly weighted. In this work, we set k as 5 and used $Precision_r$, $Recall$, and $F\text{-}measure_r$ to denote $Precision_{rank}@5$, $Recall@5$, and $F\text{-}measure_{rank}@5$, respectively. Besides, we used $P@5$ and $P@10$ to denote $Precision@5$ and $Precision@10$, respectively.

We randomly split all the questions in Dataset I and II into three chunks, respectively. To be more specific, in Dataset I, we had 2,700,000 questions for training, 149,965 for validation and 150,000 for testing. As for Dataset II, we used 1,307,023 questions for training, 192,977 for validation, and 207,023 for testing. The training set is used to learn our model, the validation set is to tune the optimal parameter settings, and the testing one is to report the final results.

5.2 Baselines

To demonstrate the effectiveness of our proposed PROFIT model, we compared it with the following state-of-the-art methods:

- **Matching**: This baseline is straightforward. We averaged the embeddings of all the words in each question-description pair and topic tag to represent the question and topic tag, respectively. Thereafter, we calculated their cosine similarity to search the top K related topic tags.
- **Naive Bayes**: It is a classical supervised-learning model for classification tasks [14]. In this baseline, we cast the question tagging problem into a regression task and each topic tag was treated as a category. We worked towards selecting the top K most relevant topic tags. We calculated the tf.idf of the short texts as their input features.
- **FastText**: This baseline scales a linear model to a very large corpus with a large output space in the context of text classification [13]. This is accomplished by enhancing the linear models with a rank constraint and a fast loss approximation. We could train fastText on more than one billion words in less than ten minutes by using a standard multicore CPU, while achieving performance on par with the state-of-the-art methods.
- **TextCNN**: Indeed, this baseline is a simple CNN with one layer of convolution on top of the word vectors obtained from an unsupervised neural language model [15]. It

[15] https://biendata.com/competition/zhihu/evaluation/.

Table 3. Performance comparison between our proposed model and several state-of-the-art baselines over two datasets, measured by four metrics.

Methods	Dataset I					Dataset II				
	$Precision_r$	*Recall*	$F\text{-}measure_r$	*P@5*	*P@10*	$Precision_r$	*Recall*	$F\text{-}measure_r$	*P@5*	*P@10*
Matching	0.1938	0.0980	0.0651	0.0512	0.0276	0.0242	0.0150	0.0093	0.0062	0.0021
Naive Bayes	0.2895	0.1289	0.0892	0.0777	0.0499	0.2615	0.1508	0.0957	0.0497	0.0193
L2R	0.8895	0.3876	0.2700	0.1548	0.1166	0.7384	0.4118	0.2643	0.1139	0.0974
FastText	1.4026	0.5385	0.3891	0.2472	0.1531	1.1427	0.5400	0.3667	0.1975	0.1167
RNN	1.4011	0.5372	0.3883	0.2469	0.1529	1.2522	0.5837	0.3981	0.2139	0.1264
TextCNN	1.4070	0.5410	0.3907	0.2502	0.1546	1.2549	0.5867	0.3998	0.2161	0.1269
RegionEmb	1.4033	0.5404	0.3901	0.2487	0.1544	1.2457	0.5864	0.3987	0.2146	0.1266
PBAM	1.4031	0.5390	0.3894	0.2473	0.1533	1.2359	0.5820	0.3957	0.2128	0.1256
DGCNN	1.4103	0.5423	0.3917	0.2519	0.1549	1.2620	0.5870	0.4006	0.2169	0.1273
PROFIT	**1.4257**	**0.5490**	**0.3964**	**0.2557**	**0.1567**	**1.3059**	**0.6108**	**0.4162**	**0.2260**	**0.1312**

achieves great success in natural language processing tasks like sentiment analysis[52], machine translation[44], and text summarization[50].

- **TextRNN**: This baseline integrates a recurrent structure to capture contextual information as far as possible when learning word representations [17].
- **L2R**: This baseline presents a convolutional neural network architecture for re-ranking the pairs of short texts, where it learns the optimal representation of text pairs and a similarity function to relate them in a supervised way from the available training data[34]. This network takes only words as the input, thus requiring minimal preprocessing. Specifically, we treated each question and its tag as a positive pair and constructed the negative pair by randomly sampling a tag not associated with the given question.
- **RegionEmb**: This baseline represents each word with two parts: the embedding of the word itself, and a weighting matrix characterizing its interaction with the local context [30]. Besides, it outperforms existing methods in the task of text classification on several benchmark datasets.
- **PBAM**: It is a recently proposed deep neural network for question tagging, which utilizes a position-based attention mechanism to model the question text [39].
- **DGCNN**: This baseline is proposed for hierarchical text classification, which utilizes a graph convolutional network to learn the original raw text and the recursive hierarchical segmentation for relationship modeling [29]. Now we used the model for qustion tagging which also has hierarchical relationships among tags.
- **PROFIT**: This is our proposed end-to-end deep interactive model for question tagging.

To ensure a fair comparison, all the above models were trained on the same training set and justified on the same testing one. In addition, all the models were carefully tuned to reach their optimal settings and their best performance in terms of multiple metrics were reported.

5.3 Overall Comparison

The comparative results over two datasets between our proposed PROFIT model and several state-of-the-art baselines for question tagging are summarized in Table 3. Let us analyze the experimental results over Dataset I first: 1) As expected, the unsupervised learning method, i.e, Matching, is the worst, since it does not encode any label information. 2) All the deep learning models remarkably outperform the shallow learning one. Deep learning algorithms seek to exploit the unknown structure in the input distribution in order to discover better representations, often at multiple levels, with higher-level learned features defined in terms of lower-level features. Automatically learning features at multiple levels of abstraction allow the classifier model to learn complex functions by mapping the input of short texts to the output label directly from data, which do not completely depend on the human-crafted. Whereas, the shallow learning methods should heavily rely on the human-crafted features. 3) Amongst the deep models, FastText, TextCNN, TextRNN, RegionEmb, PBAM, and DGCNN achieve comparable performance which is much better than that of L2R. This is probably due to the fact that for each positive sample, we only randomly constructed a negative one when training L2R; whereas the other three deep models can sufficiently incorporate the information from the negative samples. 4) Our model is consistently better than all the baselines. This is because standing on the basis of deep models, we also considered the label semantics and the label hierarchies. Beyond the traditional labeling models which understand the category labels via their associated samples, our model directly encodes the label semantics during the learning. Meanwhile, the label hierarchies convey the label correlations, enabling knowledge transfer among various categories, especially from top to down. And 5) compared with the question tagging baseline PBAM, our method achieves the better performance, which may be due to the fact that PBAM overlooks both the label semantic information and the hierarchical relationships among question topics. Moreover,compared with DGCNN, which indeed also explores the label via the recursive hierarchical segmentation, our PROFIT model has better performance. This verifies he effectiveness of our DAG mechanism to transform knowledge from parent nodes to child nodes, which contributes to overcoming the unbalance of topic tags. The standard $Precision@K$ metric shows consistent results with the other three metrics, which verifies the performance of our method. Besides, we can find that $Precision_r$ is higher than $P@5$ of the same method generally due to the difference of their formulas.

As to the performance comparison over Dataset II, we observed the following points: 1) The overall trend is almost the same to the Dataset I. Specifically, the unsupervised method is the worst, the shallow learning method is the last but one, and the deep model is the best. 2) It is worth highlighting that the performance of FastText drops considerably, much worse as compared to other deep models. This is because other deep models like TextCNN and TextRNN are able to capture the sequential characteristics of texts, which is even more obvious in the larger and more sparse Dataset II. As mentioned before, Dataset II has around 13,000 labels, more than six times of the Dataset I. However, the average label frequency of Dataset II is only around 229, which is 3,861 in Dataset I. And 3) our model is much more robust over Dataset II as compared to the other competitors. This indicates the superiority of integrating the DAG structure into our model, which is able to transfer the knowledge among labels and hence boost the performance, especially for those labels with sparse samples. Statistically, 70% of the DAG nodes are the leaf ones, whereby the average frequency of leaf nodes and the internal nodes is 84 and 524, respectively in Dataset II. In other words, it is very hard, if not impossible, to learn the discrimination of the leaf nodes. With the help of DAG structure, our model is capable of transferring the knowledge from the up ancestor nodes to down descendants. When it comes to Dataset I, the frequency of the leaf node is 2,656 on average, which is more than enough to learn a robust model for each label. That is why our model demonstrates much better performance over Dataset II.

It is worth noting that although the Dataset I is released by Zhihu Machine Learning Challenge 2017, it is intractable to compare our proposed PROFIT model with the champion solution of this Challenge, as Dataset I is released as the training set of the Challenge, while

Table 4. Component-wise validation of our proposed PROFIT model by eliminating one component each time. We use No_DES and No_DAG to denote new models without considering the question descriptions and the DAG structure, respectively.

Methods	Dataset I					Dataset II				
	$Precision_r$	*Recall*	$F\text{-}measure_r$	*P@5*	*P@10*	$Precision_r$	*Recall*	$F\text{-}measure_r$	*P@5*	*P@10*
No_DES	1.3697	0.5265	0.3803	0.2381	0.1495	1.2559	0.5896	0.4012	0.2173	0.1274
No_DAG	1.4203	0.5450	0.3939	0.2532	0.1556	1.2769	0.5962	0.4064	0.2201	0.1281
PROFIT	**1.4257**	**0.5490**	**0.3964**	**0.2557**	**0.1567**	**1.3059**	**0.6108**	**0.4162**	**0.2260**	**0.1312**

Table 5. Performance comparison between different DAG encoding methods. For simplicity, we respectively denote Father=Sum(Children)+Itself as Down-Top, Child=Sum(Fathers)+Itself as Top-Down, and the fusion of the two methods as Top-Down&Down-Top.

Method	Dataset I					Dataset II				
	$Precision_r$	*Recall*	$F\text{-}measure_r$	*P@5*	*P@10*	$Precision_r$	*Recall*	$F\text{-}measure_r$	*P@5*	*P@10*
Down-Top	1.4126	0.5444	0.3929	0.2526	0.1552	1.2827	0.6023	0.4098	0.2232	0.1292
Top-Down	**1.4257**	**0.5490**	**0.3964**	**0.2557**	**0.1567**	**1.3059**	**0.6108**	**0.4162**	**0.2260**	**0.1312**
Top-Down &Down-Top	1.4179	0.5462	0.3943	0.2535	0.1559	1.3003	0.6104	0.4154	0.2252	0.1304

the exact online testing set is unavailable to us. In fact, we have studied the solutions of the top 5 teams and noticed that they mainly adopted the ensemble strategy with the basic models, like FastText, TextRNN, and TextCNN, to fulfil the task. Accordingly, we have introduced these basic models as the baselines in this work, and our PROFIT model shows the superiority over these baselines. Moreover, we further checked our PROFIT model with the ensemble manner and achieved more superior performance (i.e., 0.4421) with respect to F-measure on Dataset I.

5.4 On the Component-wise Validation

In this subsection, we conducted experiments to answer two research questions: 1) Does the question descriptions add value to our model? And 2) how much help does our model get from the DAG structure?

In our PROFIT model, we devised a Siamese-style neural network to make full use of the brief questions and their long descriptions, whereby two disjoint CNNs were trained. These two CNNs share the same network structure, but not necessarily the identical parameters. To well answer the first research question, we eliminated one brunch of the Siamese-style neural network and only kept the CNN corresponding to the brief question. As to answer the second research question, we eliminated the DAG-guided regularizer from our model and ensured the left unchanged.

The experimental results are summarized in Table 4. We used No DES and No DAG to denote new models without considering question descriptions and the DAG structure, respectively. It can be seen that No_DES is the worst model across two datasets. Such phenomenon clearly reflects that the question descriptions contain rich context information which plays a pivotal role in enhancing the question representation learning. In addition, we noted that difference gap between our PROFIT model and the No_DAG one is widened

Table 6. Illustration of the attentive weights for node representation in the given DAG. We randomly select ten nodes with one to five parents and display their weights in the convex combination.

Id	Examples
1	**Apple Store**=0.64*Apple Store+0.36*Apple Inc.
2	**SoftBank**=0.43*SoftBank+0.57*The Japanese group
3	**Airport**=0.68*Airport+0.23*public space+0.09*public building
4	**Linux**=0.67*Linux+0.25*open-source software+0.08*OS
5	**Larry Page**=0.65*Larry Page+0.23*Google+0.09*Entrepreneur+0.03*Alphabet
6	**Nokia**=0.67*Nokia+0.26*Mobile phone manufacturer+0.04*Technology company+0.03* Listed company
7	**Milk tea**=0.42* Milk tea+0.29*Milk+0.14*Tea+0.11*Drink+0.04*Soft Drinks
8	**Library**=0.48*Library+0.23*public space+0.19*public building+0.09*Book+0.01*Organization
9	**OneNote**=0.38*OneNote+0.37*Note+0.11*Note taking app+0.08*PKM+0.05*GDT+0.01*Microsoft Office
10	**Worktile**=0.30*Worktile+0.32*Teamwork equipment+0.18*OA+0.09*Office software +0.06*PM+0.05*PM system

from Dataset I to Dataset II. This further confirms our analysis before that our PROFIT model is much more robust when applied to a large-scale and sparse dataset.

5.5 On the DAG Structure

In our PROFIT model, we assumed that the embedding of a child node can be approximated by a convex combination of the embeddings of itself and its parents. Two natural questions are about how to represent a parent node by a convex combination of itself and its children and utilize the two representing methods together. In this subsection, we carried out experiments over two datasets to validate these different representations of a node.

The comparison results among various DAG encoding methods are displayed in Table 5. To facilitate the notation, we denoted Father=Sum(Children)+Itself as Down-Top, which vividly demonstrates that the knowledge is transferred from down to top. Analogously, we used Top-Down to represent Child=Sum(Fathers)+Itself. Besides, we used Top-Down&Down-Top to represent the fusion of the two methods. From this table, it is obvious that the Top-Down method is consistently superior to the Down-Top method regarding all the metrics across two datasets, especially on Dataset II. This phenomenon is caused by the different frequency distribution between the child nodes and the parent nodes in our datasets, namely nodes locating at a deeper layer of the DAG are less frequently used to label questions in the social QA sites and vice versa. According to our statistics, although 70% of the DAG nodes are leaf nodes, they label only 25% of all the archived questions. Therefore, the nodes at higher layers have more positive samples and hence hide richer information. That is why transferring the knowledge from the parent nodes to the child ones is an effective way. Even so, it is worth mentioning that in some cases whereby the leaf nodes are more frequently used, the Down-Top method may be an optimal option. And It is worth mentioning that the fusion method (i.e., "Down-Top+Top-Down") performs better than "Down-Top" but worse than "Top-Down". The inferior performance of the fusion method to "Top-Down" suggests that transferring knowledge from down to top can interfere top-down knowledge propagation. Such phenomenon may be caused by the different frequency distribution between the child nodes and the parent nodes in our datasets, namely nodes locating at a deeper layer of the DAG are less frequently used to label questions in the cQA sites and vice versa. According to our statistics, although 70% of the DAG nodes are leaf nodes, they label only 25% of all the archived questions. Therefore, the nodes at higher layers have more positive samples and hence hide richer information. That is why it is an effective way to transfer knowledge from the parent nodes to the child ones.

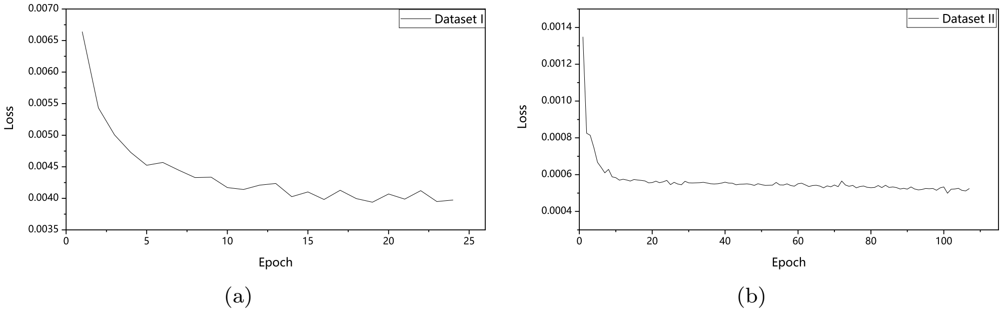


Fig. 5. Convergence analysis of our model over two datasets by measuring the loss decrease with respect to the number of epochs. (a) Loss curve over Dataset I; and (b) loss curve over Dataset II.

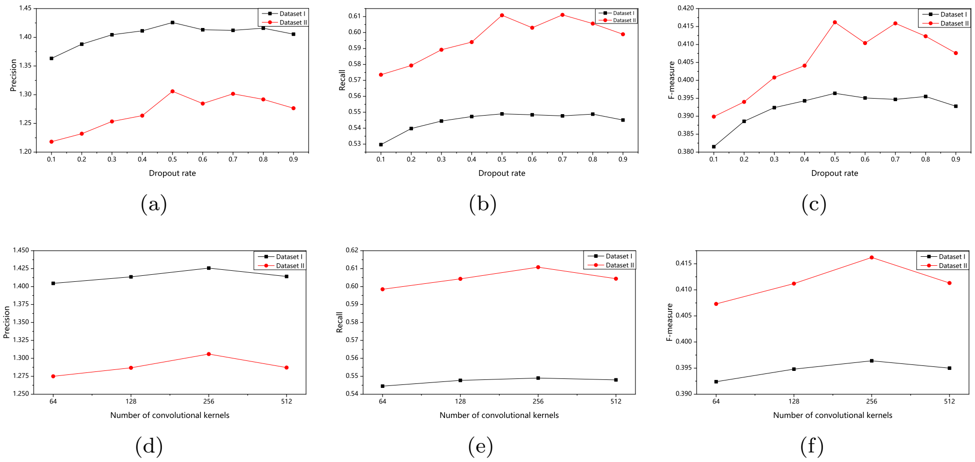


Fig. 6. Parameter analysis of our PROFIT model over two datasets. (a)-(c) performance of our model in terms of three metrics over two datasets by varying the dropout rate. And (d)-(f) performance of our model in terms of three metrics over two datasets by varying the number of convolutional kernels.

5.6 Illustration of Attentive Weights

As analyzed before, a child node may have multiple parents. On average, a child in Dataset I and II has 1.33 and 1.35 parents, respectively. We noted that a parent node usually captures only one aspect of its child node. Considering “Library” as an example, it inherits from four parents “Public Space”, “Public Building”, “Book”, and “Organization”. Each parent usually constitutes one aspect of its child with a certain degree. That was why we leveraged the attentive weighting mechanism in learning the child node representation.

To intuitively illustrate the attention results, we randomly selected ten nodes with one to five parents and listed their convex combination in Table 6. From the selected examples in Table 6, we had the following observations: 1) The weights of the children themselves are the largest ones in most cases. This indicates that beyond the knowledge inherited from the ancestor nodes, child nodes try their best to maintain their own information. And 2) it is consistent with our assumption that different parents contribute differently to their child nodes, and some even tend to have zero contribution, like “Soft Drinks” to its child “Milk Tea”.

Table 7. Case study. The test samples of PROFIT and some best baselines on Dataset II.

Questions and associated topic tags	Predicted topic tags			
	TextCNN	PBAM	DGCNN	PROFIT
Will Microsoft try to save Internet explorer? What measures will Microsoft take? **correct topic tags:, Microsoft, Internet Explorer, Browser**	**Microsoft**, Internet Explorer 10, Microsoft(China), **Internet Explorer**, Internet Explorer 9	**Microsoft**, Microsoft Windows, Windows 10, Microsoft Edge, **Internet Explorer**	**Microsoft**, OS, **Internet Explorer**, Microsoft(China), Internet Explorer 9	**Microsoft**, **Browser**, **Internet Explorer**, Microsoft Services, PC Browser
I want a powerful digital video, which can take good photos and videos that won't break my bank. What are your recommendations? **correct topic tags: Digital Video, Digital Product, Digital, DV**	**Digital**, Camera, Video Equipment, **Digital Video**, Vidicon	DSLR, Digital Camera, Electronic Products, **Digital Product**, Camera	**Digital Video**, Camera, Camera Company, **Digital**, **Digital Product**	**Digital Video**, **Digital Product**, Digital Camera, **DV**, **Digital**
What are the reasons why people like Grey's Anatomy? **correct topic tags: Grey's Anatomy, US TV Series**	**Grey's Anatomy**, **US TV Series**, Actor, TV Series Recommendation, Actress	**Grey's Anatomy**, Hospital, TV Play, **US TV Series**, Gender Relations	**Grey's Anatomy**, **US TV Series**, TV Series Recommendation, US TV Series, Japanese TV Series	**Grey's Anatomy**, US TV Series Recommendation, **US TV Series**, Doctor, TV Play

5.7 Convergence and Parameter Analysis

To demonstrate the convergence of our proposed PROFIT model, we plotted the loss curves over two datasets with respect to the number of epochs in Figure 5(a) and 5(b). It can be seen that our PROFIT model is able to converge over two datasets within less than 20 epochs. Notably, it tends to be slower on Dataset II, as compared to Dataset I. This is because Dataset II contains much more topic tags. It is mentioned that our model takes about 55 minutes and 58 minutes every epoch in the training phase for Dataset I and Dataset II, and 0.12ms and 0.83ms for one question in the testing phase for Dataset I and Dataset II, respectively.

We also studied the performance of our PROFIT model regarding the varying dropout rates and the number of convolutional kernels (also called filters). The experimental results are demonstrated in Figure 6. From Figure 6(a) to 6(c), we can see that our PROFIT model reaches the optimal performance when dropping out half neurons during the training, no matter in Dataset I or II. This indicates that our model has great transportability. Surprisingly, the recall@5 of our PROFIT model over Dataset II is much higher than that over Dataset I, although Dataset II is much larger and more sparse. After analyzing the data, we found that each question on average is annotated with 2.57 and 2.07 topic tags in Dataset I and II, respectively. Also, we noted that our model is capable of ranking 1.41 and 1.26 right topic tags on average at the top five positions over Dataset I and II, respectively, with only a slight difference. This statistic reveals that our model is robust and hence performs well over a challenging dataset.

As discussed before, on the basis of the word embedding layer, we separately employed two one-layer CNNs to capture the high-level abstracts of the input question and their description. Each TextCNN is equipped with K*5 convolutional kernels (filters) in 5 distinct sizes [2,3,4,5,7]. We varied K and recorded the performance of our model as illustrated in Figure 6(d)- 6(f). It is demonstrated that our model performs the best once K is at 256. This experimental result tells us that more kernels are not necessarily to add value, since we have to learn more parameters which require more samples.

5.8 Case Study

To gain more deep insights about the performance of our proposed PROFIT model in question tagging, we listed several example questions with their predicted topic tags via different methods in Table 7. From examples in Table 7, we had the following observations: 1) Almost all the predicted topic tags by our PROFIT model are semantically related to the question, although some of them are incorrect as compared to the ground truth. This reflects that our model can recommend semantically related labels, while the accurate question tagging is still challenging due to the fact that the correct topic tags are easy to be overwhelmed by semantically related tags. For example, PBAM recommends "Microsoft Edge" for the first question while misses the ground truth tag "Browser", which may be due to the semantic correlation between Microsoft Edge and Microsoft. 2) Our PROFIT model performs better than other baselines in the case study, which is consistent with the aforementioned information retrieval metrics. For example, only our PROFIT model manages to predict the "DV" tag for the second question. According to our statistics, "DV" is a leaf node in the DAG with limited samples. This demonstrates the advantage of taking into account the DAG structure to transform knowledge from the inter nodes to leaf nodes.

6 CONCLUSION AND FUTURE WORK

This paper presents an end-to-end deep interactive embedding model to label questions with topic tags in social QA sites, whereby the topic tags are pre-defined and pre-organized into a DAG structure. Instead of applying the pure classification models, we actually cast the question tagging task into a ranking problem. This deep model jointly learns the embeddings of questions and topic tags by projecting them into the same semantic space and then performs the interaction for similarity measure. During the learning, it leverages the DAG structure of topic tags to regularize their hierarchical relations for discriminative embedding learning, which is able to address the problem of imbalanced topic distribution by transferring the knowledge among topic tags. To justify our model, we conducted extensive experiments over two large-scale datasets: One is a benchmark dataset manually pruned for a public contest (Dataset I), and the other is a real-world dataset without any preprocessing (Dataset II). We notice that the experimental results support the following points: 1) Our proposed model remarkably outperforms several state-of-the-art methods for the question tagging task; 2) the DAG structure adds value to the topic tag embedding learning; 3) question descriptions are able to strengthen the discriminations of the short questions. And 4) as compared to the classification problem, our PROFIT model benefits from the topic semantics by incorporating the interaction component.

In future, we plan to deepen and widen our work from the following aspects: 1) Due to the practical concern that topic tags in many cQA cites, like Stack Overflow, are not pre-organized explicitly as a hierarchical structure, we will extend our method to these more challenging cQA sites. For example, we can conduct a topic tag graph by linking the related tags and synonymous tags. Besides, existing topic tag structures, like the Zhihu topic tag structure, can be transformed to the cQA cites that without pre-organized hierarchical structure. And 2) there still remains much improving space for enhancing the performance of DAG modeling. One possible improvement can be made by treating the topic hierarchical structure as a graph. Considering the compelling success of Graph Convolutional Neural Networks (GCN) in various machine learning tasks, we will explore its potential in the context of question-topic embedding learning.

7 ACKNOWLEDGEMENT

This work is supported by the National Natural Science Foundation of China, No.: 61772310, No.:U1936203 and No.:U1836216; the Shandong Provincial Natural Science and

Foundation, No.: ZR2019JQ23; the Innocation Teams in Colleges and Universities in Jinan, No.:2018GXRC014.

REFERENCES

[1] Fabiano Belém, Eder Martins, Tatiana Pontes, Jussara Almeida, and Marcos Gonçalves. 2011. Associative tag recommendation exploiting multiple textual features. In *Proceedings of the international ACM SIGIR conference on Research and development in information retrieval*. 1033–1042.

[2] Yoshua Bengio, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3, 6 (2003), 1137–1155.

[3] Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. 2011. Large-scale Question Classification in cQA by Leveraging Wikipedia Semantic Knowledge. In *Proceedings of the ACM International Conference on Information and Knowledge Management*. 1321–1330.

[4] Xin Cao, Gao Cong, Bin Cui, Christian S Jensen, and Quan Yuan. 2012. Approaches to exploring category information for question retrieval in community question-answer archives. *ACM Transactions on Information Systems (TOIS)* 30, 2 (2012), 7.

[5] Zhiyong Cheng, Xiaojun Chang, Lei Zhu, Rose C Kanjirathinkal, and Mohan Kankanhalli. 2019. MMALFM: Explainable recommendation by leveraging reviews and images. *ACM Transactions on Information Systems (TOIS)* 37, 2 (2019), 16.

[6] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F. Stewart, and Jimeng Sun. 2017. GRAM: Graph-based Attention Model for Healthcare Representation Learning. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*. 787–795.

[7] Erik Choi, Vanessa Kitzie, and Chirag Shah. 2012. Developing a typology of online Q&A models and recommending the right model for each question type. *Proceedings of the American Society for Information Science and Technology* 49, 1 (2012), 1–4.

[8] Cunxiao Du, Zhaozheng Chen, Fuli Feng, Lei Zhu, Tian Gan, and Liqiang Nie. 2019. Explicit interaction model towards text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 6359–6366.

[9] Zan Gao, Deyu Wang, Xiangnan He, and Hua Zhang. 2018. Group-Pair Convolutional Neural Networks for Multi-View Based 3D Object Retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[10] Junwei Han, Le Yang, Dingwen Zhang, Xiaojun Chang, and Xiaodan Liang. 2018. Reinforcement cutting-agent learning for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 9080–9089.

[11] Paul Heymann, Daniel Ramage, and Hector Garcia-Molina. 2008. Social tag prediction. In *Proceedings of the international ACM SIGIR conference on Research and development in information retrieval*. 531–538.

[12] Heyan Huang, Xiaochi Wei, Liqiang Nie, Xianling Mao, and Xin-Shun Xu. 2018. From Question to Text: Question-Oriented Feature Attention for Answer Selection. *ACM Transactions on Information Systems (TOIS)* 37, 1 (2018), 6.

[13] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. 427–431.

[14] Sang Bum Kim, Kyoung Soo Han, Hae Chang Rim, and Sung Hyon Myaeng. 2006. Some Effective Techniques for Naive Bayes Text Classification. *IEEE Transactions on Knowledge and Data Engineering* 18, 11 (2006), 1457–1466.

[15] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1746–1751.

[16] Jeongwoo Ko, Luo Si, Eric Nyberg, and Teruko Mitamura. 2010. Probabilistic models for answer-ranking in multilingual question-answering. *ACM Transactions on Information Systems (TOIS)* 28, 3 (2010), 16.

[17] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

[18] Yann Lcun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11, 2278–2324.

[19] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the International Conference on International Conference on Machine Learning.* II–1188.

[20] Chenliang Li, Shiqian Chen, Jian Xing, Aixin Sun, and Zongyang Ma. 2018. Seed-Guided Topic Model for Document Filtering and Classification. *ACM Transactions on Information Systems (TOIS)* 37, 1 (2018), 9.

[21] Yongqi Li, Meng Liu, Jianhua Yin, Chaoran Cui, Xin-Shun Xu, and Liqiang Nie. 2019. Routing Micro-videos via A Temporal Graph-guided Recommendation System. In *Proceedings of the 27th ACM International Conference on Multimedia.* 1464–1472.

[22] Yandong Liu and Eugene Agichtein. 2008. On the evolution of the yahoo! answers QA community. In *Proceedings of the international ACM SIGIR conference on Research and development in information retrieval.* 737–738.

[23] Liqiang Nie, Xuemeng Song, and Tat-Seng Chua. 2016. Learning from multiple social networks. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 8, 2 (2016), 1–118.

[24] Liqiang Nie, Meng Wang, Luming Zhang, Shuicheng Yan, Bo Zhang, and Tat-Seng Chua. 2015. Disease inference from health-related questions via sparse deep learning. *IEEE Transactions on knowledge and Data Engineering* 27, 8 (2015), 2107–2119.

[25] Liqiang Nie, Yi Liang Zhao, Xiangyu Wang, Jialie Shen, and Tat Seng Chua. 2014. Learning to Recommend Descriptive Tags for Questions in Social Forums. *ACM Transactions on Information Systems (TOIS)* 32, 1 (2014), 1–23.

[26] Kyosuke Nishida and Ko Fujimura. 2010. Hierarchical Auto-tagging: Organizing Q&#38;A Knowledge for Everyone. In *Proceedings of the ACM International Conference on Information and Knowledge Management.* 1657–1660.

[27] Alberto Paccanaro and Geoffrey E. Hinton. 2002. Learning Distributed Representations of Concepts Using Linear Relational Embedding. *IEEE Transactions on Knowledge and Data Engineering* 13, 2 (2002), 232–244.

[28] Aditya Pal. 2015. Metrics and algorithms for routing questions to user communities. *ACM Transactions on Information Systems (TOIS)* 33, 3 (2015), 14.

[29] Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the Conference on World Wide Web Conference on World Wide Web.* 1063–1072.

[30] Chao Qiao, Bo Huang, Guocheng Niu, Daren Li, Daxiang Dong, Wei He, Dianhai Yu, and Hua Wu. 2018. A new METHOD OF REGION EMBEDDING FOR TEXT CLASSIFICATION. In *International Conference on Learning Representations.*

[31] Bo Qu, Gao Cong, Cuiping Li, Aixin Sun, and Hong Chen. 2014. An evaluation of classification models for question topic categorization. *Journal of the Association for Information Science and Technology* 63, 5 (2014), 889–903.

[32] E. M. Rodrigues, N. Milic-Frayling, and B. Fortuna. 2008. Social Tagging Behaviour in Community-Driven Question Answering. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology.* 112–119.

[33] Avigit K. Saha, Ripon K. Saha, and Kevin A. Schneider. 2013. A discriminative model approach for suggesting tags automatically for Stack Overflow questions. In *Mining Software Repositories.* 73–76.

[34] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *Proceedings of the ACM International Conference on Research and Development in Information Retrieval.* 373–382.

[35] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops.* 806–813.

[36] Amit Singh and Karthik Visweswariah. 2011. CQC: Classifying Questions in CQA Websites. In *Proceedings of the ACM International Conference on Information and Knowledge Management.* 2033–2036.

[37] Ivan Srba and Maria Bielikova. 2016. A comprehensive survey and classification of approaches for community question answering. *ACM Transactions on the Web (TWEB)* 10, 3 (2016), 18.

[38] Ting Su, Craig Macdonald, and Iadh Ounis. 2019. Ensembles of Recurrent Networks for Classifying the Relationship of Fake News Titles. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 893–896.

[39] Bo Sun, Yunzong Zhu, Yongkang Xiao, Rong Xiao, and Yungang Wei. 2018. Automatic question tagging with deep neural networks. *IEEE Transactions on Learning Technologies* 12, 1 (2018), 29–43.

[40] Saori Suzuki, Shin'ichi Nakayama, and Hideo Joho. 2011. Formulating effective questions for community-based question answering. In *Proceedings of the international ACM SIGIR conference on Research and development in information retrieval*. 1261–1262.

[41] Nam Khanh Tran and Claudia Niedereée. 2018. Multihop attention networks for question answer matching. In *Proceedings of the international ACM SIGIR conference on Research and development in information retrieval*. ACM, 325–334.

[42] Kateryna Tymoshenko and Alessandro Moschitti. 2018. Shallow and Deep Syntactic/Semantic Structures for Passage Reranking in Question-Answering Systems. *ACM Transactions on Information Systems (TOIS)* 37, 1 (2018), 8.

[43] Gang Wang, Konark Gill, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. 2013. Wisdom in the Social Crowd: An Analysis of Quora. In *Proceedings of the International Conference on World Wide Web*. 1341–1352.

[44] Hongbin Wang, Hongxu Hou, Jing Wu, Jinting Li, and Wenting Fan. 2017. Exploring Different Granularity in Mongolian-Chinese Machine Translation Based on CNN. In *International Conference on Parallel and Distributed Computing, Applications and Technologies*. 112–116.

[45] Shaowei Wang, D Lo, B Vasilescu, and A Serebrenik. 2014. EnTagRec: An Enhanced Tag Recommendation System for Software Information Sites. *Proceedings of the IEEE International Conference on Software Maintenance and Evolution* (2014), 291–300.

[46] Shaowei Wang, David Lo, Bogdan Vasilescu, and Alexander Serebrenik. 2018. EnTagRec ++ : An enhanced tag recommendation system for software information sites. *Empirical Software Engineering* 23, 2 (2018), 800–832.

[47] Xin Xia, David Lo, Xinyu Wang, and Bo Zhou. 2013. Tag recommendation in software information sites. In *Proceedings of the Working Conference on Mining Software Repositories (MSR)*. 287–296.

[48] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 582–590.

[49] Jian Zhang, Hailong Sun, Yanfei Tian, and Xudong Liu. 2018. Poster: Semantically Enhanced Tag Recommendation for Software CQAs via Deep Learning. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)*. 294–295.

[50] Y. Zhang, M. J. Er, R. Zhao, and M Pratama. 2016. Multiview Convolutional Neural Networks for Multidocument Extractive Summarization. *IEEE Transactions on Cybernetics* 47, 10 (2016), 3230 – 3242.

[51] Z. Zhang, M. Lyons, M. Schuster, and S. Akamatsu. 1998. Comparison Between Geometry-Based and Gabor-Wavelets-Based Facial Expression Recognition Using Multi-Layer Perceptron. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, 1998. Proceedings*. 454–459.

[52] Wei Zhao, Ziyu Guan, Long Chen, Xiaofei He, Deng Cai, Beidou Wang, and Quan Wang. 2018. Weakly-Supervised Deep Embedding for Product Review Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering* 30, 1 (2018), 185–197.

[53] Tom Chao Zhou, Michael R Lyu, and Irwin King. 2012. A classification-based approach to question routing in community question answering. In *Proceedings of the 21st International Conference on World Wide Web*. 783–790.