MRT: Tracing the Evolution of Scientific Publications

Da Yin, Weng Lam Tam, Ming Ding, and Jie Tang

Abstract—The fast development of science and technology is accompanied by the booming of cutting edge research. Researchers need to digest more and more recently published publications in order to keep themselves up to date. This becomes tough in particular with the prevalence of preprint publishing such as arXiv, where inspiring works could come out without being peer-reviewed. Is that possible to design an automatic system to help researchers quickly gain a glimpse of a piece of work or gain useful background knowledge for deeply understanding it? To this end, we proposed a practical framework called **M**aster **R**eading **T**ree (MRT) to trace the evolution of scientific publications. In this framework, we can build annotated *evolution roadmaps* for publications and identify important previous works or evolution tracks by generating expressive embeddings and clustering them into various groups. With comprehensive evaluations, our proposed framework demonstrates its superior capability in capturing underlying relations behind publications over several baseline algorithms. Finally, we integrated the proposed MRT framework on AMiner, an online academic platform, where users can generate roadmaps using MRT for free and their interactions are further used to refine the model.

Index Terms—Data Mining, Information Retrieval, Document Representation, Network Embedding, Document Clustering

1 Introduction

Science evolution has been faster than ever before. The rate of growth for journal numbers has increased from 3% to 6% over the last decade, with around 3 million articles published in 2018 [1]. At top conferences in computer science such as NeurIPS or CVPR, the number of accepted papers has constantly been increasing, which reached over one thousand in 2019, over three times larger than the number ten years before. Even scanning over the latest publications in related fields could cost experienced researchers days and weeks, not to mention reading in detail. The amount of materials to read and comprehend has also been the main obstacle for novice researchers and intelligence analysts to entering new fields quickly, as mentioned in [2].

To help readers find the main topics among such massive information, concept extraction and taxonomy construction [3], [4] have been studied for identifying field structures. *Algorithm Roadmap* [2] is also proposed to sketch the dynamics of algorithms in specific areas. However, previous works mainly focus on extracting existing terms in publications and making over-simplified summarization on top of them. Analysts may be able to quickly locate their desired topics or related publications, but it is hard for researchers to reason how ideas evolved.

Take BERT [5] as an example (as shown in Figure 1). With previous techniques, it is easy to figure out the fact that BERT is one of the most influential works in the Natural Language Processing field recently, which outperforms many prior state-of-the-art models on several different tasks. However, for a fresh Ph.D. student just entering the NLP area, after reading the BERT paper, he might be wondering: what are the origins of the BERT idea? The use of transformer in BERT originated from *Attention is all you need* [6], while the idea of pre-trained language models is similarly deliberated in *ULMFit* [7] and *OpenAI GPT* [8]. The concept of word embeddings further retrospects to *ELMo* [9]

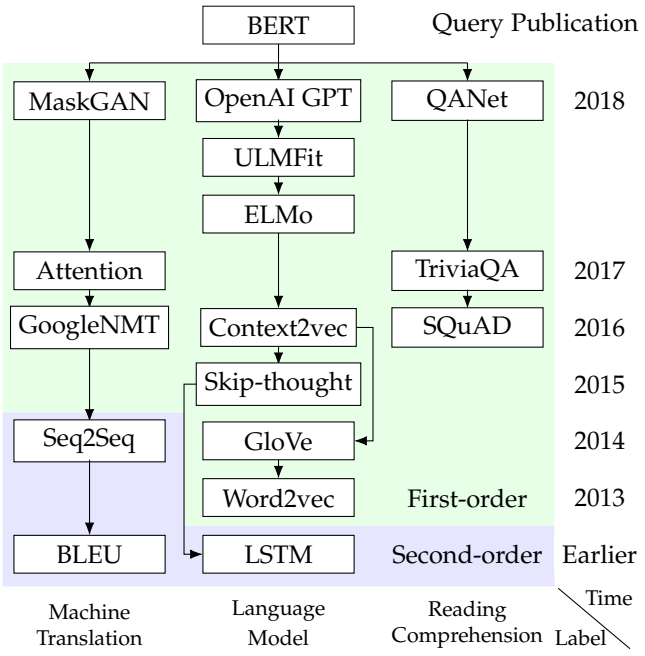


Fig. 1. A simplified evolution roadmap for BERT [5]. Each node represents one publication, and each edge indicates a citation relationship. First-order publications are directly referenced by the query publication while second-order publications are not.

- *Da Yin, Weng Lam Tam and Ming Ding are with the Department of Computer Science and Technology, Tsinghua University, Beijing, China, 100084.*
  E-mail: yd18@mails.tsinghua.edu.cn, tanyl19@mails.tsinghua.edu.cn, dm18@mails.tsinghua.edu.cn
- *Jie Tang is with the Department of Computer Science and Technology, Tsinghua University, and Tsinghua National Laboratory for Information Science and Technology (TNList), Beijing, China, 100084.*
  E-mail:jietang@tsinghua.edu.cn, corresponding author

and *word2vec* [10]. Two critical methods in BERT, *Masked Language Model* and *Next Sentence Prediction*, can be traced back to *MaskGAN* [11] and *Skip-thought* [12] as well. Some of such kind of information could be relatively easy to be found in the reference (first-order), however, for many important clues, the reader has to dig into the second-order references (references of all cited papers), which accounts for thousands of publications.

In this work, to assist readers to gain such deep comprehensions, we generate *evolution roadmaps* to trace the evolution of scientific publications and the development of their undergoing ideas. There are several challenges in discovering the evolution footprints:

Restricted Access to Academic Resource Although open academic repositories such as arXiv[1] have emerged in recent years, access to many libraries like *ACM Digital Library*[2] is still restricted. The limitation on publication contents prevents us from analyzing the full source, while the available metadata, including title and abstract, only provides a glimpse of the main body, which hinders our thorough exploration.

Difficulties for Unsupervised Deep Comprehension The deep comprehension of publications needs extensive readings. Both contents and structure information should be considered during analyzing relationships. Latest embedding methods on contents [5], [9] and graphs [13], [14], either concatenate node features or rely on fine-tuning on downstream tasks, which makes it hard to handle data in unsupervised ways.

Problems for Importance Identification While previous works [15], [16] attempt to quantify the importance of each citation relation, it is necessary but difficult to identify which aspect of the citing publication the reference made contributions to. As a general-purpose model, BERT attracts readers from various fields such as *Question Answering* and *Machine Translation*. Researchers in former field may want to focus on the references relevant to *Reading Comprehension* such as *QANet* [17], while those in latter field might be more interested in the connection between BERT and other translation models like *GoogleNMT* [18]. Distinguishing these prior works in topics can help scholars save time on finding papers related to their concerned areas.

Lack of Ground Truth Unlike the comparison of algorithms in [2], the relations between references and the citing work are not explicitly disclosed. The intentions of citations have been studied in several works [19], [20], where intentions are categorized by coarse functionalities like *Background* and *Method*, or sentiments like *Acceptance* and *Rejection*. But labeling topic information for citations is much more complex, not to mention annotating the quality of evolution roadmaps. Even for experts, opinions could differ a lot. There is no ground truth telling us how each reference is cited or how ideas evolve, which prevents us from modeling with supervision or evaluating directly.

In this paper, we propose a novel framework named **M**aster **R**eading **T**ree (MRT) to tackle the above challenges and generate *evolution roadmaps* for publications. Our contributions mainly include:

- We formalize the problem of *tracing the evolution of scientific publications*, where the *evolution roadmap* is defined to help solve it.
- We develop the MRT framework to generate *evolution roadmap*, which contains two core methods: **Calculate Embeddings** by combining textual and structural information in unsupervised ways and **Construct Roadmaps** by analyzing paper relations based on pre-computed embeddings.
- We design multiple intuitive experiments to evaluate our proposed framework and demonstrate its superiority in capturing underlying relationships between publications over several baseline algorithms.

It is also worth noticing that the formulated problem of tracing evolution roadmaps and the proposed framework can be potentially extended to other domains as well. For example, in social networks, the development process of opinions from public figures can be similarly traced. In addition, the proposed embedding generation method is applicable in many unsupervised scenarios where both textual and structural information are available.

The rest of the paper is organized as below. Related works will be discussed in section 2. The problem definition will be stated in section 3 and the framework will be introduced in detail in section 4. In section 5, we will explain how experiments are set up and evaluated. The last two sections contain a brief introduction to our deployed online system and a short conclusion.

2 Related Works

The target of our work is to find out the evolution of ideas behind scientific publications and some previous works share similar objectives with us. For example, [21] try to detect topic evolution in scientific literature by leveraging citations networks in adapting topic models. [3] use term embeddings and hierarchical clustering to construct topic taxonomy and describe the relations between concepts. [2] proposes to use *algorithm roadmap* to sketch the dynamics of research areas.

Compared to topics or concepts, the understanding of publications is more complicated. Before connecting into an academic network, these publications naturally contain lots of textual information. The deep representation of natural language is therefore essential in our task, which has been studied for decades. Traditional methods like TF-IDF, focus on the statistical attributes based on words or phrases and later topic models such as LDA [22] extract latent features behind documents. While word embeddings like GloVe [23] produce transferable representations, enabling the use of prior knowledge, pre-trained language models including ELMo [9] and BERT [5], incorporate contexts during encoding word sequences, demonstrating remarkable capability on capturing task-related information inside sentences. However, the lack of extremely lengthy inputs in the pre-training corpus has limited those BERT-based models from expressing long text efficiently, which has been addressed in several very recent works [24], [25]. Even so, they still

1. https://arxiv.org
2. https://dl.acm.org

heavily rely on the fine-tuning process in downstream tasks and meet difficulties for unsupervised settings. Sentence-BERT [26] presents a modification of BERT [5] by fine-tuning under siamese and triplet networks, which derives meaningful sentence embeddings but does not handle the long text problem as well.

Although texts of publications can provide rich information, to figure out the evolution of their ideas lying behind, the relationships between publications are indispensable. To incorporate network knowledge, various types of graph embedding techniques have been developed. Unsupervised methods, including skip-gram based, such as LINE [27] and node2vec [13] and matrix factorization based like NetSMF [28] and ProNE [29], largely focus on the neighborhood characteristics. Neural models such as GCN [14] and GraphSage [30], can capture more complex features, but the training of these models demands the supervision information on downstream tasks.

With the expressive representations for publications learned, we can categorize publications in vector spaces and summarize each cluster's main idea by labeling them. Some popular clustering methods, including k-means and spectral clustering, have been proven to be special cases in kernel k-means [31], where semi-supervised constraints can be added as well. The technique of automatic labeling for clusters has also been well studied previously. Works such as [32], use external knowledge sources like Wikipedia or WordNet to select candidates and introduce neural embeddings to improve ranking. Traditional methods like [33] borrow the idea from PageRank and [34] proposes to construct label groups by minimizing the divergence between label distribution and topic distribution.

To analyze the relevance between publications, we reform this problem into a recommendation scenario, where the opinions from users are introduced with the use of reinforcement learning methods. While value-based [35] and policy-based [36] RL algorithms have been successful in games and robotics for years, these methods were reinvented for recommendation problems recently [37], [38], [39]. Compared to traditional neural recommendation models, RL-based methods can optimize the action according to long-term rewards instead of static CTR (i.e., Click Through Rate).

Overall, our work can be broken down into several pieces. Some pieces related to popular topics such as document embedding and network embedding have made progresses in the past few years but are still far from been solved. Other related fields like clustering or automatic labeling were studied for a longer time and now have solid choices available but still need adaptations to make them practical in our circumstances.

3 Problem Definition

The problem of *tracing the evolution of scientific publications*, literally, is to analyze the relation between publications and figure out how later works are influenced by previous ones. Inspired by prior research on topic evolution, we propose to use topics to sketch the relationships. Although it is possible for one reference paper to be cited for multiple reasons, i.e., to inspire the citing paper on several topics, pointing out the ingredient of the citing topics is difficult, as various topics could be overlapping. To this end, we simplified the relation between each reference and the citing paper into one single topic. Then references sharing the same type of relation (same topic) to the citing paper form one *evolution track*, representing one origination of ideas lying behind. Finally, all the *evolution track*s provide a detailed description for the origins of the citing paper, which constitute the *evolution roadmap*.

To help readers understand our work, we firstly introduce several definitions inside the *evolution roadmap* and then formalize the problem of *tracing the evolution of scientific publications* as below.

Preliminary Suppose the citing paper we are interested in, is denoted as q (abbreviated to the query publication), and $\texttt{cite}(p_i, p_j)$ indicates publication p_i cites p_j. $\texttt{Relation}(p_j; p_i)$ refers to the relation from reference p_j to the citing paper p_i, which in our configuration, is one single topic.

Reference The direct references for q may not be enough for mining evolution, so we enlarge the scope of "reference". Formally, we define *first-order references* as the set of papers cited by q., i.e., $R_1 = \{p \mid \texttt{cite}(q, p)\}$ and *second-order references* as additional references cited by R_1, i.e., $R_2^* = \{p \mid \texttt{cite}(p', p), p' \in R_1, p \notin R_1\}$. *Higher-order references* are included as needed. For simplicity, we do not distinguish *second-order references* and *higher-order references* in following sections and we use R_2 to denote the union of them.

Evolution Track As described above, each *evolution track* C_t refers to a group of references sharing the same topic t, formalized by $C_t = \{p \mid p \in R_1 \cup R_2, \texttt{Relation}(p; q) = t\}$. Additionally, each *evolution track* is annotated by N_l labels, with the i-th label for C_t denoted as l_{ti}.

Importance Score While in each *evolution track*, multiple references share the same topic, their importance to the citing paper could be very different. Some might be closely related, like prior state of the art model for comparison, but others probably are just mentioned in one or two sentences for introducing backgrounds. Thus, we assign *importance scores* to each reference and *evolution track* to identify their impact on the citing paper. In detail, the importance is denoted as w_{p_i} for publication p_i and w_{C_t} for evolution track C_t.

Evolution Roadmap To describe the evolution of ideas behind publications, a general roadmap could be defined as a directed acyclic graph (DAG), where nodes represent papers and edges express the topics shared between papers. However, as our main interest is the query publication q and a DAG structure can be complex to interpret, we make compromises on the structure of the roadmap and use a tree data structure for the *evolution roadmap*. Formally, an *evolution roadmap* for one publication q is defined as a tree (V, E, C, W), consisting of

- V: N_p nodes, with q as the root node and other references from $R_1 \cup R_2$ as the rests.
- E: a set of edges. Each edge (p_i, p_j) indicates a potential evolution relation from p_i to p_j.

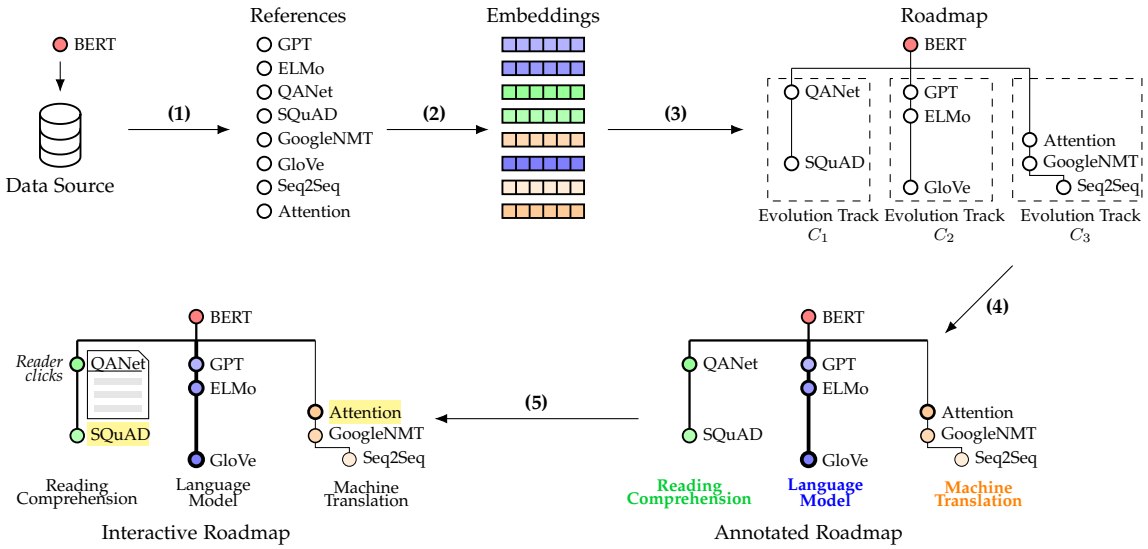


Fig. 2. The procedure of generating evolution roadmap in the MRT framework. **(1)**: We first retrieve reference data from data sources. **(2)**: Then we calculate paper embeddings according to their content and citation relation. **(3)**: We cluster papers into evolution tracks and construct the skeleton of evolution roadmap. **(4)**: We annotate each evolution tracks with labels and assign importance scores to both clusters and papers. **(5)** We recommend related papers for users when some paper is clicked by users.

- C: N_t evolution tracks, with each one annotated by N_l labels.
- W: $N_p - 1$ importance scores for each reference node and N_t scores for evolution tracks.

As mentioned above, each relation $\texttt{Relation}(p_i, p_j)$ can be treated as one single topic t and both p_i and p_j belong to C_t (except the root q belongs to none of those evolution tracks).

Finally, we define the problem of *tracing the evolution of scientific publications* as the following: Given a publication q, construct a directed acyclic graph where each node is one paper and each edge indicates an evolution relation. In this work, we propose to generate *evolution roadmaps* to solve this problem, with several compromises made considering the complexity of interpretation.

4 Framework

In this work, we propose a framework named **M**aster **R**eading **T**ree (MRT) to generate the *evolution roadmap*, which mainly contains two parts:

(1) **Calculating Embeddings** To gain a deep comprehension for all publications and analyzing their relations, we generate expressive representations encoding both textual and structural information. A combination of document embedding and graph embedding is proposed in unsupervised styles.

(2) **Constructing Roadmaps** After projecting publications into latent vector spaces, we apply clustering and automatic labeling techniques to build *evolution roadmap* based on pre-computed representations.

As illustrated in Figure 2, the **Calculating Embeddings** part refers to the second step of the whole procedure and the following three steps form the **Constructing Roadmaps** part.

First of all, we need to retrieve necessary paper data for the query publication q. As mentioned above, the restricted access to the body of academic papers prevents us from using all paper contents freely. Besides, unlike previous works [2] which only use papers from limited sources such as the ACL Anthology or NeurIPS, our proposed *evolution roadmap* involves using all references of the target paper. To tackle the above problems, we propose to build evolution roadmaps only based on the metadata of papers, which can be accessed freely from open academic data platforms (such as Semantic Scholar and AMiner). Although the text data in metadata (title and abstract) is relatively short, it usually concisely deliberates the research background, proposed methods, experiment results and concludes the contribution or potential influence to the related research area, which is informative enough for analyzing evolution roadmaps.

Additionally, to compensate the lack of details due to the use of metadata, we include *second-order* references to provide extra information beyond the *first-order* ones. The size of *first-order* references for one query publication q could vary from several to hundreds, depending on the publishing requirements and q's own contents, while the number of *second-order* references is much larger. We select the most relevant N_p publications as the inputs for the following algorithms. The selection procedure can be implemented in several different ways, such as using the number of citations, the node degrees in the sampled subgraph, or PageRank scores. We observe that the difference between these choices is minor, as the most valuable works are always kept. It is hard to evaluate their performances directly, but in the experiment section, we will see PageRank reflects the users' interests better so in this work we adopt PageRank.

4.1 Calculating Embeddings

To find out the underlying evolution tracks and estimate their importance, we need to generate high-quality expressive representations for publications.

The representations should encode both publication content and citation network structures. However, there are several problems with existing embedding techniques.

First, BERT-based [5] text embedding models usually need to be fine-tuned on downstream tasks to produce effective embeddings. Although Sentence-BERT [26] tackled this problem by fine-tuning on text similarity tasks, the lack of long texts in training instances still leaves it hard to encode long texts. Second, graph embedding methods such as GCN [14] also rely on labeled data while unsupervised methods like node2vec [13] mainly produce structural embeddings independent of node features.

Besides, we also found that the concatenation of embeddings without fine-tuning is not easy to use with cosine-similarity or other distance metrics. Let x_i^α and x_i^β be the two different types of embeddings for element i, the concatenated embedding is $[x_i^\alpha; x_i^\beta]$. The euclidean distance between element i and j can be formulated as

$$\sqrt{||x_i^\alpha - x_j^\alpha||^2 + ||x_i^\beta - x_j^\beta||^2} \tag{1}$$

and the cosine-similarity is

$$\frac{x_i^\alpha x_j^\alpha + x_i^\beta x_j^\beta}{\sqrt{||x_i^\alpha||^2 + ||x_i^\beta||^2}\sqrt{||x_j^\alpha||^2 + ||x_j^\beta||^2}} \tag{2}$$

If both embeddings are normalized, the distance or similarity can be seen as a direct addition or average for the results in two vector spaces. If not, there will be an imbalance problem with two types of embeddings, where the vector with longer length will take the major part. While most embedding methods do not ensure the output to be normalized, it is also uncertain whether two embeddings are equally important, since we do not have a fine-tuning process to learn their weights.

To tackle the problems mentioned above, we first propose to combine TF-IDF and Sentence-BERT [26] to encode publication contents and then adapt the spectral propagation process proposed in ProNE [29] to incorporate structure information as shown in Figure 3.

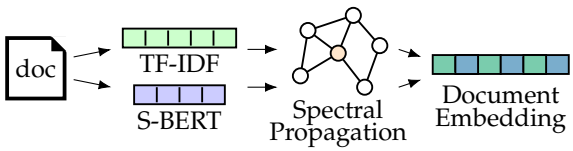


Fig. 3. The procedures of calculating embeddings.

Formally, we first build the TF-IDF and Sentence-BERT embeddings for each publication p_i as follows:

$$\tilde{x}_i^t = \texttt{TF-IDF}(p_i),\ \tilde{x}_i^s = \texttt{S-BERT}(p_i) \tag{3}$$

$\tilde{x}_i^t$ and $\tilde{x}_i^s$ are embeddings encoded by TF-IDF and S-BERT respectively. For TF-IDF, we only use words composed of English alphabets and digits followed by lemmatization. We also take n_g-gram phrases into account. We pick the top frequent n_w words or phrases and use their TF-IDF values for each publication p_i as the n_w dimension TF-IDF embedding $\tilde{x}_i^t$. For Sentence-BERT, we treat all texts of publication p_i as one single sequence and use the output sequence embedding as the S-BERT embedding $\tilde{x}_i^s$.

Then these two content embeddings are propagated on the citation graph G, where the nodes are the papers and edges are the citations.

$$\hat{x}_i^t = \texttt{Propagate}(\tilde{x}_i^t, G),\ \hat{x}_i^s = \texttt{Propagate}(\tilde{x}_i^s, G) \tag{4}$$

Finally, they are concatenated and propagated again to construct the final representation.

$$x_i = \texttt{Propagate}([\hat{x}_i^t; \hat{x}_i^s], G) \tag{5}$$

The TF-IDF algorithm encodes long texts in literal ways and the BERT-based encoder is expected to capture more latent information. Despite the difficulties for TF-IDF to find relations between documents sharing the same topic but using different words, this problem is most frequent for short sentences. But in our task, titles and abstracts for most publications provide rich textual information. Furthermore, most publications, centering around the query paper q, are mutually connected through only one or two citation hops, and their descriptions for the same referent usually share lots of expressions. As a result, TF-IDF works pretty well in finding relations between publications.

The BERT-based method has demonstrated its superior capability of capturing latent semantic information behind texts. While extremely long texts could prevent us from applying S-BERT directly, fortunately, around 98% of the text inputs for publications (the concatenation of title and abstract) have less than 512 word tokens, as shown in Figure 4. So it is relatively safe to discard overflowing parts. However, S-BERT is fine-tuned mainly on some short sentences-based corpus and such training data for long texts is hard to access or label. Therefore, S-BERT has limited performance in our task but still provides extra boosting on top of TF-IDF, which will be shown in the following experiments.

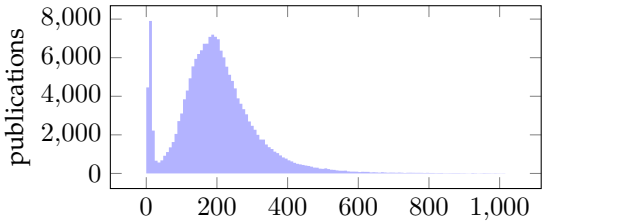


Fig. 4. The histogram of text length for publications used in this work after tokenization, where over 200,000 publications from various communities have been considered. For example, there are around 4,500 publications only have less than 8 tokens in titles and their abstracts are absent in the data source.

For encoding the structure of G, we take spectral propagation proposed in ProNE [29] to incorporate both neighborhood and global features. For each "Propagate" iteration, the node embedding x_i for publication p_i is updated by

$$x_i \leftarrow D^{-1}A(I_{N_p} - \tilde{L})x_i \tag{6}$$

where $D^{-1}A$ is the normalized adjacency matrix for graph G, I is the identity matrix and $\tilde{L}$ is the modulated Laplacian. This method is proposed in ProNE to enhance the node embedding generated by pre-factorization. In graph theory, the random walk normalized Laplacian matrix is

defined as $L = I - D^{-1}A$. It can be decomposed by $L = U\Lambda U^{-1}$, where U and Λ are constituted with eigenvectors and eigenvalues. The propagation Lx can be seen as transforming x into spectral space, scaling by eigenvalues and then transforming it back. The modulated $\hat{L}$ is then defined as $Ug(\Lambda)U^{-1}$ where g is the spectral modulator which strengthens the top smallest and largest eigenvalues, corresponding to local and global features respectively.

Therefore, each propagation step can be interpreted as aggregating neighborhood node features by random walk, considering global information at the same time.

Additionally, in ProNE, SVD is applied to the output embeddings in order to maintain the orthogonality. We found it also helpful while concatenating embeddings with various lengths, as it reduces the dimension of embeddings and keeps the most informative features.

As for the implementation details, we use the same spectral modulator and Chebyshev expansion as used in ProNE to accelerate the calculation by avoiding explicit eigendecomposition.

4.2 Constructing Roadmaps

With the embeddings generated in the previous step, we can construct evolution tracks by clustering papers and use automatic labeling method to assign labels for them. The importance scores for references and evolution tracks are also calculated in intuitive ways.

Clustering After projecting papers into latent vector space, we can apply k-means or spectral clustering to group those publications into different evolution tracks. Kernel k-means, which has been proven to be a generalized solution [31] for both k-means and spectral clustering, is more flexible where external constraints are available as well. The euclidean distance between x_i and the centroid m_{C_t} for cluster C_t can be represented as

$$||x_i - m_{C_t}||^2 = K_{ii} - \frac{2\sum_{p_j \in C_t} K_{ij}}{|C_t|} + \frac{\sum_{p_j, p'_j \in C_t} K_{jj'}}{|C_t|^2} \quad (7)$$

where K is the affinity kernel and $|C_t|$ is the size of cluster C_t. Under the current context, the affinity kernel is calculated by

$$K_{ij} = x_i^T x_j + \alpha A_{ij} + \beta \Phi_{ij} \quad (8)$$

where x_i is the representation for publication p_i which we got from the previous step. A is the symmetric adjacency matrix for citation graph G as described above. The first term derives from the original k-means algorithm and the second adjacency term originates from the intention that we want to break as few citation relations as possible from the citation graph G during clustering. The second term corresponds to the objective of ratio association in graph clustering, which is proved to equal to kernel k-means with $\sigma I + A$ as kernel in [31]. We notice that using normalized adjacency matrix in Equation 8 also works but towards a slightly different objective of minimizing graph normalized cut.

The third term Φ is the additional supervision where Φ_{ij} denotes p_i and p_j have close relationship potentially. For example, although we do not necessarily need the full text of the source publication q to generate its evolution roadmap, if the full text is provided, we can extract the mentioned positions of reference papers in q. If two references p_i and p_j are mentioned together (such as in one sentence) in q, p_i and p_j are believed to have *close relationship* and we assign $\Phi_{ij} = 1$. We will show that this type of extra information can improve the quality of generated roadmap in the experiment section. In addition, we can also introduce user feedbacks or expert opinions to add contraints on the generated roadmap. If p_i and p_j are always preferred to be categorized in the same cluster, we can alter the clustering process by increasing the value of Φ_{ij}.

After publications are categorized into different evolution tracks, they need to be connected into the *evolution roadmap*. Similar to *algorithm roadmap* proposed in [2] where temporal order is used to connect *algorithm*s, we sort papers in each evolution track by their publishing years and citation order. The first-order references are connected into the *primary timeline* and second-order ones form the *secondary timeline*. The latest publication p' in the *secondary timeline* is connected to the earliest publication in the *primary timeline* that is published after p'. Finally, the latest publications in the *primary timelines* for all evolution tracks are joined with q to form the skeleton of the *evolution roadmap*. The clustering method is depicted in Figure 5.

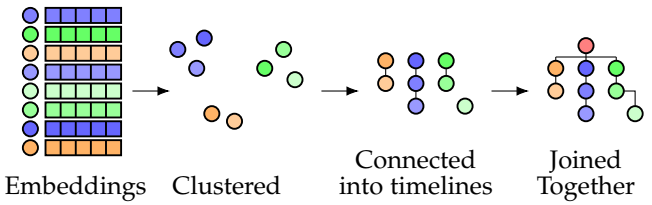


Fig. 5. To generate the skeleton of *evolution roadmap*, we first cluster papers into evolution tracks based on the pre-computed embeddings and then connect them into timelines by their publishing year and citation order.

Labeling For each evolution track generated in the previous step, we create a group of labels to identify it. The process of automatic labeling includes two procedures: candidate label extraction and ranking.

While extracting label candidates, using external knowledge bases such as Wikipedia may have difficulties in covering the latest academic terminologies. Besides, scientific publication titles and abstracts are usually concise and informative. Therefore, we extract bi-gram and tri-gram phrases directly from the raw text. The threshold of frequency is set to be the half or one third to the most frequent one depending on the length of the candidate phrase. Phrases with a frequency lower than the threshold are discarded and the remaining ones are kept as candidates.

Adapting based on [34], we rank our candidate labels following three criteria:

1) Good evolution track label should have high coverage for the contents of papers inside it.
2) Good evolution track label should be discriminative to labels assigned to other evolution tracks.
3) Good evolution track label should be relevant to the content in the query publication q since those evolution tracks are extended from it.

The content for each cluster C_t can be seen as a distribution of words and the co-occurrences of words with label l can be also interpreted as the contents covered by label l. The KL divergence between them reflects to what extent the label l can represent cluster C_t, and can be written as

$$\begin{aligned} &KL(C_t||l) \\ &= \sum_{word_i} p(word_i|C_t) \log \frac{p(word_i|C_t)}{p(word_i|l)} \\ &= -\sum_{word_i} p(word_i|C_t) \log \frac{p(word_i, l|\cdot)}{p(word_i|\cdot)p(l|\cdot)} \\ &\quad + KL(C_t||\cdot) + \sum_{word_i} p(word_i|C_t) \log \frac{p(word_i|l,\cdot)}{p(word_i|l)} \\ &= -\mathbb{E}_{C_t}[PMI(word, l|\cdot)] + KL(C_t||\cdot) - Bias(l,\cdot) \end{aligned} \tag{9}$$

where $(\cdot)$ is the global corpus consisting of q and its references. PMI is the pointwise mutual information. The second cluster constant term $KL(C_t||\cdot)$ is identical while choosing labels. The third bias term $Bias(l,\cdot)$ is mainly introduced by using $(\cdot)$ as the context. While C_t is selected from $(\cdot)$ in our task, this term can also be ignored, which is same with [34]. Then the ranking score can be defined as

$$\begin{aligned} &Score(l, C_t) = (1 + \frac{\mu}{N_t - 1})\mathbb{E}_{C_t}[PMI(word, l|\cdot)] \\ &- \frac{\mu}{N_t - 1}\sum_{j=1}^{N_t} \mathbb{E}_{C_j}[PMI(word, l|\cdot)] + \phi\mathbb{E}_q[PMI(word, l|\cdot)] \end{aligned} \tag{10}$$

where μ and ϕ controls the discriminative power and the query publication coverage, respectively.

While some evolution tracks might contain multiple subtopics, we create a group of N_l labels to identify each track. The selection of N_l labels in one label group also considers Maximal Marginal Relevance criterion [40], which shares a similar formula as described above. The general idea is to make the coverage of label groups as large as possible.

Importance Score References and evolution tracks are further annotated with importance scores to help readers identify their relative influence or contribution to the development of the query publication.

Suppose the index for q is i_q, the importance score w_{p_i} for p_i is directly assigned as the kernel weight K, defined in Equation 8, relative to the query publication q and the score for evolution track C_t is defined as the sum of all reference scores inside it.

$$w_{p_i} = K_{i_q i},\ w_{C_t} = \sum_{p_i \in C_t} w_{p_i} \tag{11}$$

This design is mainly based on the similarity between generated paper embeddings. We will show in the experiment section that this method performs better than simply using the citation numbers or PageRank scores.

Recommending Although importance scores can be used to identify the most influential papers for the query publication q, users may have different interests and their focus points might also change during navigating the evolution roadmap. To better help readers focus on the most relevant publications, we develop an agent for the roadmap to recommend potentially appealing works. The whole scenario is illustrated in Figure 6.

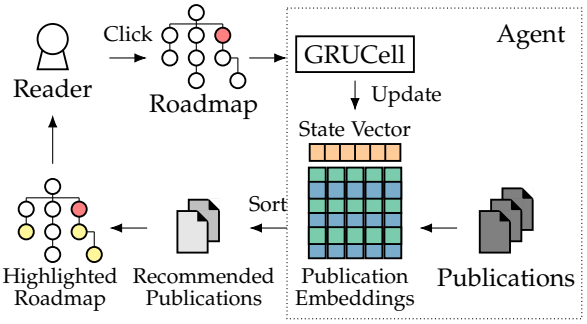


Fig. 6. The recommending procedure: when a reader is viewing the evolution roadmap, he might click or hover on some reference paper to see more details of it. This action can be viewed as a potential interest for that paper and our agent will update the state vector, which encodes the user's history. According to the updated state vector, the agent will select relevant papers which the user might want to read next and recommend those papers by highlighting them on the evolution roadmap.

Our agent's job is to recommend k papers after receiving a user action. For new roadmaps, since we do not have user responses, to tackle the cold start problem, we develop agents following the default strategy: recommend publications with the closest embeddings to the current clicked one.

After user feedback collected for popular roadmaps, we train agents with the REINFORCE algorithm [38], where the reading histories are encoded into state vectors and publications in roadmap G are treated as action candidates. Formally, the Markov Decision Process (MDP) is set up with

1) $\mathcal{S}$: a continuous state space where state vectors encode previous reading history.
2) $\mathcal{A}$: a discrete action space containing all the publications in the roadmap.
3) $\mathcal{P}$: the state transition probability.
4) $\mathcal{R}$: the reward function where r_t is the immediate reward for time t. If the next user click falls into the recommended action sets, the immediate reward will be 1, otherwise 0.
5) γ: the discount factor for future rewards. Set to be 0.9.

The goal for the RL algorithm is to find a recommendation policy $\pi : \mathcal{S} \to \mathcal{A}$ to maximize the expected cumulative reward $\mathbb{E}_{\tau\sim\pi}[\mathcal{R}(\tau)]$ where τ is the recommending trajectory.

For simplicity, $\mathcal{S}$ shares the same dimension d with the publication embeddings x. The initial state s_0 is set to be x_{i_q}, the embedding of the query publication q. We model $\mathcal{P}$ with a single-layer GRU network [41] as follows:

$$s_{t+1} = \texttt{GRU}(s_t, c_t) \tag{12}$$

where c_t is the user's click at time t, and the policy is calculated with

$$\pi_\theta(p_i|s_t) = \texttt{Softmax}(s_t^T \mathbf{W} x_i) \tag{13}$$

where $\mathbf{W} \in \mathbb{R}^{d\times d}$ is a trainable parameter matrix. At time t, publications are sampled from $\pi_\theta(p_i|s_t)$ during training while publications with top-k highest probabilities are recommended during inference. Following [38], the gradient

of expected rewards for top-k recommendation is approximated by

$$\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta}[\mathcal{R}(\tau)] \approx \sum_{\tau\sim\pi_\theta}\sum_{t=0}^{|\tau|}\hat{r}_t \nabla_\theta \log \alpha_\theta(p_i|s_t) \tag{14}$$

where $\hat{r}_t = \sum_{t'=t}^{|\tau|}\gamma^{t'-t}r_t$ denotes the expected future rewards at time t and $\alpha_\theta(p_i|s_t) = 1-(1-\pi_\theta(p_i|s_t))^\kappa$ denotes the probability of taking p_i in the first κ samplings. κ ($\geq k$) is the total number of samplings to sample k unique publications from $\pi_\theta(p_i|s_t)$ with replacement.

In brief summarization, our proposed framework constructs evolution roadmaps based on the publication embeddings. The evolution relations are implicitly encoded in the paper representations and then explicitly expressed by clustering and annotating. While the unsupervised framework might be a bit ad-hoc, we also introduce some user feedback to make the whole algorithm more practical. Besides, despite the lacking of ground truth data to evident our methods, we designed several intuitive experiments to show our choices are reasonable and explainable.

5 Experiments

In this section, we will introduce several experiments designed for evaluating different parts of our proposed framework. The problem we want to solve is relatively novel and complex. Without established benchmarks or available existing datasets, we attempt to measure the performance through several indirect criteria. It is hard to say any of these empirical results is convincing solely, but we believe these outcomes together evident that our framework is a practical solution for the raised problem.

All the experiments in this paper use the configurations described in Table 1.

TABLE 1
Main configurations for experiments

Symbol	Description	Value
N_p	Number of publications for each roadmap	100
N_t	Number of evolution tracks	6
N_l	Number of labels for each evolution track	5
k	Number of recommended publications	5
n_g	The maximum number of words in one phrase used in TF-IDF	5
n_w	Number of top frequent words used in TF-IDF	2000

5.1 Dataset

The quality of the generated roadmap is closely associated with the completeness of the reference lists. For example, BERT does not only utilize the works from ACL or EMNLP. Hence, instead of using one single conference as our data source, we use open academic platforms including AMiner [42] (for the online system) and Semantic Scholar [43] (for the online system and experiments), where metadata of publications from various sources like DBLP are integrated. This is also a notable difference compared with many prior works such as [2], where only papers from the same conference or dataset are considered.

For evaluations, we use publications from KDD and ACL conferences in 2018 and 2019 as query publications and generate roadmaps for them. Their references are retrieved from Semantic Scholar[3]. The statistics of each dataset are shown in Table 2.

TABLE 2
Dataset statistics for evaluations.

Dataset	Papers[1]	Retrieved References[2]	Citation Links[3]
KDD	534	126,499	1,663,063
ACL	679	88,876	3,202,684

[1] *Papers* refer to the publications used as the query publication q. This is also the number of *evolution roadmap*s we tested.
[2] *Retrieved References* refer to the first-order and second-order references we retrieved from Semantic Scholar, which are not necessarily inside the same conference with the query publications.
[3] *Citation Links* indicate how many links are considered between publications. This is the number of links we used while using PageRank to select related papers.

5.2 Evaluations and Results

There are mainly two parts contained in the framework. The first part is to generate publication embeddings and we designed *Neighborhood Similarity* experiment to test the embedding quality. In the second part, for each of four minor tasks, we also conducted another four experiments to test the effectiveness of our proposed methods.

Neighborhood Similarity We first evaluate the document representations generated in the first part. There are two difficulties in measuring the quality of these embeddings. First, our downstream tasks are also unsupervised, making it hard to rely on end-to-end tests. Second, unlike words or sentences, it is even tough for human experts to judge the similarity between publications.

Therefore, we proposed to use *Neighborhood Similarity* to evaluate based on the assumption that similar publications should have more overlaps between their *neighborhoods*. The *neighborhood* $\mathcal{N}(p_i)$ for publication p_i is defined as

$$\mathcal{N}(p_i) = \{p \mid \texttt{cite}(p, p_i) \vee \texttt{cite}(p_i, p)\} \cup \{p_i\} \tag{15}$$

and the *Neighborhood Similarity* $sim_\mathcal{N}$ between publication p_i and p_j is defined as

$$sim_\mathcal{N}(p_i, p_j) = \frac{|\mathcal{N}(p_i) \cap \mathcal{N}(p_j)|}{\sqrt{|\mathcal{N}(p_i)| \cdot |\mathcal{N}(p_j)|}} \tag{16}$$

which is also known as Jaccard index. Notice that this equation is used to calculate the ground truth similarity between publication p_i and p_j. The general purpose of this test is to assess if the cosine similarity $cos(x_i, x_j)$ between generated publication representations is in accord with the neighborhood similarity described above.

Like other similarity tests [44], we compared the Spearman rank correlation [45] between embedding cosine similarity and neighborhood similarity over several methods. The baseline methods include using TF-IDF, S-BERT, ProNE, and node2vec individually, and the proposed methods include their combinations. The results are shown in Table 3.

As the ground truth is closely related to explicit connections between publications, structure-based algorithms, including ProNE and node2vec, outperform content-based ones by a large margin. Simply concatenating different embeddings (TF-IDF+S-BERT) turns out to be a bad idea

3. https://api.semanticscholar.org/

TABLE 3
Neighborhood Similarity Experiment

Method	KDD	ACL
TF-IDF[1]	0.50	0.49
S-BERT[2]	0.41	0.36
ProNE[3]	0.72	0.75
node2vec[4]	0.65	0.64
TF-IDF+S-BERT	0.41	0.36
TF-IDF+ProNE	0.78	0.79
S-BERT+ProNE	0.75	0.77
TF-IDF+S-BERT+ProNE	**0.81**	**0.82**

[1] For TF-IDF, we select top frequent 2000 features and use n-grams ranging from 1 to 5, as shown in Table 1.
[2] For S-BERT, we use the pre-trained model of bert-base-nli-stsb-mean-tokens.
[3] For ProNE, the embedding dimension is 32 and the order of Chebyshev expansion is 10, according to [29].
[4] For node2vec, the embedding dimension is 32. Walk length and number of walks are set to be 20 and 60, respectively. The window size is 5.

that might be caused by the reasons we mentioned above. The propagation on content embeddings yields significant improvement compared to the original ProNE algorithm. As for document content representation, TF-IDF is yet one of the most powerful competitors. The Sentence-BERT encoder is able to provide extra enhancement beyond TF-IDF, probably due to its capability of encoding deeper information, but itself still meets difficulties while dealing with long texts. We also test the performance of using BERT to replace S-BERT. The performance is far behind the S-BERT (only 0.11 on KDD dataset), which corresponds to our analysis described in Section 4.1.

It is worth to be mentioned that this metric is similarly proposed as *second-order proximity* in LINE [27] but there is a slight difference. While we were generating embeddings for papers, the network structure we were using is a subgraph of citation network on N_p selected papers. The *Neighborhood Similarity*, on the other hand, is defined by the complete neighborhood on the whole citation network, where many more papers outside N_p selected papers are also taken into account. This means in this experiment, we generate embeddings based on a subgraph structure and expect these embeddings to reflect some global attributes.

Besides, there are two reasons why we do not directly use this similarity to generate evolution roadmaps. First, our generated embeddings only use information from a small subgraph while the neighborhood similarity uses global information which requires more data to fetch. For example, some famous works might have thousands of citations. Since our subgraph is explored through reference links, we do not need to fetch those citations. Second, the citation information is always changing. New citation links may come out as papers published. The subgraph created by reference links will be determined when the query publication is decided so our proposed method for generating publication embeddings is time-invariant.

Co-mention and MST Trials To evaluate the structures of generated roadmaps, we designed two trials to test the quality of clustering and linking.

The mentions of first-order references inside the query publication provide strong implications for some publication pairs. For example, *"They either rely on pattern-based methods [14, 32] which extract hierarchical relation leveraging linguistic features, or clustering-based methods [11, 42], which cluster concepts to induce an implicit hierarchy."* is a piece of text from [2]. Co-mentions inside the same boxes, marked as *strong co-mentions*, such as 14 and 32, indicate their similar functionalities. Co-mentions in the same paragraph, marked as *weak co-mentions*, such as 14 and 11, also implies potential connections, especially under the context of the citing publication.

Similar to [2], we use *pdftotext* provided from *xpdf*[4] to extract the *strong co-mention* and *weak co-mention* pairs described above from the full text of query publications. The *co-mention hit rate* is defined as the percentage of co-mention pairs sharing the same cluster assignment.

The *co-mention hit rate* has preferences over allocations with uneven cluster sizes since huge clusters are particularly competitive. To avoid being blinded by this bias, we introduced another trial focusing more on the roadmap structure. As clusters are reorganized into a tree-structure graph, it is helpful to compare the result with an unconstrained solution. To achieve that, we set up a fully connected graph $\hat{G}$ with nodes representing all publications. Every two nodes for publication p_i, p_j are connected by an edge with weight $sim_N(p_i, p_j)$ defined in equation 16. Kruskal's algorithm [46] is applied on $\hat{G}$ and the total weight for edges from the maximum spanning tree is recorded as $BestScore$, which is compared with that for the generated roadmap, denoted as $RoadmapScore$. The relative MST score is defined as $\frac{RoadmapScore}{BestScore}$ where higher scores usually mean better structures. The average of all samples inside the dataset is reported in Table 4.

TABLE 4
Co-mention and MST Trials

Method	Co-mention*		MST	
	KDD	ACL	KDD	ACL
w/o supervision				
Hierarchical	0.63, 0.48	0.66, 0.51	0.55	0.57
Spectral	0.62, 0.48	0.65, 0.51	0.55	0.57
K-means**	**0.73**, **0.57**	0.77, 0.60	0.57	0.59
Kernel k-means	**0.73**, 0.56	**0.78**, **0.61**	0.57	0.59
w/ supervision				
Strong Co-mention	0.81, 0.58	0.85, 0.64	0.57	0.59
Weak Co-mention	0.84, 0.73	0.88, 0.77	0.57	0.59

* The co-mention columns include strong co-mention hit rate (left) and weak co-mention hit rate (right).
** K-means is also a special case for kernel k-means, setting $\alpha = \beta = 0$.

Compared to hierarchical or spectral clustering, k-means works best with high-quality embeddings. The kernel k-means ($\alpha = 1.0$, $\beta = 1.0$), extended from k-means, benefits from incorporating supervision information. For example, the use of strong co-mentions can improve weak co-mention scores and vice versa. We noticed that the additional adjacency matrix introduced in kernel k-means works well for some samples but the overall improvement is limited compared to the original k-means, since the document embeddings have already been propagated.

Inverse Label Distance and Overlap Rate The quality of annotated labels is similarly evaluated using the full source of the query publication as above. If the mentioned location

4. https://www.xpdfreader.com/

for publication p_i from cluster C_t is close to the occurrence of the j-th label l_{tj} of C_t, then we say label l_{tj} is high-quality.

Formally, for each first-order reference p_i and each label l_{tj} in C_t, we extracted their paragraph positions in the source of q. The closest paragraph distance between the occurrences of p_i and l_{tj} is denoted as dis_{ij}. For example, if p_i and l_{tj} occur in the same paragraph in the source of q, then $dis_{ij} = 1$. If l_{tj} does not appear in the source of q, then $dis_{ij} = \infty$. The *Inverse Label Distance* for roadmap G is defined as

$$\texttt{ILD}(G) = \frac{1}{N_t} \sum_{t=0}^{N_t-1} \max_j \frac{1}{|C_t|} \sum_{p_i \in C_t} \frac{1}{dis_{ij}} \tag{17}$$

Apart from the quality of each cluster label, we also need to consider them collectively. A high overlap rate between clusters (evolution tracks) should be avoided as well. We thus use the overlap rate to measure the labeling algorithm performance further. The overlap rate is defined as the proportion of duplicated labels among all selected labels across clusters, i.e.,

$$\texttt{Overlap}(G) = 1 - \frac{|\{l_{tj} \mid \forall t, j\}|}{N_t N_l} \tag{18}$$

We developed two trivial baseline algorithms for comparison. The first one is to use the top frequent bi-gram and tri-gram phrases from publication titles and abstracts as cluster labels directly, denoted as *Frequency*. The second one is to use TF-IDF to select labels where papers in one cluster are treated as a long document, denoted as *TF-IDF*. Results are reported in Table 5.

TABLE 5
Inverse Label Distance and Overlap Rate for labeling

Method	ILD		Overlap	
	KDD	ACL	KDD	ACL
Baseline Methods				
Frequency	0.68	0.69	0.14	0.21
TF-IDF	0.66	0.64	**0.07**	**0.09**
Proposed Methods				
$\mu = 0.8, \phi = 0.1$	0.75	0.71	0.13	0.16
$\mu = 0.0, \phi = 0.1$	0.78	0.73	0.40	0.43
$\mu = 0.8, \phi = 0.0$	0.73	0.69	0.11	0.14
$\mu = 0.8, \phi = 0.5$	**0.79**	**0.76**	0.24	0.27

The results show that our proposed method ($\mu = 0.8, \phi = 0.1$), extended from [34], outperforms the *Frequency* baseline algorithms for both inverse label distance and overlap rate. The *TF-IDF* method, though reduce the co-occurrence of labels across clusters, harms some high-quality labels due to their occasional mentions in other cluster papers and has significantly lower ILD scores. The ablation tests show that the discriminative term, controlled by μ, can efficiently reduce the overlap rate when generating labels across clusters. The additional query publication coverage term, controlled by ϕ on the other hand, can improve the coherence between labels and the query publication at the cost of higher overlapping probability.

Importance Evaluation with User Click The calculation of importance for publications is difficult to judge as there is no ground truth data available. In addition, even for experts, labeling importance for publications is not easy and the decision might be rather subjective as well. For this reason, we choose to use online feedback from users to evaluate the importance score.

Borrowing the idea from item recommendation and search engine, we use user clicks for ranking publications. Although various users may hold different opinions for which publication is more important, the overall click through rate for publications generally reflects the interests of the majority. The importance scores should be coherent to the common interests so we measure the Spearman correlation between user clicks and importance scores. In addition, we use Normalized Discounted Cumulative Gain (NDCG) to check if the selection for top-K items is reasonable since most users will focus more on the top important publications.

As for the experiment settings, we collected the 10 most popular MRTs on AMiner with around 10 thousand user clicks in total. Each one received clicks from at least 50 unique users on the internal publications. The citation numbers, node degrees and PageRank scores for publications are treated as baseline methods. The averaged Spearman correlation and NDCG results are reported in Table 6.

TABLE 6
Importance Evaluation with User Click

Method	Spearman	NDCG@5	NDCG@20
Citation Number	-0.23	0.19	0.28
Out-degrees	-0.15	0.21	0.36
In-degrees	0.36	0.56	0.65
PageRank	0.38	0.61	0.70
Importance Score	**0.41**	**0.87**	**0.79**

The out-degrees, in-degrees and PageRank scores are all calculated based on the subgraph of citation network. The subgraph has N_p papers as nodes and all their internal citation links.

From the results, we can see that our proposed importance score provides better ranking compared to baseline algorithms, concerning users' interests. The citation number mainly suffers from two critical problems under our circumstances. First, both citation numbers and PageRank scores favor old well-known pioneer works more due to their high impacts but overlook the contribution of recent emerging works. Second, with the existence of the query publication, the users' interests for reference publications will be biased, which is neglected while using the citation number.

The PageRank algorithm tackles the above problems in indirect ways. By running on the subgraph centered on the query publication, the PageRank scores will implicitly consider the bias of the query publication. In Table 6, the PageRank scores have much higher coherence to users' interest while the citation number itself has poor performance on that. This observation also explains why we use PageRank scores to select N_p papers before calculating embeddings, instead of using citation numbers. In addition, the *in-degrees* work similar to PageRank and their performances are close. So it might be also practical to use it when selecting papers, as it requires less time to compute compared to PageRank.

In terms of our proposed importance score, it directly models the similarity between reference publications and the query publication, which receives even better results than PageRank scores. We speculate that directly modeling

the environment condition (the query publication) helps capture the interests of users a lot.

Recommendation Evaluation For dynamic recommendation, we evaluated the performance of the REINFORCE algorithm based on the average rewards from user click trajectories. This experiment is conducted based on the user click data collected from the AMiner online platform, and we focused on several top popular roadmaps. Over 10 thousand raw click events are collected and pre-processed into around 1,600 trajectories. In detail, all click data are first grouped by IP addresses, with each sequence sorted by timestamp. Those click sequences are further divided into different sessions where time intervals for adjacent clicks are shorter than 10 minutes. Continuous duplicated clicks in sessions are merged as well. As for the experiment, 80% of the data are used as training data and the rest 20% are utilized as validation data. The baseline algorithm is our default strategy mentioned above: recommending using publications with the closest embeddings to the clicked one.

TABLE 7
Average Rewards for Dynamic Recommendation

Roadmap	Models	
	Baseline	REINFORCE
BERT	0.32	**0.66**
GAN	0.28	**0.40**
ResNet	0.67	**0.78**
GraphSage	0.75	**0.83**

Results shown in Table 7 demonstrate that the REINFORCE algorithm performs better than the static baseline algorithm. However, due to the lack of data, most models are far from being fully fit. For the most popular roadmap, the BERT one, we collected over 3,500 user clicks and extracted around 600 user click trajectories, with an average path length of 3.24. We observe that about 20% of users will jump back to previously visited references after one or two hops. The relation between the current click and the last click is also affected by behavior patterns. If the focus of one user moved between publications within the same cluster or the same year previously, the probability for the later occurrence of the same pattern would rise by 10% to 20%. Besides, some users seem to prefer moving along citation links instead of timelines.

Human Evaluation Finally, we conducted a simple human evaluation to assess the quality of generated roadmap in an end-to-end manner. We invited three domain experts and show each person 40 generated roadmaps in several fields such as NLP, CV and graph learning. Half of the roadmaps are generated only using TF-IDF and k-means, which are treated as the control group. The rest are generated with the proposed framework. For each roadmap, experts give rates (1 to 5) to the overall quality of the whole roadmap. The control group received an average score of 3.68 while the proposed method received 3.82, which indicates that our proposed method is comparatively better than the baseline.

However, there are some problems with the evaluation. For each rating, we asked experts to give a confidence score, to identify their familiarity with the query paper and related fields. We found that for those roadmap ratings with low confidence, the TF-IDF-based control group is even slightly better. It is because when experts explore unfamiliar fields, they usually check if papers in the same cluster share the same keywords, which is exactly how TF-IDF works but often neglects the citation relations. Secondly, when evaluating clustering performance, instead of directly inspecting the relationships of papers, the experts tend to compare papers with cluster labels. This reflects that the roadmap quality in human evaluation heavily relies on the tagging performance and sometimes overlooked the paper clustering performance. We concluded from the above findings that it is hard to make a comprehensive evaluation for different components in the proposed framework fairly through human evaluations and knowledgeable experts are also necessary to reach reasonable conclusions.

TABLE 8
Average Running Time for Each Algorithm

Algorithm	Time(s)
Select reference papers (PageRank)	$0.51^{0.25}$
Encode papers (TF-IDF, S-BERT, ProNE)	$1.39^{0.33}$
Cluster papers (Kernel k-means)	$0.48^{0.34}$
Generate labels (Automatic Labeling)	$0.29^{0.09}$

5.3 Discussions

Time Cost We further evaluate the time costs for different algorithms on a server with 56 Intel Xeon E5-2680 CPU cores and one GeForce RTX 2080 graphical card. We report the average running time and standard deviations for the generation of roadmaps described above in Table 8. The encoding process takes up the longest time, which is almost the total time of all the rest calculations. While only using CPU, the time cost for encoding papers rises up to 4.90 seconds which results in an average calculation time of around 6 seconds for each roadmap on our server. The calculation of S-BERT embeddings is the main calculation bottleneck. In addition to that, the time costs for retrieving paper metadata from open academic platforms differ a lot, concerning the network traffic and cache hit rate. In practice, this retrieving process takes up tens of seconds or even several minutes if the cache is completely missed.

Hyperparameters While most of our experiments are difficult to make fair comparisons between different configurations, the hyperparameters in Table 1 are mostly decided by case studies and user surveys. The increase of evolution tracks can slightly improve the clustering performance but we find it hard to display too many evolution tracks on screen for users. The topics of tracks also tend to overlap with each other with the increase of N_t as well. The number of selected publications is linearly related to the time cost of encoding papers, which is the major cost of the calculation of roadmaps. As a result, we cannot arbitrarily increase the number of publications. The other settings either make no significant difference to the performance or share the same tradeoff strategy as described above.

5.4 Case Study

Influential works like BERT [5] attract discussions from different fields and have profound impacts on subsequent studies. The generated roadmap for BERT is illustrated

in Figure 7. Label groups and importance scores for each *evolution track* are shown in Table 9.

TABLE 9
Details for evolution tracks in the BERT roadmap

	Evolution Track Labels	Importance
1	**Natural Language**, Language Model, Language Inference, Wide Range, NLP Task	45.88
2	**Language Model**, Natural Language, Neural Network, Vector Space, Learning Algorithm	19.91
3	**Reading Comprehension**, Question Answering, Model Achieves, Stanford Question Answering, Natural Language	14.90
4	**Machine Translation**, Source Sentence, Neural Network, Language Model, Neural Machine Translation	12.87
5	**Neural Network**, Deep Convolutional, Deep Convolutional Neural, Object Recognition, Computer Vision	5.84
6	**Deep Architecture**, Learning Deep, Unsupervised Learning, Learning Algorithm, Task Including	4.00

Two most significant evolution tracks in the roadmap of BERT, share the same topic of *Natural Language Model*, but focus on slightly different sub-topics, as shown in the label groups. *Reading Comprehension* and *Machine Translation* are task-oriented clusters with both model papers and datasets papers contained. Works like *OpenAI GPT* and *GloVe*, play particularly important roles in the evolution of BERT. The former one, frequently compared with BERT, is quite similar in functionality and model structure but has different implementations. The latter one, though only mentioned once in BERT, is highly connected to other references of BERT, can be seen as an essential milestone in the evolution process of various ideas.

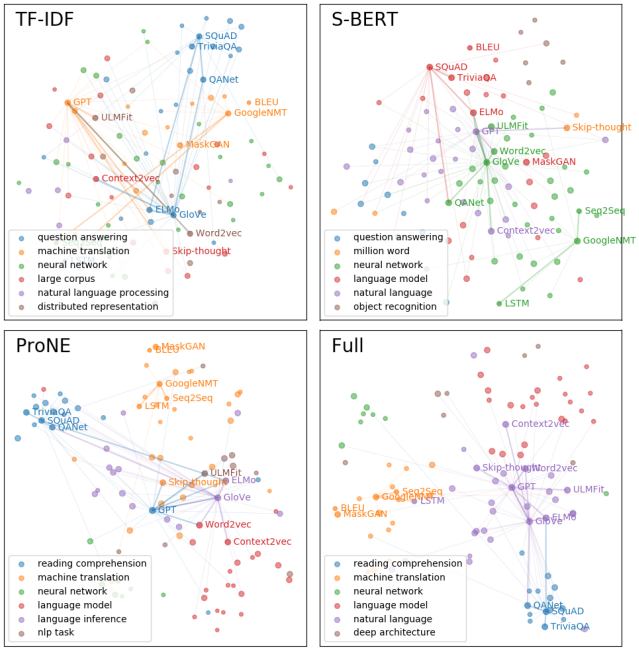


Fig. 8. The paper embeddings generated by different methods, plotted with t-SNE [47]. Node colors represent clustering results. The size of points represents the importance score. Part of the citation relationships are plotted as edges in the figure. The papers mentioned in Figure 1 are also annotated in this figure.

As a comparison, the reference paper embeddings produced by four different methods in the BERT roadmap are

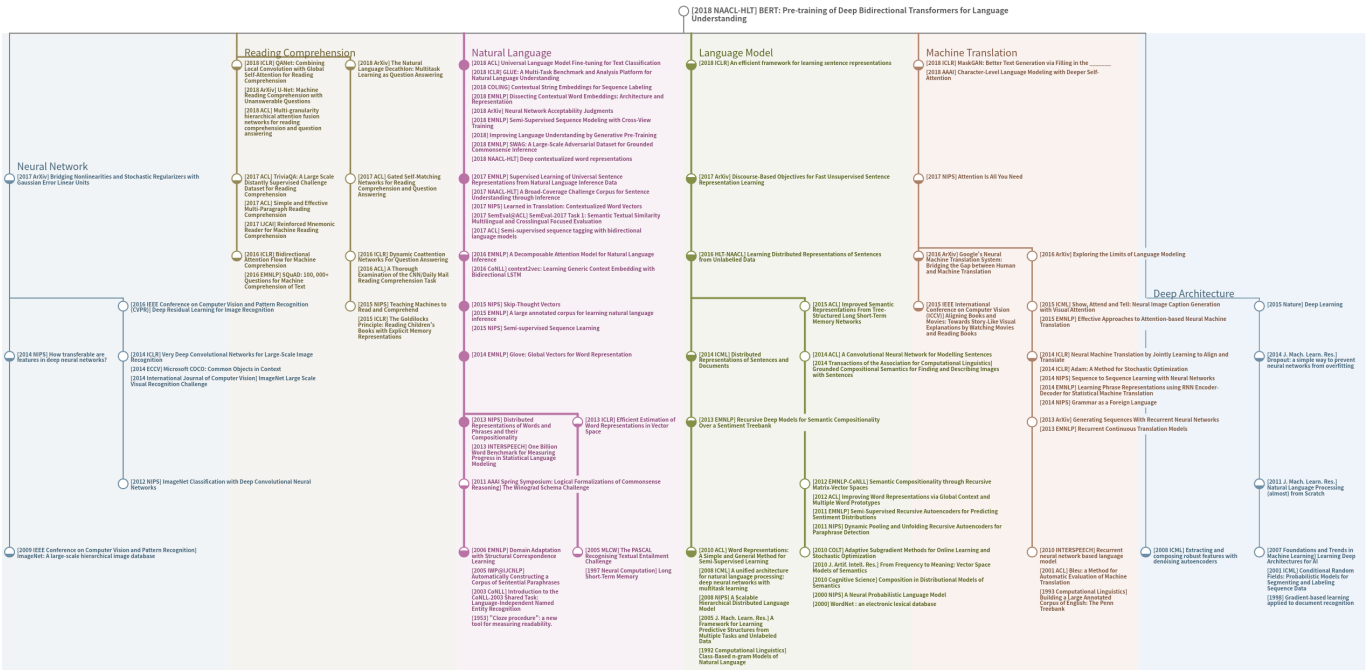


Fig. 7. A full static version of evolution roadmap for BERT. Each column represents one *evolution track* and papers are positioned by time order from top to bottom. Due to the page limit, we only show the most important label in each label group for every evolution track. The circle on the left side of papers is filled by the importance of papers adjacent to it. The more the circle is filled, the higher importance scores the papers on the right side will have. Evolution tracks are also positioned by their importance. The evolution tracks close to the center like **Natural Language** and **Language Model** are more important than evolution tracks on the sides. The recommendation function and more details for the evolution roadmap can be viewed on AMiner.

plotted in Figure 8. Using the additional adjacency matrix in kernel k-means ($\alpha = 1.0$ in Equation 8), we explicitly introduce the citation relationships as soft constraints in the clustering process, where we can see that the clustering results are much less coherent with the embedding positions generated by pure text encoding methods (TF-IDF and S-BERT) than others (ProNE and Full). The TF-IDF method, mainly focusing on the common words between papers, can only capture the literal meaning of paper contents. In the abstract of the ULMFit paper, the authors use *NLP* instead of *Natural Language Processing*, making it hard for TF-IDF to connect this paper with other natural language processing papers. This problem is solved in S-BERT as the BERT model can understand *NLP* and *Natural Language Processing* refer to the same thing. However, S-BERT meets difficulties in distinguishing NLP application papers. For example, the QANet paper, targeting the reading comprehension problem, heavily uses ideas from machine translation, such as back-translation. Thus, S-BERT categorizes it into the same group with other machine translation papers like GoogleNMT. The ProNE method, simply relies on the graph structure, is unable to make adequate decisions on papers with many graph links such as GloVe or GPT, since these papers are connected to almost every other cluster. The mixture of all three methods finally gives proper clusters to the reference papers.

Overall, the MRT framework can generate readable evolution roadmaps given the metadata of publications despite a few drawbacks. First, clustering with hard boundaries is not an optimal solution for many publications. For example, works like GloVe and LSTM contribute to multiple topics concurrently, which makes their hard clustering locations unstable and confusing. However, the soft assignment is not convenient for readers to recognize, especially when hundreds of nodes are provided. Second, low-quality phrases exist in cluster labels. Although it is possible to filter out phrases like "Task Including" using Part-Of-Speech (POS) tagging, these rules fail when high-quality candidates are dropped due to wrong POS tags. Since keywords often exist in publication titles that are not complete sentences usually, the bad precision of POS tagging or other methods frequently causes the low recall problem.

6 Deployed System

The MRT framework has been deployed onto AMiner[5] for free use. Over 10 thousand accesses have been made and around 700 evolution roadmaps have been generated by users. In practice, the generation of each roadmap could sometimes take tens of seconds due to the network retrieval of paper metadata and the calculation of paper embeddings without GPU, which makes it hard to provide synchronous online service. Therefore, we adopt a task queue and maintain a backend service to handle user requests asynchronously. The users will be notified by email on the completion of roadmap generation.

7 Conclusion

In this work, we proposed a novel framework called MRT to generate evolution roadmaps for publications helping researchers trace the movements of ideas lying behind. The framework is mainly composed of two parts: calculating embeddings and constructing roadmaps. Embedding methods are blended to encode both semantic and structural information. Semi-supervised clustering and automatic labeling techniques are applied to generate the annotated roadmap. Finally, a reinforce-based algorithm is developed for making recommendations helping readers locate their interested materials quickly. The integrated framework is evaluated through several experiments and the deployed system has attracted thousands of usage. The flexibility of MRT allows substituting internal modules with more advanced models without changing other parts. The proposed problem of *tracing evolution roadmaps* can be potentially extended to other domains, such as social networks, to help analyze the historical development process of various things. We leave better algorithm exploration and several raised problems for future work.

5. https://www.aminer.cn/mrt

Acknowledgments

The work is supported by the National Key R&D Program of China (2018YFB1402600), NSFC for Distinguished Young Scholar (61825602), and NSFC (61836013).

References

[1] R. Johnson, A. Watkinson, and M. Mabe, "The stm report," *An overview of scientific and scholarly publishing. 5th edition October*, 2018.

[2] H. Zha, W. Chen, K. Li, and X. Yan, "Mining algorithm roadmap in scientific publications," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 1083–1092.

[3] C. Zhang, F. Tao, X. Chen, J. Shen, M. Jiang, B. M. Sadler, M. T. Vanni, and J. Han, "Taxogen: Constructing topical concept taxonomy by adaptive term embedding and clustering," in *KDD 2018*, 2018.

[4] J. Shang, J. Shen, L. Liu, and J. Han, "Constructing and mining heterogeneous information networks from massive text," in *KDD '19*, 2019.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[7] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[8] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[9] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *ArXiv*, vol. abs/1802.05365, 2018.

[10] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[11] W. Fedus, I. Goodfellow, and A. M. Dai, "Maskgan: better text generation via filling in the_," *arXiv preprint arXiv:1801.07736*, 2018.

[12] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in neural information processing systems*, 2015, pp. 3294–3302.

[13] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 855–864.

[14] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[15] M. Valenzuela, V. A. Ha, and O. Etzioni, "Identifying meaningful citations," in *AAAI Workshop: Scholarly Big Data*, 2015.

[16] X.-D. Zhu, P. D. Turney, D. Lemire, and A. Vellino, "Measuring academic influence: Not all citations are equal," *ArXiv*, vol. abs/1501.06587, 2015.

[17] A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le, "Qanet: Combining local convolution with global self-attention for reading comprehension," *arXiv preprint arXiv:1804.09541*, 2018.

[18] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[19] A. Cohan, W. Ammar, M. van Zuylen, and F. Cady, "Structural scaffolds for citation intent classification in scientific publications," in *NAACL-HLT*, 2019.

[20] W. Yu, M. Yu, T. Zhao, and M. Jiang, "Identifying referential intention with heterogeneous contexts," *WWW*, 2020.

[21] Q. He, B. Chen, J. Pei, B. Qiu, P. Mitra, and L. Giles, "Detecting topic evolution in scientific literature: how can citations help?" in *CIKM*, 2009, pp. 957–966.

[22] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[23] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[24] R. Pappagari, P. Zelasko, J. Villalba, Y. Carmiel, and N. Dehak, "Hierarchical transformers for long document classification," *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 838–844, 2019.

[25] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *ArXiv*, vol. abs/2004.05150, 2020.

[26] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv:1908.10084*, 2019.

[27] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW '15*, 2015.

[28] J. Qiu, Y. Dong, H. Ma, J. Y. Li, C. Wang, K. Wang, and J. Tang, "Netsmf: Large-scale network embedding as sparse matrix factorization," *ArXiv*, vol. abs/1906.11156, 2019.

[29] J. Zhang, Y. Dong, Y. Wang, J. Tang, and M. Ding, "Prone: fast and scalable network representation learning," in *Proc. 28th Int. Joint Conf. Artif. Intell., IJCAI*, 2019, pp. 4278–4284.

[30] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017.

[31] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, "Semi-supervised graph clustering: a kernel approach," *Machine learning*, vol. 74, no. 1, pp. 1–22, 2009.

[32] H. Poostchi and M. Piccardi, "Cluster labeling by word embeddings and WordNet's hypernymy," in *ALTA*, 2018, pp. 66–70.

[33] R. Mihalcea and P. Tarau, "Textrank: Bringing order into texts," in *EMNLP 2004*, 2004.

[34] Q. Mei, X. Shen, and C. Zhai, "Automatic labeling of multinomial topic models," in *KDD '07*, 2007.

[35] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing atari with deep reinforcement learning," *ArXiv*, vol. abs/1312.5602, 2013.

[36] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, pp. 229–256, 1992.

[37] X. Zhao, L. Zhang, Z. Ding, D. Yin, Y. Zhao, and J. Tang, "Deep reinforcement learning for list-wise recommendations," *arXiv preprint arXiv:1801.00209*, 2017.

[38] M. Chen, A. Beutel, P. Covington, S. Jain, F. Belletti, and E. H. hsin Chi, "Top-k off-policy correction for a reinforce recommender system," in *WSDM '19*, 2018.

[39] G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie, and Z. Li, "Drn: A deep reinforcement learning framework for news recommendation," in *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 167–176.

[40] J. Carbinell and J. Goldstein-Stewart, "The use of mmr, diversity-based reranking for reordering documents and producing summaries," *SIGIR Forum*, vol. 51, pp. 209–210, 1998.

[41] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[42] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 990–998.

[43] W. Ammar and D. G. et al., "Construction of the literature graph in semantic scholar," *ArXiv*, vol. abs/1805.02262, 2018.

[44] D. M. Cer, M. T. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, "Semeval-2017 task 1: Semantic textual similarity - multilingual and cross-lingual focused evaluation," in *SemEval@ACL*, 2017.

[45] W. W. Daniel, "The spearman rank correlation coefficient," *Biostatistics: A Foundation for Analysis in the Health Sciences*, 1987.

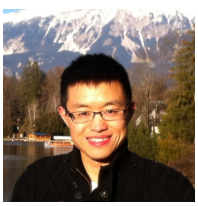
[46] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," 1956.

[47] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

Da Yin is a master candidate in the Department of Computer Science and Technology in Tsinghua University and the Computer Science Department in Carnegie Mellon University. He got his bachelor degree in Computer Science and Technology from Tsinghua University. His research interests include text mining and social network.

Weng Lam Tam is a master candidate in the Department of Computer Science and Technology in Tsinghua University. She got her bachelor degree in Computer Science and Technology from Tsinghua University. Her research interests include data mining and academic knowledge graph.

Ming Ding is a PhD student in the Department of Computer Science and Technology in Tsinghua University. He also got her bachelor degree in Computer Science and Technology from Tsinghua University. His research interests include graph learning, natural language processing and cognitive artificial intelligence. He has published many papers on top conferences, such as KDD, ACL, IJCAI, etc.



Jie Tang is a full professor with the Department of Computer Science and Technology of Tsinghua University. His main research interests include data mining, social network and machine learning. He has been a visiting scholar with Cornell University, Chinese University of Hong Kong, Hong Kong University of Science and Technology, and Leuven University. He has published more than 200 research papers in major international journals and conferences.