Set-aware Entity Synonym Discovery with Flexible Receptive Fields

Shichao Pei, Lu Yu, Xiangliang Zhang *Senior Member, IEEE*

Abstract—Entity synonym discovery (ESD) from text corpus is an essential problem in many entity-leveraging applications, e.g., web search and question answering. This paper aims to address three limitations that widely exist in the current ESD solutions: 1) the lack of effective utilization for synonym set information; 2) the feature extraction of entities from restricted receptive fields; and 3) the incapacity to capture higher-order contextual information. We propose a novel set-aware ESD model that enables a flexible receptive field for ESD by making a breakthrough in using entity synonym set information. The contextual information of entities and entity synonym sets are arranged by a two-level network from which entities and entity synonym sets can be mapped into the same embedding space to facilitate ESD by encoding the high-order contexts from flexible receptive fields. Extensive experimental results on public datasets show that our model consistently outperforms the state-of-the-art with significant improvement.

Index Terms—Entity Synonym Discovery, Entity Synonym Set, Graph Neural Network, Flexible Receptive Field.

1 Introduction

Recent years have witnessed a rapidly increasing amount of research attention and industry demand on entity-leveraging applications, which provide feasible approaches to understanding users' desires and organizing knowledge from multiple data sources in information systems. However, one major obstacle in entity understanding is that users or content creators always have different language habits to express their needs and thoughts by using various alternate forms, *i.e.*, entity synonyms, to reference one same real-world entity. For example, people can query the location of "Washington" with a question "Where is the District of Columbia?" or "Where is Washington or D.C.?". In this example, "District of Columbia" and "D.C." are different forms of the query entity "Washington". It is nontrivial to recognize entity synonyms. The string matching strategy fails when some synonyms have different surface forms, such as "D.C" and "Washington". Leveraging existing synonyms in knowledge bases is another inefficient way since most of such information is manually created and typically suffers from limited coverage and diversity. The demand for entity understanding and the difficulty of automatically recognizing synonyms motivate the study of entity synonym discovery (ESD). In particular, for a query entity, entity synonym discovery refers to finding a set of entities from structured or unstructured data, which have the same semantic meaning as the query entity but with different surface forms. ESD can connect each other the different surface forms for the same entity and benefit downstream applications such as web search and question answering by the linked text surfaces with less-noisy and well-recognized entities [1], [2], [3], [4].

Most of ESD pioneering works focus on structured data such as query logs [3], [5], which provide more focused entity information. Yet, discovering entity synonyms from raw text corpora is more practical and challenging, also can provide more abundant synonym information for real-world application. Solutions of ESD on unstructured data mainly involve two key components: 1) *entity feature extraction*, which transforms collected entity mentions in a text corpus to vectorized representations by leveraging different features [6], [7], [8], [9], [10]; and 2) *relation discrimination*, which tries to distinguish the relation between the query entity and candidate entities in the text corpus by ranking-based [7] or classification-based discriminators [11] with given synonym seeds as the training set. Although effective, current solutions have the following shortcomings:

- **The lack of effective utilization for synonym set information.** The given synonym seeds can be regarded as the distant-supervised training set, including entity pairs that identify the synonymous relation. Some entity pairs can be collected as a synonym set by transitivity. For instance, "D.C." is a synonym of "Washington" which is a synonym of "District of Columbia", so these three entities form a synonym set, which indicates that all entities in the set refer to the same real-world entity. The synonym set guides training and contains more comprehensive information about the referenced real-world entity than a single entity. For example, in Figure 1(a), "DC" and "D.C." have almost the same surface forms and the same contexts "The United States", it is toilless to misrecognize "D.C." as the synonym of "DC". Yet, it is not hard to tell that "Washington" and "District of Columbia" are not synonyms of "DC" if using synonym set S1 to replace "D.C.". Hence, "D.C." is not the synonym of "DC" because of the collective decision with the help of the synonym set S1. A majority of current solutions neglect the synonym set information. SynSetMine [11] is the unique one that designed a complicated set-instance classifier to compare the query entity with synonym sets

- *S. Pei, L. Yu and X. Zhang are with the Division of Computer, Electrical, and Mathematical Sciences and Engineering, King Abdullah University of Science and Technology (KAUST), Thuwal 23955-6900, Saudi Arabia. E-mail: {shichao.pei, lu.yu, xiangliang.zhang}@kaust.edu.sa.*
- *X. Zhang is the corresponding author.*

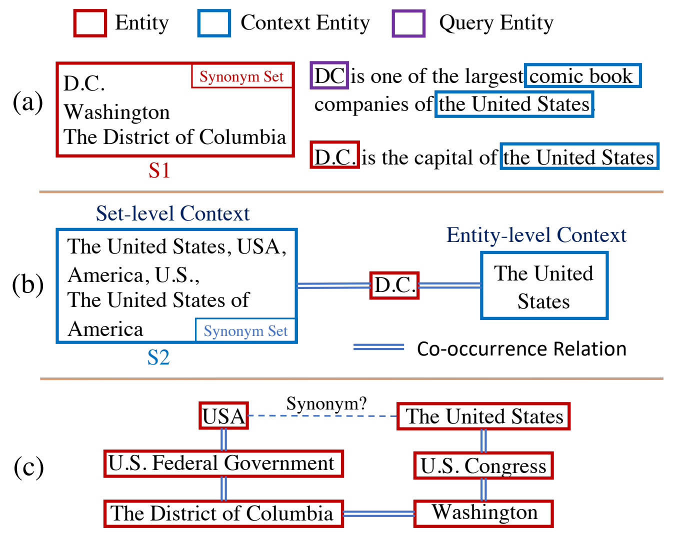


Fig. 1: A toy example of entity synonym discovery in the text corpus. (a) With the same surface form and context entity, ”DC” might be mismatched to ”D.C.” but will not happen if using a synonym set. (b) Set-level context provides more comprehensive information than entity-level context. (c) First-order and second-order proximity cannot model entity ”USA” and ”The United States” from different sentences across many other entities.

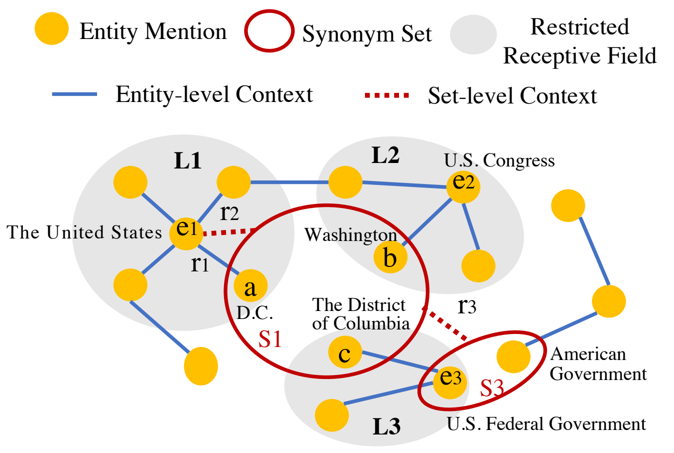


Fig. 2: An illustration of the restricted receptive field and two kinds of contextual relations. Node in yellow denotes entity mention. Edge denotes the contextual relation. The red circles S1 and S3 denote synonym sets. For instance, S1 is a synonym set of entities $fa; b; cg$. L1-L3 are restricted receptive fields defined by the local neighbors of e_1, e_2, e_3, respectively. Blue edge, like $r1$, refers to the entity-level contextual relation between two entities, and red dotted edges are the set-level contextual relations between two sets such as $r3$ or between an entity and a set such as $r2$.

on their surface forms without representation learning. Nevertheless, learning the representation for entities and synonym sets is an essential step in a unified framework because only a direct and straightforward comparison operator in the same feature space is needed instead of complex and manually designed classifiers. Also, the contextual information of synonym sets, which is crucially complementary information for surface forms and plays a significant role in matching entities, can be naturally captured by representation learning.

- **Restricted receptive field.** Entity feature extraction, as mentioned earlier, is a crucial component of ESD. Most of the existing works extract features for entities only from the entity-level contexts, e.g., learning feature vectors for entity e_1, e_2, e_3 from the *local restricted receptive field* L1, L2, and L3, respectively, in Figure 2. The entity-level context forms the entity-level contextual relation, such as the blue edge $r1$ in Figure 2. However, such restricted receptive fields confine the expressive power of the learned features because the set-level context has not been taken into consideration and it can provide more comprehensive contextual information than the entity-level context. For example, in Figure 2, entities a, b, and c are synonyms and form a synonym set S1. We can find that a is one of the entity-level contexts of entity e_1. Intuitively, the red circle S1 is a set-level context that contains more information about surface forms and contexts of the referenced real-world entity than single entity a. We can find another similar example in Figure 1(b), where a synonym set about “The United States” can enrich the contextual information of “D.C.”, which only has an entity-level context “The United States” in Figure 1(a). The set-level context forms the set-level contextual relation, such as red edges $r2$ and $r3$ in Figure 2. However, noisy contexts may be introduced into the receptive field of entities because of the expanded contextual scope. Hence, feature vectors of entities and synonym sets should be learned from a *flexible receptive field* that embodies all entity-level contexts and a part of set-level contexts. In other words, entities and synonym sets are allowed to flexibly capture the useful contextual information from the set-level context, to complement the entity-level context, and to simultaneously avoid the noise and enhance the expressiveness of learned feature vectors.
- **The incapacity to capture higher-order contextual information.** Collected entities from text corpus usually go across different sentences, even documents. Synonyms may be far from each other. As shown in Figure 1(c), “USA” and “The United States” come from different sentences, and it is hard to use first-order and second-order proximity [7], [12] to model their relationship because of the long distance between them. The n-order ($n > 2$) context [13], [14], which preserves more complex co-occurrence patterns, deserves more attention.

We target to break the above limitations by designing a *set-aware* model with the *flexible receptive field* for entity synonym discovery but face the following major challenges. First, entities in a synonym set usually have distinct features and lay in different areas of feature space. How to aggregate features of these entities and obtain the representation for a synonym set is problematic. The pooling mechanism has emerged as a powerful paradigm to aggregate features of elements in an arbitrary set. Nevertheless, the direct pooling would result in that the aggregated feature may be far from the original features of entities or ignore some features of entities. It could not represent the complete information of a synonym set. Second, the construction of flexible receptive fields needs to model the entity-level context and set-level context jointly. However, entities in text corpus and entity synonym sets form a complicatedly inclusive structure. It is challenging to arrange the connections between them, e.g., there are relations between entity e_1 and entity a, entity e_1 and set S1, set S1 and entity e_3, also set S1 and set S3 in Figure 2. Third, entities in text corpus and entity synonym sets should be mapped into the same feature space, and entity should form a compact cluster in the feature space if they are synonyms. However, encoding entities and synonym sets from the flexible receptive field causes difficulty in integrating the complex contextual structure.

The aforementioned challenges motivate us to propose an entity synonym discovery model named **EnSyn**, whose framework is presented in Figure 3. First, we construct a two-level network shown in Figure 3(b) to arrange the entity-level and set-level contextual relations to construct the flexible receptive field. Specifically, the upper level is constructed by entity mentions detected in text corpus based on entity recognition tools [7]. This level only provides entity-level contextual information for entities. The lower level is constructed by entity synonym sets (collected by entity linking tools [15]) and the entities which do not belong to any set (the entity can be regarded as a set that only contains the entity itself). This level provides set-level contextual information for entities and synonym sets. The links between two levels propagate the entity-level context of entities down to characterize the meaning of synonym sets and propagate the set-level context up to enhance the feature representation learning of entities. This process of *bidirectional propagation of contextual information* makes entities and synonym sets learn representations from flexible receptive fields in a mutually reinforced manner, rather than from the single lower level, to avoid the noise imported by enriched contextual information. For the two-level network, we propose a within-level model based on Graph Neural Networks (GNNs) [16] to encode high-order contextual information of each level, and design a between-level association model to implement the bidirectional propagation between two levels. Lastly, the entity feature extraction and relation discrimination stages are unified in a network representation stage of EnSyn where representations of entities and synonym sets are learned and mapped into the same feature space. The whole process is optimized by Bayesian Personalized Ranking (BPR) loss [17] instead of training extra relation discriminators.

We highlight the following unique advantages of our proposed **EnSyn** model:

- It considers the synonym set information and jointly learns the representations for entities and synonym sets by encoding the entity-level and set-level contextual information in a network representation learning framework.
- It allows a flexible receptive field for ESD, by learning two within-level models and one between-level association model. Also it captures the high-order contextual information of entities.
- Its evaluation on three real-world datasets shows its superior performance over the state-of-the-art methods, with significant improvement on ESD.

The rest of the paper is organized as follows: Section 2 presents the related work. We provide preliminaries and define the problem, meanwhile illustrate the framework of our proposed entity synonym discovery method in Section 3. We report the experimental design and results in Section 4 and Section 5 concludes our work.

2 Related Work

In this section, we discuss three lines of related work, including other solutions to ESD, a basic introduction of entity linking with the discussion about differences between ESD and entity linking, and different graph neural networks.

2.1 Entity Synonym Discovery

With the rapid development of entity-leveraging applications, e.g., web search and question answering, recognizing synonyms that refer to the same real-world entity has become an essential task for such applications. The discovered synonym helps understand the query from users.

Most ESD pioneering works focus on structured data such as query logs [3], [5], which provide more focused entity information. StrucSyn [18] explored the sub-query and resolved ambiguous entity names by proposing a heterogeneous graph-based data model that can capture the interactions between synonyms, web pages and keywords. Besides, various features for entities in the structured data have been proposed, such as query context similarity [19], [20], query click similarity [3], [5], [21], and pseudo-document similarity [19]. Our work, distinguishing with them, aims to discover entity synonyms from raw text corpora, which is more practical and challenging for real-world application. Ranking-based and classification-based approaches were proposed to distinguish between candidate entities and the query entity by representing entities with different features, such as co-occurrence statistics [8], textual pattern [6], distributional similarity [9], and semantic types of entities [10], [22]. Recently, Qu et al. [7] proposed integrating distributional features and textual patterns to discover the entity synonym with knowledge bases automatically. SurfCon [12] combined surface form information and distributional features for synonym discovery on privacy-aware clinical data. Our work focuses on learning the distributional features in a network representation model and can smoothly integrate the surface form. And SynonymNet [23] made use of multiple pieces of contexts in which the entity is mentioned, and compared the context-level similarity via a bilateral matching schema. SynSetMine [11] was the first to take the entity synonym set information into account. They developed a Set-Instance Classifier to distinguish between entity synonym sets and the query entity. However, the method did not learn the representation for synonym sets and ignore the set-level contextual information for entities and synonym sets.

2.2 Entity Linking

Entity linking or entity disambiguation is the task of mapping entity mentions in text corpus to corresponding entities in a given knowledge base, facilitating a variety of downstream applications such as knowledge base population

[24], question answering [25], and information integration [26]. Like ESD, early studies [24], [27], [28] applied a large number of hand-designed features to disambiguate entity mentions based on the lexical matching between mention's surrounding words and the entity profile in the reference knowledge base. With the surge of deep learning models, representation-based methods [29], [30] have shown effectiveness by modeling textual information and knowledge bases. Furthermore, factor graph models [31], [32] were proposed with the assumption of topical coherence. And the pairwise conditional random field [33], PageRank [34], and random walk [35] were employed to enhance entity linking performance. Moreover, some studies leveraged the sequence model [36] or reinforcement learning [37] to link mentions and entities in knowledge bases.

The main difference between ESD and entity linking is that entity linking aims to map entity mentions in text corpus to corresponding unique entities in a knowledge base. Still, ESD aims to find entities in text corpus that share the same semantic meaning with a given query entity. In other words, ESD does not leverage knowledge bases and assign any unique identity to the discovered entities but rather indicates whether they refer to the same entity or not. Although knowledge bases have been exploited to provide distance supervision in the recent ESD solutions [7], [11], knowledge bases are inherently not involved in the process of synonym discovery and only used for extracting training (synonym) seeds. Therefore, any method that can collect the training seeds can be used as a substitution for the knowledge bases. Following the previous work [7], [11], in our work, an entity linking tool is used as a sub-component in our framework to obtain the weak supervised training signal from knowledge bases.

2.3 Graph Neural Networks

Graph neural networks (GNNs) are a class of neural networks for learning from arbitrary structured graph data [38], [39]. It was proposed because of the success of Convolutional Neural Networks [40], and was developed due to the computational inefficiency and ability to generalize [41]. GNNs have been widely used in many tasks which include, but not limited to, social network prediction [16], [42], recommendation system [43], graph representation [44], graph classification [45], and sentiment classification [46].

Many GNN formulations are based on the notion of graph convolutional operations. Bruna et al. [47] first introduced the convolution operator in the Fourier domain by computing the eigendecomposition of the graph Laplacian. Then GCN [16] was proposed to simplify spectral convolutions by restricting the layer-wise convolution operation to alleviate overfitting on local neighborhood structures. Furthermore, due to the computational-consuming full graph Laplacian for large graph, GraphSage [42] proposed an inductive framework that generates embeddings by employing uniform neighbor sampling and feature aggregation from the local neighborhood. Also, PinSage [43] and FastGCN [48] attempted to resolve the limitation by leveraging the importance-based sampling approach. Meanwhile, AdaptGCN [49] designed an adaptive parameterized and trainable sampler to find sampling importance rather than fixed sampling methods. Besides, GAT [50] further applied the self-attention mechanism into the neighborhood aggregation step. JK [51] showed that the high-order propagation benefits graph representation learning. Moreover, several GNN-based methods have been designed for different graph types, such as directed graph [52], heterogeneous graph [53], dynamic graph [54], and edge-informative graph [55]. Yet, to the best of our knowledge, we are the first to explore utilizing GNNs for entity synonym discovery.

TABLE 1: Notations

Symbol	Description
e	An entity
$\mathcal{S}$	An entity synonym set
$\mathcal{D}$	Text corpus
$\mathcal{E}$	A set of entity mentions
G_e	Entity-level co-occurrence network
G_s	Set-level co-occurrence network
v	A node for a set $\mathcal{S}$
$\mathcal{R}$	A set of entities that do not belong to any set $\mathcal{S}$
$\mathcal{I}$	A set of between-level relations
$\mathcal{C}$	A set of nodes in graph G_s
$\mathcal{K}$	A knowledge base

3 Methodology

In this section, we first introduce the preliminaries, and formulate the task, then discuss our proposed method EnSyn. Figure 3 shows its overall framework which involves, 1) data preparation, including collecting entity mentions and synonym seeds using existing entity recognition and entity linking tools; 2) constructing a two-level network using the collected entity mentions and extracted synonym sets from the synonym seeds; 3) learning the representation of entity mentions and entity synonym sets by unifying within-level models and a between-level association model; and 4) inference stage. We will discuss each part in detail.

3.1 Preliminaries

We first present some definitions and the data preparation process, and then describe the construction of two-level network. Notations used throughout the paper are summarized in Table 1.

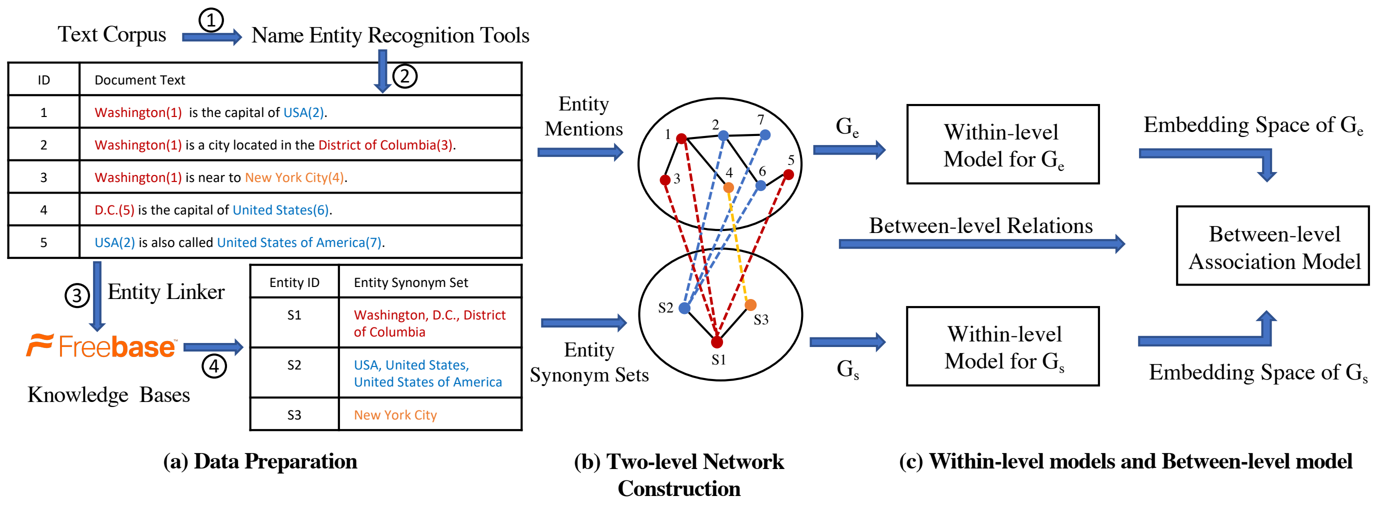
Definition 1. *Entity. An entity e represents a concept or a subject in the real-world. An entity can be a word or a phrase and can be used in a sentence.*

Definition 2. *Entity Synonym. For an entity e_i in a text corpus, its synonym refers to entity e_j, which has a different surface form but has the same semantic meaning as e_i.*

Definition 3. *Entity Synonym Set. An entity synonym set S_i contains a set of entities $\{e_{i_1} \dots e_{i_n}\}$ that are synonyms.*

Definition 4. *Entity Synonym Seeds. A set of entity pairs identifying the synonymous relation between two entities, which can be used as the training set for ESD.*

Definition 5. *Knowledge Base. A knowledge base K consists of numerous facts about a set of entities. In our work, we only focus on the entity synonym facts, which are collected as the synonym seeds to discover previously unknown synonyms.*



ID	Document Text
1	Washington(1) is the capital of USA(2).
2	Washington(1) is a city located in the District of Columbia(3).
3	Washington(1) is near to New York City(4).
4	D.C.(5) is the capital of United States(6).
5	USA(2) is also called United States of America(7).

Entity ID	Entity Synonym Set
S1	Washington, D.C., District of Columbia
S2	USA, United States, United States of America
S3	New York City

Fig. 3: The framework of the proposed model for entity synonym discovery. In detail, the framework includes (a) data preparation: extracting the entity mentions and obtaining the synonym sets; (b) Two-level network construction: organizing the entity-level contextual information and set-level contextual information; (c) Within-level and Between-level association models: encoding the contextual information from the flexible receptive field.

Definition 6. *Set-level Context. Set-level context refers to a kind of relationship in a co-occurrence network. If entity e has a link with a synonym set in the co-occurrence network, we regard the synonym set as the set-level context of entity e. Likewise, if a synonym set S_1 has a link with synonym set S_2 in the co-occurrence network, we regard the synonym set S_2 as the set-level context of S_1.*

Definition 7. *Receptive Field. The receptive field of entity e_i in a co-occurrence network is a set of first-order neighbors of e_i.*

3.1.1 Data Preparation

Building an ESD model needs training data and validation data for evaluating the performance of the ESD model. Therefore, we start with data preparation, which recognizes the entity mentions in the text corpus, collects synonym seeds, and prepares entity synonym sets. We organize all collected synonym seeds in the form of synonym set, and use a part of them to **train** the proposed EnSyn model. The remaining is used for **evaluation**, verifying if the discovered synonyms by EnSyn are correct. In detail, we follow the procedure in [7] to automatically collect the synonym seeds. First, existing named-entity recognition tools [56] are employed to detect *entity mentions* in the given text corpus D. We denote the set of detected entity mentions as E. Then an existing entity linker such as DBpedia Spotlight [15] can be leveraged for linking the detected entity mentions to a given knowledge base K. Furthermore, the synonym seeds are collected from the linked corpus. All entity mentions linked to the same entity in K are grouped as an entity synonym set. Finally, we obtain *entity mentions* E and *entity synonym sets* $\{S_1 ... S_n\}$ for the next network construction.

3.1.2 Two-level Network Construction

1) **Entity-level co-occurrence network construction**: We first leverage all detected entity mentions E as nodes to construct an *entity-level co-occurrence network* G_e. Specifically, for each entity e_i in E, we link it with its k-nearest entity mentions in the text corpus as edges. The constructed network G_e including entity mentions E and their co-occurrence relations implies the entity-level contextual information.

2) **Set-level co-occurrence network construction.** Next, we construct a *set-level co-occurrence network* G_s to embody the set-level contextual information. Starting from a set of entity synonym sets $\{S_1 ... S_n\}$, we create a set of nodes $V = \{v_1 ... v_n\}$, where v_i denotes a set S_i. We then copy the set of entities $R = E \setminus \{S_1 \cup ... \cup S_n\}$ from G_e. Note that entities in R do not belong to any set S, or in other words, each entity in R is a set that only contains one entity by itself . Hence, we use all elements in R and V as nodes of network G_s. The *set-level co-occurrence network* G_s is then constructed on nodes $C = V \cup R$. For edges, v_i is linked with v_j if entities in S_i and S_j have an edge in G_e (e.g., $S1$ and $S2$ in Figure 3 are linked because 1 and 2 in G_e have an edge). Besides, nodes in R are linked with any v_i or any others in R if their corresponding entities (linked by the between-level relations discussed below) in G_e have an edge (e.g., $S3$ in Figure 3 is linked with $S1$ because 4 in G_e is linked with 1). The constructed network G_s implies set-level contextual information of synonym sets and entities.

3) **Between-level relation construction.** Every v_i in G_s is linked with e_l in G_e if e_l belongs to S_i (e.g., $S1$ in Figure 3 is linked with $\{1,3,5\}$ in G_e). Entities in R are linked with their copies in G_e (e.g., $S3$ in Figure 3 is linked with 4 in G_e). We denote such links as between-level relations I .

Therefore, the two-level network with G_e, G_s and I preserves the complete entity-level and set-level contextual information, and provides a flexible receptive field by the bidirectional propagation of contextual information on I for further entity and set encoding.

3.2 Task Formulation

The entity synonym discovery task aims to find synonyms for a given query entity. Specifically, with a given query entity in entity mentions E, previous solutions aim at finding out the most relevant candidates in E as the discovered synonyms. Because of the utilization of the synonym sets and the advantage of direct comparison between embeddings of entities and synonym sets in the same feature space, our work aims to find out the most relevant candidate in C of G_s as the discovered synonyms with a given query entity. We unify the representation learning of entities and their relation discrimination in a single objective function, and the constructed G_e and G_s can capture the entity-level and set-level contextual information. We use the relations presented in the two-level network for learning entity and set embeddings which serve to discover unknown relations among entities, i.e., synonyms. We next introduce our models for entity and synonym set embedding with within-level relations (in G_e and G_s) and between-level relations (between G_e and G_s).

3.3 Within-level Model

The aim of the within-level model is to encode the entity-level contexts of G_e and the set-level contexts of G_s into separated embedding spaces. We define the within-level model as:

$$\mathbf{E} = f_{\text{within}}(G_e) \quad \text{and} \quad \mathbf{C} = f_{\text{within}}(G_s) \tag{1}$$

where $\mathbf{E}$ is the embedding of entities in G_e, and $\mathbf{C}$ is the embedding of nodes in G_s, function f_{within} is the within-level model. Taking network G_e as an example, we build upon the message-passing architecture of GNN [42] in order to capture contextual information along the network structure and refine the distributional representation of entities. We encode network G_s in the same way but another GNN with different weights. We first illustrate the design of first-order propagation and then generalize it to high-order propagation which benefits the network representation learning [51].

3.3.1 First-order Propagation

For a connected entity-entity pair (e_i, e_j), we define the message from i to j as:

$$\mathbf{m}_{j\leftarrow i} = f_m(\mathbf{e}_i, \mathbf{e}_j, p_{ij}) \tag{2}$$

where $\mathbf{e}_i$ and $\mathbf{e}_j$ are the embedding of e_i and e_j, $\mathbf{m}$ is the information to be propagated and $\mathbf{m}_{j\leftarrow i}$ is the message embedding. f_m is the message encoder which takes $\mathbf{e}_i$ and $\mathbf{e}_j$ as the input, and p_{ij} is introduced to control the decay factor on each propagation on pair (e_i, e_j), because the message being propagated should decay with the path length. In our work, we define f_m as follow:

$$\mathbf{m}_{j\leftarrow i} = \frac{1}{\sqrt{|N_i||N_j|}}(\mathbf{W}_1\mathbf{e}_i + \mathbf{W}_2(\mathbf{e}_i \quad \mathbf{e}_j)) \tag{3}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are trainable weight parameters. We set p_{ij} as the graph Laplacian norm $1/\sqrt{|N_i||N_j|}$ following the graph convolutional network (GCN) [16], where N_i and N_j refer to the first-hop neighbors of entity e_i and e_j, respectively. And we design that the propagated message not only depends on the e_i, like the conventional GCN models, but also encodes the affinity between e_i and e_j, with   indicating the element-wise product in Eq. (3). This operator encodes the message with more information about the pair instead of only the source entity.

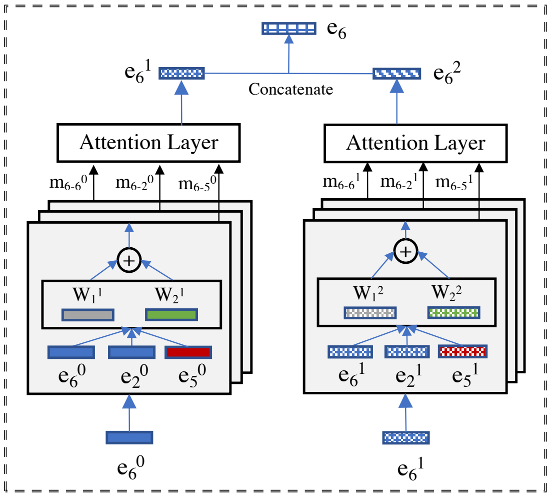


Fig. 4: An example of using entity e_6 to describe a simple two-layer embedding propagation with the self-attention layer.

Next, we define the message aggregation operator. In particular, we aggregate messages propagated from e_j's neighborhood to update e_j's representation as follow:

$$\mathbf{e}_j^1 = \text{LeakyReLU}(\mathbf{m}_{j\leftarrow j} + \sum_{i\in\mathcal{N}_j} \mathbf{m}_{j\leftarrow i}) \tag{4}$$

where message passed over the activation function of LeakyReLU [57], and $\mathbf{e}_j^1$ denotes the embedding of entity e_j obtained after the first propagation layer. All messages coming from N_j (neighbors of e_j) are aggregated in Eq. (4). Meanwhile, we take the self-connection of e_j into consideration, for maintaining the information of original features. The self-connection message is:

$$\mathbf{m}_{j\leftarrow j} = \mathbf{W}_1\mathbf{e}_j \tag{5}$$

where $\mathbf{W}_1$ is the same weight parameter with that in Eq. (3). However, due to the fact that contexts usually have different importance scores which reflect the different relevancy and proximity between entities. We further design an attention-based message aggregation operator following graph attention network [50] as an alternative aggregator as follow:

$$\mathbf{e}_j^1 = \text{LeakyReLU}(\ _{jj}\mathbf{m}_{j\leftarrow j} + \sum_{i\in\mathcal{N}_j} \ _{ij}\mathbf{m}_{j\leftarrow i}) \tag{6}$$

where   is the attention coefficient computed as follow:

$$_{ij} = \text{softmax}_i(\mathbf{m}_{j\leftarrow i}) = \frac{\exp(\ _1(\mathbf{m}_{j\leftarrow i}))}{\sum_{k\in\{\mathcal{N}_j\cup\{j\}\}}\exp(\ _1(\mathbf{m}_{j\leftarrow k}))} \tag{7}$$

Note that the self-attention parameter is calculated by a masked manner, *i.e.*, it only computes attention weights for the neighborhoods of entity e_j. The different importance scores are assigned to the neighborhood, and the embedding for entity e_j can be obtained via the self-attention mechanism. We apply LeakyReLU as the activation function $_1$. Next we discuss how to extend the first-order propagation to high-order propagation.

3.3.2 High-order Propagation

To capture the high-order contextual information, we stack more propagation layers [51]. By stacking l propagation layers, an entity can receive the messages propagated from its l-hop neighbors. We de ne the embedding of entity e_j and the messages at l-th propagation as:

$$e_j^l = \text{LeakyReLU}(m^l_{j \; j} + \sum_{i \in N_j} m^l_{j \; i}) \quad (8)$$

$$e_j^l = \text{LeakyReLU}({}^l_{jj} m^l_{j \; j} + \sum_{i \in N_j} {}^l_{ij} m^l_{j \; i}) \quad (9)$$

$$m^l_{j \; i} = \frac{1}{\sqrt{|N_i||N_j|}}(W_1^l e_i^{l \; 1} + W_2^l (e_i^{l \; 1} \quad e_j^{l \; 1})) \quad (10)$$

$$m^l_{j \; j} = W_1^l e_j^{l \; 1} \quad (11)$$

where W_1^l and W_2^l are the trainable weight parameters. $e_i^{l \; 1}$ is the entity e_i's embedding generated from the $(l$-1)-th message propagation, and contains the messages propagated from its $(l$-1)-hop neighbors. ${}^l_{jj}$ and ${}^l_{ij}$ are self-attention coef cients at l-th propagation. With the representations $e_j^1 ::: e_j^l$ from propagation layers $\{1 ::: l\}$, we concatenate them to compose the nal representation $e_j = [e_j^1; :::; e_j^l]$ for entity e_j, because concatenation can preserve the representations re ecting the connectivity between the entity and its varying order neighbors. A simple example is shown in Figure 4 to illustrate the structure of two-layer embedding propagation. Similarly, we can obtain the high-order representation of nodes in network G_s.

Two entities with a link are expected to be closer in their embedding space than those without a link. We thus de ne the objective function of within-level model based on BPR loss [17], which has been widely used in pairwise ranking models. For G_e, the objective function is

$$L_{G_e} = \sum_{(e_i;e_j;e_n) \in T_e} \ln \sigma(\hat{y}^e_{ij} \quad \hat{y}^e_{in}) + {}_e \| {}_e \|_2^2 \quad (12)$$

where $\hat{y}$ predicts the relation of two entities: $\hat{y}^e_{ij} = e_i^T e_j$. $T_e = \{(e_i; e_j; e_n) | (e_i; e_j) \in Z^+; (e_i; e_n) \in Z \quad \}$ indicates the pairwise training data, Z^+ denotes the observed links and Z denotes the unobserved links; σ refers to the sigmoid function, ${}_e$ controls the strength of L_2 regularization to prevent over- tting, ${}_e$ is all trainable parameters of network for G_e. We de ne the same objective function for G_s as L_{G_s} as follow:

$$L_{G_s} = \sum_{(c_i;c_j;c_n) \in T_s} \ln \sigma(\hat{y}^s_{ij} \quad \hat{y}^s_{in}) + {}_s \| {}_s \|_2^2 \quad (13)$$

where $T_s = \{(c_i; c_j; c_n) | (c_i; c_j) \in Z^+; (c_i; c_n) \in Z \quad \}$ and $\hat{y}^s_{ij} = c_i^T c_j$.

3.4 Between-level Association Model

The aim of the between-level association model is to capture the associations between the entity-level network G_e and set-level network G_s based on the between-level relations I in the two-level network. Due to the restricted receptive elds in G_e and the desire to mitigate the effect of noise in the enriched set-level contexts in G_s, the between-level relations can assist the representation learning from the exible receptive eld so that entities can learn to select and balance the contextual information from the entity-level and set-level contexts. In particular, contextual information in entity-level and set-level networks can be bidirectionally propagated on the between-level relations to refresh the representations of entities and sets. Next, we devise two kinds of between-level assoication models which adopt different strategy to associate entities and sets, respectively.

3.4.1 Transformation-based Association Model

We rst propose a transformation-based approach, which aligns embedding spaces of G_e and G_s obtained from the previous within-level models. Speci cally, the embedding of an entity e in G_e will be mapped to an embedding in embedding space of G_s after transformation. The motivation is that the embedding of e should be close to the embedding of c in G_s if there is a link between them in I. We de ne the transformation as below:

$$c \quad f_T(e); \forall (e; c) \in I \quad (14)$$

$$f_T(e) = {}_2(W_T e + b_T) \quad (15)$$

where f_T is a non-linear af ne transformation function, W_T is a weight parameter and b_T is a bias vector. We apply tanh as the activation function ${}_2$.

The loss function of the transformation-based association model is de ned based on margin-based ranking loss, and it aims at making the score of positive samples smaller than the score of negative samples:

$$L^T_{Bet} = \frac{1}{|I|} \sum_{\substack{(e;c) \in I \\ (e;c^0) \notin I}} \quad {}^T + \|c \quad f_T(e)\|_2 \quad \|c^0 \quad f_T(e)\|_2 \quad {}_+ \quad (16)$$

where $[x]_+ = \max\{0; x\}$ denotes the positive part of x, ^{T} is a margin hyper-parameter which is greater than 0, and c^0 is a negative sample of e.

3.4.2 Grouping-based Association Model

Different from transformation-based association model, the grouping-based model assumes that the network G_e and network G_s can be embedded into the same space, and forces any entity $e \in E$ to be close to its corresponding set $c \in C$. Speci cally, the association loss for a given pair of between-level relation $(e; c)$ is de ned as the distance between the embeddings of e and c compared with margin ^{G}. The loss is de ned as follow:

$$L^G_{Bet} = \frac{1}{|I|} \sum_{(e;c) \in I} \quad \|c \quad e\|_2 \quad {}^G \quad {}_+ \quad (17)$$

3.5 Optimization and Inference

The overall objective function combines that for the within-level models and the between-level model, as follows:

$$L = L_{G_e} + L_{G_s} + \quad L_{Bet} \quad (18)$$

The optimization process is given in Algorithm 1. Embeddings are initialized by Xavier initializer [58]. We leverage SGD as our optimizers and iteratively train the three

Algorithm 1: EnSyn

```
Input: Co-occurrence networks $G_e$ and $G_s$, the set of
       between-level relations $I$.
Output: Embedding of entities $E$ and sets $C$.
1  Initialize embeddings $E$, $C$, and weight parameters for
     within-level model and between-level association
     model;
2  for iteration = 1, ... MaxIter do
       // Within-level model for $G_e$.
3      for batch = 1, ... NumBatch_$G_e$ do
4          Sample a batch $T_e$ of triples $(e_i; e_j; e_n)$ in $G_e$;
5          Update embedding $E$, according to $L_{G_e}$ with
             the batch $T_e$.;
6      end
       // Within-level model for $G_s$.
7      for batch = 1, ... NumBatch_$G_s$ do
8          Sample a batch $T_c$ of triples $(c_i; c_j; c_n)$ in $G_s$;
9          Update embedding $C$, according to $L_{G_s}$ with
             the batch $T_c$.;
10     end
       // Transformation-based between-level
          association model.
11     for batch = 1, ... NumBatch_$I$ do
12         Sample a positive batch $\hat{I}_p$ and a negative batch
             $\hat{I}_n$ from $I$;
13         Update $W_T$, $b_T$ according to $L^T_{Bet}$ with the
             batch $\hat{I}_p$ and $\hat{I}_n$.
14     end
15 end
```

TABLE 2: Used Datasets and Knowledge Bases.

Dataset	Wiki	NYT	PubMed
# Documents	100,000	118,664	1,554,433
# Sentences	6,839,331	3,002,123	15,051,203
# Entity Mentions	98,664	31,702	96,588
# Entity Synonym Sets	4,920	1,494	17,972
# Edges in G_e	7,466,706	2,665,409	9,892,634
# Edges in G_s	7,257,121	2,581,956	8,840,049
# Training Entities	9,886	2,999	37,058
# Test Synonym Sets	615	184	5,408
# Test Entities	755	222	8,009
Knowledge base	Freebase	Freebase	UMLS

components until convergence. In the inference stage, a given query entity e in G_e is matched with candidates c in network G_s, based on their inner product value of embedding vectors: $e^T c$ (grouping-based association model) or $c^T f_T(e)$ (transformation-based association model).

4 EXPERIMENTS

In this section, we conduct experiments on three real-world datasets different in size and evaluate our proposed method for entity synonym discovery. Specifically, we aim to answer the following research questions:

- RQ1: How does our proposed EnSyn perform as compared with state-of-the-art approaches?
- RQ2: How can set-level co-occurrence network and the surface information (i.e., text information of each entity) be helpful for improving EnSyn?
- RQ3: How do the attention layer and different between-level association models influence our proposed model?
- RQ4: How do different hyper-parameter settings affect our proposed model?

4.1 Experimental Design

We first describe the datasets, baseline methods, and model variants, also outline the experimental settings of our proposed EnSyn in the section.

4.1.1 Datasets.

We use public benchmark datasets provided in [7], [11]:

- Wiki : 100K articles from Wikipedia;
- NYT : 118,664 articles from 2013 New York Times;
- PubMed : about 1.5M paper abstracts from PubMed dataset.

DBpedia Spotlight [15] is applied as the entity linker for Wiki and NYT datasets, and PubTator [59] is used for PubMed. The entity mentions which appear less than ten times are filtered out. Freebase [60] and UMLS [61] are used as the knowledge base, shown with the datasets summary in Table 2.

4.1.2 Baselines.

To comprehensively evaluate the effectiveness of our proposed method, we include two categories of methods for performance comparison:

1) Context-based methods:

- Word2vec [62]: A distributional word embedding method. We obtain the embedding by doing SVD decomposition over the PPMI co-occurrence network [63]. We train a bilinear scoring function following [7], taking obtained embeddings as features;
- Node2vec [64]: A widely used graph embedding method. It defines a flexible notion of the neighborhood and proposes a biased random walk algorithm to learn the representation. The same bilinear scoring function is trained using obtained embeddings as features;
- DPE [7]: A method combines textual patterns and distributional features. We first adopt the version DPE-NoP without textual patterns as a context-based method, then employ the full version of DPE for the comparison.
- Hierarchical Multi-Task Word Embedding (HMT) [22]: A method jointly models the neighboring word semantic type and neighboring word. We leverage the version without type prediction due to the lack of type information in the datasets we used.
- SynonymNet [23]: A method makes use of multiple pieces of contexts in which the entity is mentioned, and compares the context-level similarity via a bilateral matching schema.

2) Surface-involved methods: To validate the effectiveness of our proposed method with surface information, we compare with several methods that utilized the surface form information.

- CharNgram [65]: we take the average of the pretrained character n-gram embeddings as the feature for each entity, then train the same bilinear scoring function as above.
- Concept Space [10]: A medical synonym extraction method jointly leverages word embeddings and heuristic rule-based string features.
- Planetoid [66]: An inductive network embedding method incorporates embedding techniques into the

TABLE 3: Ranking performance comparison (the best results of baselines are marked as * along with underline). The results in bold are our method and the improvement over the best baseline algorithm.

Methods	Wiki			NYT			PubMed		
Measures	Hits@5	NDCG@5	MRR	Hits@5	NDCG@5	MRR	Hits@5	NDCG@5	MRR
Word2vec	0.2386	0.1854	0.1743	0.2063	0.1472	0.1537	0.3546	0.2915	0.2745
Node2vec	0.2644	0.2087	0.1952	0.2336	0.1698	0.1709	0.3622	0.2896	0.2663
HMT	0.2843	0.2238	0.2197	0.2301	0.1724	0.1685	0.3682	0.3001	0.2845
DPE-NoP	0.3125	0.2404	0.2369	0.2486	0.1989	0.1885	0.3895	0.3227	0.3162
SurfCon w/o pre	0.3446	0.2815	0.2656	0.2719	0.2308	0.2008	0.3956	0.3394	0.3282
DPE	0.3574	0.2845	0.2781	0.2794	0.2275	0.2104	*0.4092	0.3457	0.3324
SynonymNet	*0.3726	*0.3104	*0.2875	*0.3035	*0.2527	*0.2368	0.4073	*0.3521	*0.3384
EnSyn	0.4759	0.3722	0.3635	0.3605	0.2996	0.2714	0.4753	0.4196	0.3850
Improvement %	27.72	19.91	26.43	18.78	18.56	14.61	16.15	19.17	13.77
EnSyn-restricted	0.3662	0.3006	0.3001	0.2966	0.2449	0.2221	0.4322	0.3663	0.3408
EnSyn-G w/o Att	0.4395	0.3420	0.3280	0.3242	0.2661	0.2458	0.4460	0.3928	0.3542
EnSyn-G	0.4576	0.3580	0.3461	0.3438	0.2855	0.2612	0.4585	0.4032	0.3690
EnSyn w/o Att	0.4623	0.3614	0.3521	0.3408	0.2878	0.2568	0.4671	0.4080	0.3761
CharNgram	0.4411	0.3557	0.3399	0.4414	0.3288	0.3091	0.4140	0.2992	0.2966
Concept Space	0.4052	0.3093	0.2874	0.3226	0.2658	0.2471	0.3743	0.3016	0.2987
Planetoid	0.4524	0.3461	0.3376	0.4435	0.3310	0.3038	0.4225	0.3086	0.3162
SurfCon	*0.4608	*0.3664	*0.3469	*0.4498	*0.3361	*0.3134	*0.4354	*0.3562	*0.3381
EnSyn with pre	0.5270	0.4153	0.4021	0.4651	0.3475	0.3248	0.4781	0.4052	0.3770
Improvement %	14.37	13.35	15.91	3.40	3.39	3.63	9.81	13.76	11.51

graph-based semi-supervised learning setting. We use the output of CharNgram [65] as the input feature and take the intermediate hidden layer as representations. Similarly, a bilinear scoring function is used for training the model.

SurfCon [12]: An approach utilizes surface form information and entity-level context information. A version without the pretrained surface form information, SurfCon w/o pre is also evaluated as one addition baseline in Context based methods.

We do not compare with SynSetMine [11], as it focuses on clustering and cannot generate the embedding for entities. Also, it only models entities in the synonym set rather than all entity mentions from a text corpus.

4.1.3 Model Variants

To validate the effectiveness of different components of EnSyn, we use EnSyn to denote the model with attention layers and the transformation-based between-level association model. EnSyn-G is the model with attention layers and the grouping-based between-level association model. EnSyn w/o Att refers to the model EnSyn without attention layers. Similarly, EnSyn-G w/o Att refers to the model EnSy-G without attention layers. Furthermore, EnSyn with pre denotes the model EnSyn with the pretrained n-gram embedding for initialization. EnSyn-restricted is a variant of EnSyn without using the set-level network G_s, but only the entity-level network G_e. Speci cally, we train EnSyn-restricted by a bilinear scoring function, taking obtained embeddings as features.

4.1.4 Experimental Settings

We adopt widely-used evaluation metrics: Mean Reciprocal Rank (MRR), Hits, and Normalized Discounted Cumulative Gain (NDCG) for evaluating entity synonym discovery task. Large values indicate better performance. In our evaluation, Hits@N measures whether the testing entity is ranked in the Top-N list with 1 for yes and 0 for no, and NDCG@ N accounts for the position of the hit by assigning a higher score to the hit at top positions, while MRR is the average of the reciprocal ranks of the returned list.

We implement the proposed EnSyn framework using the Python library Tensor ow and conduct all the experiments on a Linux server with GPUs (GeForce RTX 2080 Ti) and CPU (Intel Xeon). We set the dimension of entity embedding as 100 for all methods and the dimension of character n-gram as 100. We nd the optimal parameters or follow the setting in the original paper of baselines. For our EnSyn method, we apply a grid search for hyper-parameters and nd the best con guration: the coef cient $_e$ and $_c$ are 0.001, the number of layers is 4, margin hyper-parameter ^{G} and ^{T} is 0.5, window size k is 5, trade-off parameter is 1.0. We use Adam optimizer [67] to optimize the loss function L with learning rate 0.001. Meanwhile, we use L_2 norm to avoid over- tting. We randomly sample 50% of entities in entity synonym sets as the training set and use the rest of entities for testing, and randomly sample 10,000 non-synonyms as negative samples, then mix them with the synonym for each query entity in testing for ef ciency [12]. Each evaluation is repeated ve times, and averaged results are reported.

4.2 Performance Evaluation Results (RQ1)

In this section, we explore if our proposed EnSyn outperforms the state-of-the-art approaches. Table 3 shows the experimental results. In the rst part of Table 3, we can observe our proposed EnSyn consistently outperforms all baseline methods on all datasets under different evaluation metrics. Speci cally, our method built upon GNN can capture the high-order contextual information from the exible receptive eld. Hence, it outperforms those only modeling the rst-order entity-level contextual information, as shown in the comparison of results between EnSyn and the rst three baselines in Table 3. Furthermore, SurfCon

Fig. 5: Entity synonym discovery evaluation with different values of N on NDCG@N and Hits@N.

w/o pre and SynonymNet have better results than other baselines because it adopts a context matching mechanism to take second-order entity-level contextual information into consideration. Besides, SynonymNet makes use of multiple pieces of contexts, which could provide more contextual information than that in SurfCon w/o pre. However, they still ignore the essential entity synonym set information and the set-level contextual information, so it is inferior to our method. Besides, Figure 5 shows that EnSyn is stably superior to other baselines with different lengths of the retrieval list.

4.3 Component Analysis (RQ2 and RQ3)

First, we investigate how the synonym set information in the set-level network G_S and the surface information are helpful for improving EnSyn. To answer the question, we compare EnSyn with its variants EnSyn-restricted and EnSyn with pre. In the rst part of Table 3, EnSyn is consistently better than EnSyn-restricted, showing the contextual information from the restricted receptive eld is insuf cient for learning representation, also demonstrating the usefulness of the exible receptive eld and entity synonym set information in our constructed G_S. In the third part of Table 3, our model EnSyn with pre achieves signi cantly better performance than all other baselines. Comparing with the results of EnSyn in the rst part of Table 3, more accurate synonyms can be discovered by EnSyn with pre with the help of surface form, especially in datasets Wiki and NYT. The usefulness of surface form is also shown in the results of CharNgram, and in the comparison between SurfCon w/o pre and SurfCon. In addition, EnSyn with pre outperforms Concept Space and Planetoid because the concept space model simply concatenates the distributional embeddings with the rule-based features (the number of shared words), and Planetoid relies on one embedding vector for each entity which only uses the surface form embedding as input. Besides, noise in the raw text is not avoidable. On PubMed dataset, both SurfCon and EnSyn with pre do not bene t much from the pretrained embedding of surface form.

Next, we explore the effect of the attention layer in the within-level model by comparing EnSyn and EnSyn w/o Att, EnSyn-G and EnSyn-G w/o Att. In the second part of Table 3, we can nd that the models with attention layers achieve better performance than the models without attention layers. It shows the effectiveness of the self-attention mechanism in the entity synonym discovery scenario because the self-attention is able to automatically assign different relevant scores to the contexts. The high relevant contexts could obtain the high score, which bene ts the representation learning for entities and synonym sets. Furthermore, we analyze the effect of different between-level association models by comparing EnSyn and EnSyn-G. As the results shown in the second part of Table 3, the model EnSyn using the transformation-based association model outperforms EnSyn-G, which leverages the grouping-based association model. The reason could be that the transformation-based model utilizing trainable weight parameters to align two different embedding spaces has a smaller impact on the characterization of the structure of entity-level and set-level networks. Besides, the transformation-based model can search from larger feature space to nd the optimal transformation, compared with the direct minimization for the distance between linked entities in the grouping-based model.

4.4 Study of EnSyn (RQ4)

In this section, we investigate the impact of different parameters on the performance from several aspects, such as the number of layers, the message aggregation operators, and the number of k-nearest entity mention, etc.

4.4.1 Effect of Layer Numbers

We vary the number of GNN layers in our model to evaluate whether EnSyn can bene t from multiple embedding propagation layers. EnSyn-K indicates the model with K embedding propagation layers. Figure 6 shows that the performance can be enhanced with the number of layers increasing. Speci cally, EnSyn-1, which only considers the

TABLE 4: Effect of different message aggregation operators.

	Wiki		NYT		PubMed	
Measures	Hits@5	NDCG@5	Hits@5	NDCG@5	Hits@5	NDCG@5
EnSyn-GCN	0.3567	0.2973	0.2632	0.2108	0.3972	0.3358
EnSyn-GraphSage	0.3715	0.3046	0.2747	0.2142	0.4163	0.3427
EnSyn-GIN	0.3965	0.3125	0.2834	0.2381	0.4147	0.3484
EnSyn-1 w/o Att	0.4179	0.3181	0.2891	0.2367	0.4237	0.3562
EnSyn-GAT	0.4272	0.3254	0.3024	0.2492	0.4328	0.3659
EnSyn-1	0.4304	0.3298	0.2997	0.2514	0.4345	0.3690

Fig. 6: Effect of embedding propagation layer numbers.

Fig. 7: Effect of k-nearest entity mention numbers.

rst-order neighbors, is inferior to all other variants. Again, we see it is necessary to model the high-order contextual information. Furthermore, EnSyn-4 does not improve much, comparing with EnSyn-3. This is due to the limitation of GNNs, which may introduce noise when propagating too many layers.

4.4.2 Effect of Message Aggregation Operators

To investigate the effect of different message aggregation operators, we replace the aggregation operator in our method with GCN [16], GraphSage [42], GIN [68], and GAT [50]. We set all models with one propagation layer, and use single-head attention for GAT. Table 4 summarizes the experimental results. First, we can observe that our model EnSyn-1 w/o Att consistently outperforms the rst three methods with different aggregation operators on three different datasets. We attribute the enhancement to the design of fusing pairs of connected entities into message aggregation instead of a single entity. The fusion can encode the af nity between connected entities and make the message containing more information about the connection. Then, due to the advantages of the self-attention mechanism, our model EnSyn-1 and GAT achieve better performance than the variants EnSyn-1 w/o Att , EnSyn-GCN , EnSyn-GraphSage, and GIN , which neglect the important weights of different contexts, and would harm the effectiveness of aggregation. Further, the performance of EnSyn-1 is on par with that of GAT because of the adopted similar self-attention mechanism. We notice that EnSyn-1 slightly outperforms GAT . The reason could be the design of fusing pairs of connected entities into the message aggregation.

Fig. 8: Results of different methods on dataset Wiki and NYT when varying the dimension of embeddings.

4.4.3 Effect of k-Nearest Entity Mention Numbers

We vary the number of k-nearest entity mentions in our model to evaluate the effect of k. Figure 7 shows that the model with k=5 performs the best. Setting k=5 is reasonable because a small k (k=1) is insuf cient to collect semantically related neighbors. However, a large k (k = 10 or 15) will introduce noisy contexts for entities. Hence, we set k=5 in our experiments.

4.4.4 Effect of Dimension of Embeddings

The dimension of embedding is an essential factor for representation learning, which could directly affect the performance of machine learning models. To investigate the effect of dimension of embedding, we xed all other hyper-parameters and changed the dimension of embedding in $\{50, 75, 100, 125, 150\}$. Figure 8 shows how the dimension of embedding in uences the performance of different ESD methods on Wiki and NYT datasets. We see that our EnSyn is consistently better than other baselines, and the model

TABLE 5: Case studies on the Wiki and NYT dataset. Bold strings with underline are correct synonyms or synonym sets.

Query	Federal Government		Geoscience	
Synonym Set	US Federal Government U.S. Federal Government American Government		Earth Science Earth Sciences Geological Science	
Method	EnSyn-restricted	EnSyn	EnSyn-restricted	EnSyn
Ranked Output	U.S	**All entities in Set**	Science	Science
	Government	U.S.A	Department	**All entities in Set**
	US Federal Government	U.S.	College	Earth
	California	Government	**Earth Science**	Planet Earth
	White House	President	Earth	Natural Resouce

Query	Forex		WW2	
Synonym Set	Foreign Exchange Market FX Currency Market		World War II The Second World War WWII	
Method	EnSyn-restricted	EnSyn	EnSyn-restricted	EnSyn
Ranked Output	Central Bank	Exchange-traded Fund	**WWII**	**All entities in Set**
	Exchange-traded Fund	**All entities in Set**	WWI	World Wars
	Money	US Dollar	**World War II**	WWI
	Currency Market	Forex Dealer	Allies	Army
	FX	Money	History	Country

Fig. 9: Results of our proposed method on dataset Wiki and NYT when varying the proportion of synonym sets in set-level network.

would lead to under tting and be hard to converge if the dimension is not large enough. In addition, the results also show the stability of our model when varying the dimension.

4.4.5 Effect of Proportion of Synonym Sets in Set-level Network

In the construction of the two-level network, we leverage synonym sets to form the set-level network G_s to provide the set-level contextual information. It is crucial to explore the effect of the proportion of synonym sets explicitly in the network G_s. We randomly sample 10%, 30%, 50%, 70%, 90% of the synonym sets from Wiki and NYT datasets to construct the network G_s, and compare the performance of EnSyn using different two-level networks with different set-level networks. Figure 9 shows the results of Hits@5 and MRR of EnSyn when varying the proportion of synonym sets in G_s. First, as expected, EnSyn has better performance with the growth of the proportion of synonym sets because a small proportion is insuf cient to offer the set-level contextual information, and the constructed G_s still suffers from the restricted receptive eld. Instead, encoding the network G_s with a large proportion of synonym sets can effectively capture the set-level contextual information for the next step on updating representations for entities and synonym sets from the exible receptive eld.

4.5 Case Studies

Case studies are presented in Table 5. We output the ranked results of EnSyn and EnSyn-restricted with a given query entity. EnSyn-restricted only leverages the entity-level occurrence network G_e. Hence, it returns a list of entities in G_e as the left part shown in “Ranked Output”. EnSyn takes the synonym set information into consideration and encodes the two-level network, and it will output the list of nodes in G_s as the retrieval result as the right part shown in “Ranked Output”. In particular, we denote the correct output as “All entities in Set” for EnSyn. We can nd that EnSyn can correctly return all synonyms in the set to the query, and rank out more meaningful synonyms, than EnSyn-restricted . For example, EnSyn returns “Earth” and “Planet Earth” to query “Geoscience”, rather than giving the irrelevant but frequently co-occurred entities like “Department” and “College”. We see that set-aware ESD has better performance than those ignoring the entity synonym set.

5 Conclusions and Future Work

In this paper, we introduced a novel model for entity synonym discovery in the text corpus. First, we proposed to fuse the synonym set information into the network representation learning framework because the synonym set contains more comprehensive information about the referenced real-world entity than single entity and is able to be the context for other entities to enhance the expressiveness of the learned embedding. Second, due to the restricted receptive eld of entities and the desire to mitigate the in uence of noise from the synonym sets, we introduced a exible receptive eld to capture contextual information in the modeling process in order to select and balance the contextual information from the entity-level and set-level contexts. We constructed a two-level network to describe the connections between entities and entity synonym sets and to organize the entity-level and set-level contextual information. To encode the two-level network, we designed a within-level model based on graph neural networks to encode the high-order context in each level and a between-level association model to achieve the representation learning from the exible receptive eld by bidirectional propagating the contextual information. Our experiments on real-world datasets demonstrated that our approach achieved superior results compared with other state-of-the-art methods. In future work, we are going to expand the exploration and discussion about the utilization of attribute information of entities to enhance the performance of entity synonym discovery. We also will investigate how to aggregate the synonym set if the attribute is given.

ACKNOWLEDGMENTS

This work was supported by the King Abdullah University of Science and Technology (KAUST), Saudi Arabia.

REFERENCES

[1] F. Wu and D. S. Weld, “Open information extraction using wikipedia,” in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 118–127.
[2] G. Zhou, Y. Liu, F. Liu, D. Zeng, and J. Zhao, “Improving question retrieval in community question answering using world knowledge,” in *Twenty-Third International Joint Conference on Arti cial Intelligence*, 2013.
[3] T. Cheng, H. W. Lauw, and S. Paparizos, “Entity synonyms for structured web search,” *IEEE transactions on knowledge and data engineering*, vol. 24, no. 10, pp. 1862–1875, 2011.
[4] E. M. Voorhees, “Query expansion using lexical-semantic relations,” in *SIGIR?94*. Springer, 1994, pp. 61–69.
[5] S. Chaudhuri, V. Ganti, and D. Xin, “Exploiting web search to generate synonyms for entities,” in *Proceedings of the 18th international conference on World wide web*, 2009, pp. 151–160.
[6] N. Nakashole, G. Weikum, and F. Suchanek, “Patty: a taxonomy of relational patterns with semantic types,” in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 2012, pp. 1135–1145.
[7] M. Qu, X. Ren, and J. Han, “Automatic synonym discovery with knowledge bases,” in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 997–1005.
[8] M. Baroni and S. Bisi, “Using cooccurrence statistics and the web to discover synonyms in a technical language.” in *LREC*, 2004.
[9] P. Pantel, E. Crestan, A. Borkovsky, A.-M. Popescu, and V. Vyas, “Web-scale distributional similarity and entity set expansion,” in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*. Association for Computational Linguistics, 2009, pp. 938–947.
[10] C. Wang, L. Cao, and B. Zhou, “Medical synonym extraction with concept space models,” in *Twenty-Fourth International Joint Conference on Arti cial Intelligence*, 2015.
[11] J. Shen, R. Lyu, X. Ren, M. Vanni, B. Sadler, and J. Han, “Mining entity synonyms with ef cient neural set generation,” in *Proceedings of the AAAI Conference on Arti cial Intelligence*, vol. 33, 2019, pp. 249–256.
[12] Z. Wang, X. Yue, S. Moosavinasab, Y. Huang, S. Lin, and H. Sun, “Surfcon: Synonym discovery on privacy-aware clinical data,” in *SIGKDD*. ACM, 2019, pp. 1578–1586.
[13] D. Zhu, P. Cui, Z. Zhang, J. Pei, and W. Zhu, “High-order proximity preserved embedding for dynamic networks,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 11, pp. 2134–2144, 2018.
[14] S. Cao, W. Lu, and Q. Xu, “Grarep: Learning graph representations with global structural information,” in *Proceedings of the 24th ACM international on conference on information and knowledge management*, 2015, pp. 891–900.
[15] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes, “Improving ef ciency and accuracy in multilingual entity extraction,” in *Proceedings of the 9th International Conference on Semantic Systems*, 2013, pp. 121–124.
[16] T. N. Kipf and M. Welling, “Semi-supervised classi cation with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.
[17] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, “Bpr: Bayesian personalized ranking from implicit feedback,” in *Proceedings of the twenty- fth conference on uncertainty in arti cial intelligence*. AUAI Press, 2009, pp. 452–461.
[18] X. Ren and T. Cheng, “Synonym discovery for structured entities on heterogeneous graphs,” in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 443–453.
[19] K. Chakrabarti, S. Chaudhuri, T. Cheng, and D. Xin, “A framework for robust discovery of entity synonyms,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 1384–1392.
[20] Y. Li, B.-J. P. Hsu, C. Zhai, and K. Wang, “Mining entity attribute synonyms via compact clustering,” in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 867–872.
[21] X. Wei, F. Peng, H. Tseng, Y. Lu, and B. Dumoulin, “Context sensitive synonym discovery for web search queries,” in *Proceedings of the 18th ACM conference on Information and knowledge management*, 2009, pp. 1585–1588.
[22] H. Fei, S. Tan, and P. Li, “Hierarchical multi-task word embedding learning for synonym prediction,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 834–842.
[23] C. Zhang, Y. Li, N. Du, W. Fan, and P. Yu, “Entity synonyms discovery via multipiece bilateral context matching,” in *IJCAI*, 2020.
[24] M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin, “Entity disambiguation for knowledge base population,” in *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 277–285.
[25] W. Shen, J. Wang, and J. Han, “Entity linking with a knowledge base: Issues, techniques, and solutions,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 2, pp. 443–460, 2014.
[26] G. Demartini, D. E. Difallah, and P. Cudr é-Mauroux, “Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking,” in *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 469–478.
[27] D. Milne and I. H. Witten, “Learning to link with wikipedia,” in *Proceedings of the 17th ACM conference on Information and knowledge management*, 2008, pp. 509–518.
[28] Z. Chen and H. Ji, “Collaborative ranking: A case study on entity linking,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2011, pp. 771–781.
[29] Y. Cao, L. Huang, H. Ji, X. Chen, and J. Li, “Bridge text and knowledge by learning multi-prototype entity mention embedding,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1623–1633.
[30] N. Gupta, S. Singh, and D. Roth, “Entity linking via joint encoding of types, descriptions, and context,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2681–2690.
[31] O.-E. Ganea, M. Ganea, A. Lucchi, C. Eickhoff, and T. Hofmann, “Probabilistic bag-of-hyperlinks model for entity linking,” in *Proceedings of the 25th International Conference on World Wide Web*, 2016, pp. 927–938.
[32] C. Ran, W. Shen, and J. Wang, “An attention factor graph model for tweet entity linking,” in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 1135–1144.
[33] O.-E. Ganea and T. Hofmann, “Deep joint entity disambiguation with local neural attention,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2619–2629.
[34] X. Han, L. Sun, and J. Zhao, “Collective entity linking in web text: a graph-based method,” in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 2011, pp. 765–774.
[35] S. Zwicklbauer, C. Seifert, and M. Granitzer, “Robust and collective entity disambiguation through semantic embeddings,” in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 425–434.
[36] T. H. Nguyen, N. R. Fauceglia, M. R. Muro, O. Hassanzadeh, A. Gliozzo, and M. Sadoghi, “Joint learning of local and global

features for entity linking via neural networks," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2310–2320.

[37] Z. Fang, Y. Cao, Q. Li, D. Zhang, Z. Zhang, and Y. Liu, "Joint entity linking with deep reinforcement learning," in *The World Wide Web Conference*, 2019, pp. 438–447.

[38] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *arXiv preprint arXiv:1901.00596*, 2019.

[39] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *arXiv preprint arXiv:1812.08434*, 2018.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classi -cation with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[41] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[42] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[43] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *SIGKDD*. ACM, 2018, pp. 974–983.

[44] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *NeurIPS*, 2018, pp. 4800–4810.

[45] H. Gao and S. Ji, "Graph u-nets," in *International Conference on Machine Learning*, 2019, pp. 2083–2092.

[46] C. Zhang, Q. Li, and D. Song, "Aspect-based sentiment classi- cation with aspect-speci c graph convolutional networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 4560–4570.

[47] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," in *ICLR*, 2014.

[48] J. Chen, T. Ma, and C. Xiao, "Fastgcn: fast learning with graph convolutional networks via importance sampling," *arXiv preprint arXiv:1801.10247*, 2018.

[49] W. Huang, T. Zhang, Y. Rong, and J. Huang, "Adaptive sampling towards fast graph representation learning," in *Advances in neural information processing systems*, 2018, pp. 4558–4567.

[50] P. Velickovi ć, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *International Conference on Learning Representations*, 2018.

[51] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5453–5462.

[52] M. Kampffmeyer, Y. Chen, X. Liang, H. Wang, Y. Zhang, and E. P. Xing, "Rethinking knowledge graph propagation for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 487–11 496.

[53] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The World Wide Web Conference*, 2019, pp. 2022–2032.

[54] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-second AAAI conference on arti cial intelligence*, 2018.

[55] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European Semantic Web Conference*. Springer, 2018, pp. 593–607.

[56] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.

[57] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Recti er nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1, 2013, p. 3.

[58] X. Glorot and Y. Bengio, "Understanding the dif culty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on arti cial intelligence and statistics*, 2010, pp. 249–256.

[59] C.-H. Wei, H.-Y. Kao, and Z. Lu, "Pubtator: a web-based text mining tool for assisting biocuration," *Nucleic acids research*, vol. 41, no. W1, pp. W518–W522, 2013.

[60] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 1247–1250.

[61] O. Bodenreider, "The uni ed medical language system (umls): integrating biomedical terminology," *Nucleic acids research*, vol. 32, no. suppl_1, pp. D267–D270, 2004.

[62] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[63] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in neural information processing systems*, 2014, pp. 2177–2185.

[64] 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*, 2016, pp. 855–864.

[65] K. Hashimoto, C. Xiong, Y. Tsuruoka, and R. Socher, "A joint many-task model: Growing a neural network for multiple nlp tasks," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 1923–1933.

[66] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, 2016, pp. 40–48.

[67] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[68] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *International Conference on Learning Representations*, 2019.

Shichao Pei Received the BEc. degree in the Software Engineering from Xi'an Jiaotong University in 2016, the Master in Computer Science from King Abdullah University of Science and Technology (KAUST) in 2017, respectively. He is currently working toward the Ph.D. degree in the the Division of Computer, Electrical, and Mathematical Sciences and Engineering (CEMSE), King Abdullah University of Science and Technology. He works in the elds of data mining, information retrieval and recommender systems. His work has appeared in international journals and conferences.

Lu Yu received B.E. degree in Computer Science from University of Electronic Science and Technology of China, Master degree from Hangzhou Normal University. He is currently a Ph.D. candidate in College of Computer Science, King Abdullah University of Science and Technology. His research interests include recommender systems, graph representation learning, and information retrieval. He has over 20 publications appeared in several top conferences such as AAAI, WWW, IJCAI, and journals including INS, KBS. He has been served as a reviewer for journals INS, Plos One, and top conferences including AAAI, IJCAI, ICML, CIKM etc.

Xiangliang Zhang is currently an Associate Professor and directs the Machine Intelligence and Knowledge Engineering (MINE) Laboratory at the Division of Computer, Electrical, and Mathematical Sciences and Engineering, King Abdullah University of Science and Technology (KAUST), Saudi Arabia. She was a European ERCIM Research Fellow at the Norwegian University of Science and Technology, Norway, in 2010. She received the Ph.D. degree in computer science from INRIA-University Paris-Sud, France, in July 2010. She has authored or co-authored over 130 refereed papers in various journals and conferences. Her current research interests and experiences include machine learning, and data mining.