Ain Shams Engineering Journal 15 (2024) 102263

Contents lists available at ScienceDirect

Ain Shams Engineering Journal

journal homepage: www.sciencedirect.com

Knowledge graph-based recommendation system enhanced by neural collaborative filtering and knowledge graph embedding

Zeinab Shokrzadeh [a], Mohammad-Reza Feizi-Derakhshi [b,*], Mohammad-Ali Balafar [b], Jamshid Bagherzadeh Mohasefi [c]

[a] Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran
[b] Department of Computer Engineering, University of Tabriz, Tabriz, Iran
[c] Department of Computer Engineering, Urmia University, Urmia, Iran

A R T I C L E I N F O

Article history:
Received 9 December 2022
Revised 30 January 2023
Accepted 25 March 2023
Available online 21 April 2023

Keywords:
Social tagging systems
Recommender systems
Collaborative filtering
Knowledge graph
Neural collaborative filtering
Knowledge graph representation learning

A B S T R A C T

Recommendation systems are an important and undeniable part of modern systems and applications. Recommending items and users to the users that are likely to buy or interact with them is a modern solution for AI-based applications. In this article, a novel architecture is used with the utilization of pre-trained knowledge graph embeddings of different approaches. The proposed architecture consists of several stages that have various advantages. In the first step of the proposed method, a knowledge graph from data is created, since multi-hop neighbors in this graph address the ambiguity and redundancy problems. Then knowledge graph representation learning techniques are used to learn low-dimensional vector representations for knowledge graph components. In the following a neural collaborative filtering framework is used which benefits from no extra weights on layers. It is only dependent on matrix operations. Learning over these operations uses the pre-trained embeddings, and fine-tune them. Evaluation metrics show that the proposed method is superior in over other state-of-the-art approaches. According to the experimental results, the criteria of recall, precision, and F1-score have been improved, on average by 3.87%, 2.42%, and 6.05%, respectively.

1. Introduction

With the advancement of internet and communication technologies, people are facing increase in data. Obtaining useful information from these big data is one of the important challenges. To address this challenge, recommendation systems (RSs) are one of the most useful tools [1]. To have highly accurate recommendation systems is an important challenge that has led researchers to conduct extensive studies in this area. So far, many solutions have been proposed to provide recommendation systems that can follow the recommendations according to users' interests. This path continues and requires more research and efforts.

To create a personalized recommendation system, it is required to employ the history data of users' online behavior s such as ratings, clicks, tags, and comments, and model users' taste in items according to these historical interactions, which is also known as collaborative filtering [2]. After presenting the idea of collaborative filtering, various approaches have been introduced for this purpose.

The main element of recommendation systems is the recommendation algorithm, which are mostly grouped into collaborative filtering (CF), content based, and hybrid recommendation systems [3]. Recommender systems, based on CF, model user interests according to the similarity of users or items from the interaction data, while content-based systems utilizes content features of item. CF-based recommender systems have been widely used because they are effective for capturing user preferences and can be easily developed in multiple scenarios, without trying to extract features in content-based recommender systems [4,5]. However, CF-based recommendations suffer from data fragmentation and cold start problems [6]. To address these issues, hybrid recommender systems have been proposed to equate interaction-level similarity and content-level similarity [7]. In this process, various types of side information have been investigated, such as item attributes [2], item reviews, and social networks of users [8].

* Corresponding author.
E-mail addresses: Shokrzadeh82@yahoo.com (Z. Shokrzadeh), mfeizi@tabrizu.ac.ir (M.-R. Feizi-Derakhshi), balafarila@tabrizu.ac.ir (M.-A. Balafar), J.bagherzadeh@urmia.ac.ir (J. Bagherzadeh Mohasefi).

https://doi.org/10.1016/j.asej.2023.102263
2090-4479/

Recently, knowledge graphs have attracted increasing attention and become a popular means of collecting, displaying, and extracting structured knowledge that consists of various entities, their attributes, and their inter-relationships [9]. Knowledge graphs have launched by Google in 2012 for the first time [10]. Since then, various efforts have been proposed to enhance and apply such knowledge structure in various fields such as adaptive learning, multimedia e-learning, software development, social media analysis, multi-task feature learning, air traffic management, data warehouses, etc. Knowledge graphs have also been used in various recommendation systems such as travel attraction recommendations [11], personalized tweet and follower recommendations [12], differentiated fashion recommendations [13], mashup-based APIs recommendations [14], points of interest recommendations [15,16], group recommendation [17], etc.

In this paper, a knowledge graph-based recommendation system was proposed enriched with knowledge graph representation learning and neural collaborative filtering. The proposed recommendation system first presents data in the form of a knowledge graph which is constructed based on the user-user, user-tag, user-source, and tag-source relationships. These components engender multi-hop neighbors in this graph that can abolish the ambiguity and redundancy problems. Although there are a number of studies that have been conducted on representing data in form of knowledge graph to recommend items. However other works just used some part of data to construct their desired knowledge graph. Following this, the next step enriches the original knowledge graph using advanced knowledge graph embedding techniques (ConvE, ComplEx and TransE) to learn low-dimensional vector representations of knowledge graph elements. The final stage is a neural collaborative filtering framework that receives an advantage, because learning a neural network provides capability of performing classification tasks without additional weights. Our neural framework just relies on matrix operations, uses the pre-trained embeddings obtained from the knowledge graph, and fine-tunes them. The major contributions of this paper can be summarized as follows:

- To the best of our knowledge, this paper is the first attempt to completely consider all entities and their full connections to construct a knowledge graph. In other words, all types of relationships including user-user, user-tag, user-source and tag-source are observed in the created knowledge graph. The main advantage of this knowledge graph is that it presents an opportunity to alleviate the ambiguity and redundancy problems.
- A further advantage is that issues such as cold-start, ambiguity and redundancy can be addressed with the help of multi-hop neighbors in our constructed knowledge graph. Also, because of the similar neighborhood sub-graphs, synonymous tags can be semantically considered close to each other.
- The existence of user-user, user-tag, user-source, and tag-source interactions in our knowledge graph enables us to deal with the issue of cold start for a new tag or source by inferring the connections.
- Another significant benefit is to apply entity embedding learning in order to enrich the representations of entities. For this purpose, three different categories of existing knowledge graph embedding techniques (ConvE, ComplEx and TransE) were used.
- Further gain to consider is that this work presents a neural collaborative filtering framework that uses matrix operations and fine-tuning pre-trained embeddings of users, tags and sources to recommend various items. It means that embeddings generated by using knowledge graph embedding methods are injected as pre-trained vectors into the neural collaborative filtering network which are considered to be the inputs.
- Multiplying the embeddings of users and tags, and calculating the dot product of the output of multiplication and embedding of source in the neural network facilitates to enrich the representations users, tags and sources. Subsequently, our system is capable of recommending various items such as user, tag and source just by changing the input vectors. For example, to recommend tags, the embeddings of users and sources are fed into the matrix multiplication operation and then the dot product of the previous operation and tag vectors were done.

The structure of the article is stated as follows: Section 2 review the fundamental concepts of knowledge graph, knowledge graph embeddings and its methods and the related works. In section 3 the research method is explained in detail. Section 4 is experimental evaluation. Finally, in section 5, conclusions are descripted.

2. Background

In this section, the concepts construct in the background of the study are expressed. First, a brief explanation of knowledge graphs will be provided. Then, the knowledge graph embedding techniques employed in proposed solutions were mentioned.

In the real world, taking the form of a graph in a number of different scenarios have received remarkable attention [18]. For instance, graph representation serves as the underlying support for various crucial domains in the field of health and diseases like drug analysis and disease diagnosis [19]. Moreover, graph structure has become a significant strategy for many applications including data mining, natural language processing, and question-answering systems [20].

One of the particular and popular types of graph structure is a knowledge graph in which the node and edge respectively demonstrate the entity and relation between them. Since a knowledge graph is a structure used to represent diverse types of information in the form of different types of entities such as concepts, situations, objects, events, and portrays different types of relations, it is renowned as a semantic network [10]. In traditionally, a knowledge graph is represented by a triplet (s, p, o) where s, o indicates the subject and predicate entities accordingly, and p is the relation associating s and o. Additionally, labels and attributes as extra information can be assigned to entities and relations[21].

The goal of knowledge graph analysis is to provide a quantitative understanding of complex structures. Drawbacks of conventional approaches mainly include high computational cost and memory requirements associated with the high dimensions [22]. In fact, a group of methods have become a bottleneck in massive network analysis.

To prevail over the mentioned obstacles, a lot of research efforts have been devoted to learning low-dimensional vector representations for knowledge graph components. Specifically, knowledge graph embedding transforms entities and relations into a low-dimensional space in which the knowledge graph information is preserved [23]. Knowledge graph embedding techniques based on their solutions may comprise some functions including similarity function and loss function. The similarity function is responsible for estimating the identity of entities. In actual fact, the loss function looks over the quality of the vector representations. This function includes a scoring function that assigns a score to each graph triple [23].

Knowledge graph embedding methods can be classified into three groups: neural network, semantic matching and translational distance models [24]. Neural network techniques apply neural network models to obtain better embeddings of entities and relations. In the semantic matching methods, the scoring function is similarity-based that determines the plausibility of facts by matching the latent semantics of entities and relations [24]. On the other coins, translational distance models utilize distance-based scoring functions to determine the plausibility of a fact as the distance between the two entities [25]. In this study, ConvE, ComplEx, and TransE graph knowledge embedding approaches will be employed. Although there are many cutting-edge approaches such as DKN [26] and KBGAN [27], to name a few, these three models were hand-picked for two fundamental reasons. Firstly, these models cover three different categories of graph representation learning models. The selected methods belong to neural network, semantic machine and translational groups. Secondly, to remove any biases in the results because of using trendy approaches, these renowned techniques were employed. In other words, to imply that the innovation of our proposed method does not rely on the type or the innovations used in their suggested structures, these knowledge graph representation learning models were used. In the following paragraphs, these methods will be described in more detail.

ConvE [25] uses entity and node embedding training with 2D convolutional neural networks. This architecture is designed in a way to address knowledge graph embedding for different types of graphs. Embedding drop-out is one of the novelties of this method used to spread the features among different embedding values rather than putting everything under specific values and indices. Link prediction is one of the tasks that this approach used to train the embeddings, which is a very important task in various use cases. In case of recommendation systems, predicting the possibility of a link between different nodes is more like the recommendation itself. The score function of this technique is summarized in Eq. (1):

$$f_{convE} = f\left(vec\left(g\left(\left[\bar{e_s}; \bar{r_p}\right] * \Omega\right)\right)W\right)e_0 \tag{1}$$

where g is a non-linear activation function, $*$ is a linear convolution operator, vec shows 2D reshaping. Also, f is the logistic sigmoid function. e_s, e_o and r_p denotes the embeddings of the subject, predicate, and object. Finally, $\bar{e_s}$ and $\bar{r_p}$ are 2D reshaping of e_s and r_p, respectively.

ComplEx [28] uses complex valued embeddings to provide knowledge graph embedding. Using such values can handle binary relations in embedding vectors and symmetric and asymmetric relations can also be handled as well. With regard to other methods of knowledge graph embeddings, this approach is much simpler because it only uses dot-product operations which makes it a good candidate for fast knowledge graph embedding. The scoring function of this approach is defined as Eq. (2):

$$f_{ComplEx} = Re\left(r_p, e_s, \bar{e_o}\right) \tag{2}$$

here Re indicates taking the real part of a complex value.

TransE [29] is a translational distance model that uses translation-based transformation for embedding knowledge graph entities. This approach uses a better approach compared to other non-deep learning-based approaches. TransE uses a multi-relational approach to create embeddings. Multi-relational approach means taking into account different entity types and nodes with same regard to created unified embeddings. In this method, the score function uses L_1 or L_2 norm to measure the similarity between the embedding of the subject. In Eq. (3) shown the scoring function. In this equation, the number of entities is indicated by n.

$$f_{TransE} = -||e_s + r_p - e_o||n \tag{3}$$

2.1. Related works

Today, anyone is in the ocean of information. Finding useful information from big data is very difficult and time consuming. Recommendation systems are a good solution to find useful information according to users' interests. Usually, recommendation system is a collection of algorithms that discover data patterns from the accessible dataset by learning and computing preferences of user [18]. Finally, based on the correlation between the interests and needs, the system provides the most relevant and useful results [18].

A knowledge-based recommendation system offers items to the user based on scope knowledge about how the items satisfy the user preferences [30].

Knowledge graphs have shown great potential in supporting recommendation systems [31], [32,33,34,35,36,44]. The main idea of using knowledge graphs in thus knowledge-intensive applications is how to transform their heterogeneous and semi-structured data into user-item relations, and extract useful features from the KG [37]. Based on how these works employ the KG information, they are grouped into three categories: embedding-based methods, path-based methods, and unified methods [37]. The embedding-based methods generally use the information from the KG directly to enrich the representation of items or users [37]. As mentioned in the background, in order to exploit the KG information, knowledge graph embedding(KGE) algorithms need to be used to encode the KG into low-rank embedding. All KGE algorithms can be divided into two classes [38]: translation distance methods, such as TransE [29], TransH [39], TransR [40], TransD [41], etc., and semantic matching methods, such as DistMult [42].

In path-based approaches create a user-item graph and use the entity connectivity patterns in the graph to make recommendations [43].

In [26] Wang et al. proposed a news recommendation system called DKN. This method models the news by combining the textual embedding of sentences learned via Kim CNN [45] and embedding of entities in news content at the knowledge-level by TransD [25].

Huang et al. [46] represented a KSR framework for sequential recommendations. In KSR, they used a GRU network with a knowledge-enhanced key-value memory network (KV-MN) to model comprehensive user preference from sequential interaction [49]. The network of GRU utilizes sequential user preferences as input, while the KV-MN module uses knowledge base information that learned with TransE to model the user's feature-level preference [46].

In [47] Zhang et al. represented CFKG. In this method, they create a user-item knowledge graph. In this graph, behaviors of user like purchase and mention are regarded as one relation type between entities, and multiple types of item side information as review, brand, category, etc. are contained [41]. This model, defines a function to measure the distance between two items by learning the embedding of items and relationships in the graph.

Wang et al. [48] proposed SHINE, which presented the celebrity recommendation as a task of predicting sentiment links between entities in a graph [10].

Yang et al. [49] represented the movie recommender system. They used a model based on GAN called KTGAN. In the first step, KTGAN incorporating the Metapath2Vec model [50] on the KG of movies by learns the knowledge embedding vector for movie vector, and the tag vector embedding by the Word2Vec model [51] on attributes of movie.

Wang et al. [63] to aggregate information from knowledge graph employed graph attention networks. They used high order connectivity in knowledge graph.

Most embedding-based methods construct KGs with various kinds of side information of item to enrich the item representation, and such information can be used to more accurately model the user representation [10]. Some models construct user-item graphs by introducing users to the graph, which can directly model the user tastes [10]. The core of embedding-based methods is entity embedding, and some papers modify embedding with GAN [49] or BEM [52] for better recommendation. The methods of Embedding-based use the information in the graph structure intrinsically.

Recently, the producing of recommendations with knowledge graphs as side information has attracted remarkable attention. In fact, with accurate presentation the information such as profiles of users, tags, items and relationships of them, helps extract useful knowledge regarding the target user's interests.

In addition to the advantage of interpretability and accuracy, another advantage of recommendations based on KG is that this kind of side information can be naturally incorporated into recommendation systems for various applications. To demonstrate the effectiveness of KG as side information, recommendation systems based on KG have been evaluated on datasets under different scripts [10]. These results are shown in Table 1.

Knowledge graphs provide additional information to solve problems when faced with collaborative and content-based filtering methods. On the other hand, it is possible to increase the accuracy and variety of recommendations in the system with the help of the semantic relations of the KG [54].

Usually, the combination of knowledge graphs with collaborative filtering is one of the most common methods. In this article, using the knowledge graph, the embedding technique, the neural networks and collaborative filtering engender a solution to increase the accuracy of the recommendation system has been presented.

Studying and reviewing the research done in the field of recommender systems, some of which were briefly reviewed in this section, it is concluded that most of these systems suffer from insufficient accuracy in providing recommendations. This low accuracy is caused by a misdiagnosis of communication between users, items and sources. In other words, if users' interests can be correctly identified and explicit and implicit connections between users and tags and sources can be recognized correctly, recommendations with higher accuracy will definitely be obtained. Reviewing the methods and solutions presented in various articles, the researchers found this weakness in traditional and modern solutions and decided to look for a suitable solution. Therefore, they tried to use the strengths and weaknesses of previous studies and presented an efficient method to provide high accuracy in suggestions. Table 2 shows a number of related articles studied.

Table 1
A collection of different datasets in different application scenarios [13].

Scenario	Dataset
Movie	MovieLens-1 M
	MovieLens-20 M
	DoubanMovie
	MovieLens-100 K
Book	DoubanBook
	Amazon-Book
	DBbook2014
	Book-Crossing
	IntentBooks
News	Bing-News
Product	Alibaba Taobao
	Amazon Product data
POI	Yelp challenge
	CEM
	Dianping-Food
Music	Last.Fm
	KKBox
Social Platform	Weibo
	MeetUP
	DBLp

Table 2
Some of the related articles studied.

Ref	KG	Embedding-based method	Type of recommend
[11]	√	√	Item recommender
[35]	√	√	service recommendation
[37]	√	√	service recommendation
[26]	√	√	News recommendation
[46]	√	√	Sequential recommender
[48]	√	√	Celebrity recommender
[49]	√	√	Movie recommender
[63]	√	√	Item recommender

3. Research method

The aim of this research is to design a neural collaborative filtering model that is able to recommend multi-items such as user, tags and sources. The overall structure of the proposed method is shown in Fig. 1. The steps taken for this method includes:

1- constructing knowledge graph from dataset
2- pre-trained knowledge graph embeddings
3- collaborative filtering
4- classification

In the first step, our solution presents information in the form of a knowledge graph. The main advantage of this representation of data is to help address the ambiguity and redundancy problems. In the next step, the model takes advantage of the knowledge graph representation learning technique to learn representations of graph entities. Subsequently, a neural collaborative-filtering network were presented to fine-tune pre-trained vectors for a recommendation task. The final step includes a neural collaborative-filtering framework that trains a neural network for a classification task without additional weights. In another scene, vector representations of items generated from knowledge graph embedding methods are used as input features of the neural collaborative network. In order to capture items' representations, our method is comprised of two different neural networks: one for node representation, and the other one for learning a shared representation between items. Then, the task of recommendation is modeled as a classification problem. However, as far as the proposed structure includes four different stages, these phases were divided into two groups and their procedures were described in detail in the following two subsections. The first subsection provides information about constructing a knowledge graph and knowledge graph embedding. In contrast, the characteristic of the neural network and the classification procedure are described in the second subsection.

3.1. Constructing knowledge graph and learning embeddings

Modern deep-learning architectures require a self-supervised training before fine-tuning the dataset. In order to perform this task for a recommendation system, a knowledge graph-based node embedding architecture was used to self-supervise the model on this data. The data is first transformed into a knowledge graph structure containing three different types of nodes: Users, Tags, and Sources. Relations between these entities are created by using the dataset relations which, are as follows:

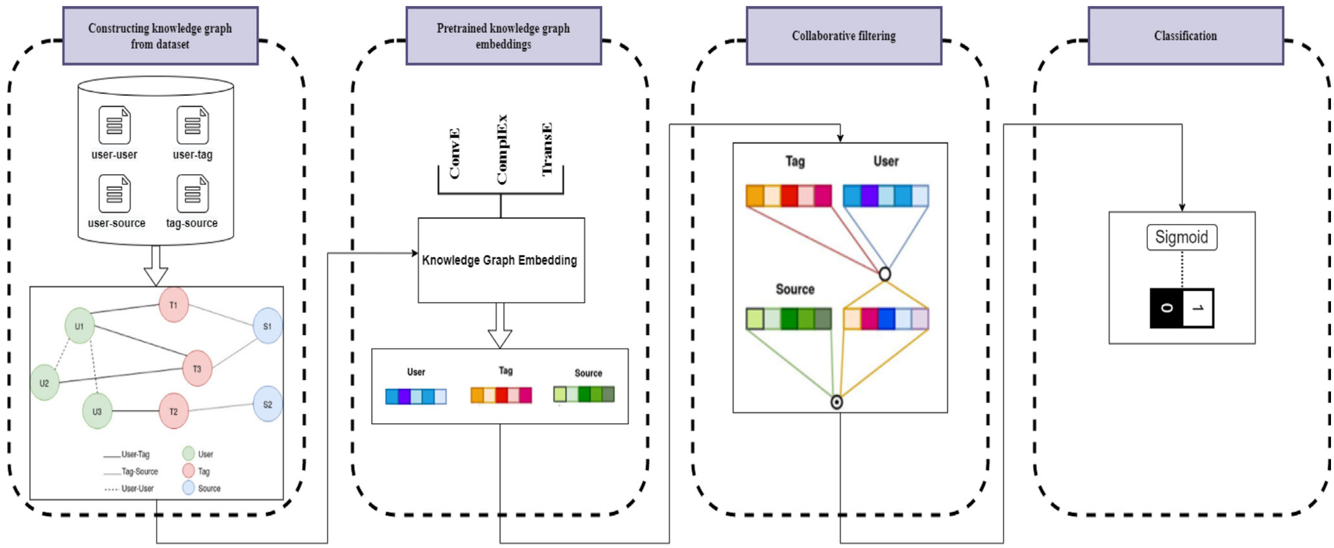


Fig. 1. Proposed architecture for source recommendation.

-User-User Relation: User A follows User B or vice versa.

-User-Tag-Source Relation: User A used Tag T as Keyword for Source S.

Putting all of these relations in a single graph shows a unified version of data in form of a graph with these relations.

Organizing data as a user-tag-source knowledge graph provides more contextual semantic information, which can play a significant role in dealing with the polysemy problem. Moreover, the multi-hop neighbors in this knowledge graph not only alleviate cold start, but also, they help distinguish between ambiguous entities. To show how the created knowledge graph is capable of solving these issues, consider a record in Delicious dataset that User U_1 uses Tag ″apple″, which is a polysemous tag. The relation between Tag-Source in the knowledge graph contributes to understand the differences between concepts. Sources with titles like "iPad", "iPhone", "Mac", "Installing", "Ecamm" and "Android" indicate that tag "Apple" is a technology company. Therefore, our system cannot confuse "Apple", the brand, with "apple" fruit. Moreover, due to the fact that, a part of graph relies on the user-user interaction that comforts to make recommendations for new users who just follow their friends and there are no histories about their tagging behaviors. Additionally, the existence of user-user, user-tag and tag-source interactions in our knowledge graph enables to deal with the issue of cold start for new tag or source by inferring the connections.

After constructing our desirable knowledge graph, the next step of our proposed method is to map the component of this graph into low-dimensional vectors. The contemporary achievements in the knowledge graph to represent a knowledge graph's entities and relations as continuous vector spaces are knowledge graph embeddings. The key property of knowledge graph embeddings is to capture the semantics and inherent structures of entities and relations [37]. To achieve the goal, three different knowledge graph embedding techniques were used, including ConvE [48], ComplEx [27], and TransE [28]. The details of how these methods work were described in background section.

The key idea behind the second phase of Fig. 1 is to leverage the productive facts in the knowledge graph to enhance the representations of users, tags and sources. Also, these embeddings allow to find the top similar tags, users or sources. That means that this part of our model guarantees that entities of the same type to stay close to each other in the embedding space.

3.2. Neural collaborative filtering and classification

The next step of the model is a neural collaborative filtering framework that treats recommending as a classification task. The generated embeddings obtained from previous step work as weights for this neural network. That means that instead of training the neural network with randomly initialized weights, our proposed approach uses the pre-trained vectors obtained from knowledge graph embeddings.

Apart from pre-trained and extracting vectors from the pre-trained model, it is necessary to fine-tune it. For fine-tuning, a neural collaborative filtering network is used by only focusing on these vectors and fine-tuning them. In order to avoid extra parameters and focus solely on the vectors that were pre-trained from knowledge graph embedding approaches, the architecture, shown in Fig. 2, was employed. In the input layer, the user, tag and source are one-hot encoded. We utilized just the identity of a user, tag and source as input features and convert them to binarized sparse vectors with one-hot encoding. Employing this kind of representation for inputs help our model easily adapt to cope with the cold-start problem by using content features to represent users, tags and sources.

The embedding layer is above the input layer. The embedding layers include user, tag and source vectors obtained from the knowledge graph embedding technique. In the context of the latent factor model, the obtained user (item) embedding can be inferred as the latent vector. Afterwards, in this article, there is a neural architecture that is termed as neural collaborative filtering layer. This framework maps the latent vectors to recommend a source. The first layer of this neural network is a multiplication layer that executes element-wise multiplication on the user and tag embeddings. This layer recognizes the latent structures of user–tag interactions. Subsequently, the next layer is the dot product of user-tag matrix multiplication and source vector which, learns the interaction between user, tag and source latent features. The result of this layer connects to the last layer where the output needs to be constrained in the range of [0, 1]. In order to easily achieve this goal, the sigmoid function was used as the activation function for the output layer.

As mentioned, the immediately following step is a binary classification which is the main task that is why embeddings are finedtuned. This classification is used to ensure that each User-Tag-

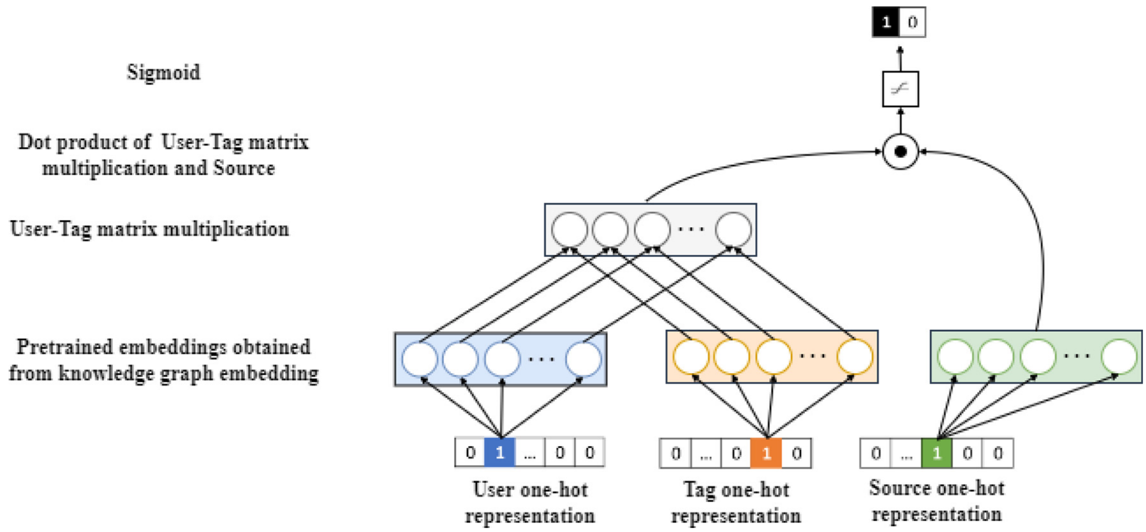


Fig. 2. Architecture of neural network.

Source vector is fine-tuned in a way that: if a user tagged source S with tag T then the result is 1 and 0 otherwise. According to this analogy, the pre-trained vectors are fine-tuned and can be put into a vector search database.

Training a binary classification on recommendation-based data is also one of the important novelties of this work that is accomplished by creating negative data. In order to create data for each of the three recommendation tasks (User recommendation, Tag recommendation, and Source recommendation) syntactic data creation was used by randomly sampling (uniform distribution) over users, tags, and resources. For example, in the case of user recommendation, there is a user-tag-source triplet (U, T, S) that has happened before. User U has tagged source S with tag T. It is accepted as positive sample, and negative samples were created by fixing S and T and changing U with other users that are not in the dataset. The same applies to tag and source recommendations.

For more clarification, as far as, operations used in neural network architecture as mathematical expressions were presented, the following equations have presented. One-hot encoded vectors regarding a triplet, user, tag, and source are created from their respective ids. Some examples are shown in Eq. (4).

$$(u_{id}, t_{id}, s_{id}) \xrightarrow{one-hotencoder} \begin{array}{l} u_{oh}[0010\cdots 0]_n \\ t_{oh}[0000\cdots 1]_m \\ s_{oh}[0001\cdots 0]_k \end{array} \tag{4}$$

where u_{oh}, t_{oh}, s_{oh} are one-hot vectors user, tag, and source. In this Equation n, m, k show the number of users, tags, and sources, respectively. Each one-hot vector is the input of the proposed architecture. In this architecture, three different dense layers are transforming one-hot vectors into embeddings which, for all of the triplet items are in the same size d. Eq. (5) show these embeddings that are also pre-trained from knowledge graph embedding.

$$u_{oh} \rightarrow \overrightarrow{u} \rightarrow [u_{e_1}, \cdots, u_{e_d}]$$

$$t_{oh} \rightarrow \overrightarrow{t} \rightarrow [t_{e_1}, \cdots, t_{e_d}]$$

$$s_{oh} \rightarrow \overrightarrow{s} \rightarrow [s_{e_1}, \cdots, s_{e_d}] \tag{5}$$

where $\overrightarrow{u}$, $\overrightarrow{t}$, $\overrightarrow{s}$ are embedding vectors and $u_{e_i}, t_{e_i} and s_{e_i}$ vector values.. Vector multiplication between elements of user and tag vectors are applied that Eq. (6) shows it.

$$\overrightarrow{u} \times \overrightarrow{t} = [u_{e_1} \times t_{e_1}, u_{e_2} \times t_{e_2}, \cdots, u_{e_d} \times t_{e_d}] \tag{6}$$

Dot-product multiplication ensures that the resulting vector from previous equation is combined with the source. This operation ensures the presence of user, tag and source embeddings in our collaborative filtering model which engenders an accurate recommendation system. Eq. (7) shows it. From this equation, it is clear that we combined all the elements of embedding vectors one by one.

$$\left(\overrightarrow{u} \times \overrightarrow{t}\right) \cdot \overrightarrow{s} = \sum_{i=0}^{d} u_{ei} t_{ei} s_{ei} \tag{7}$$

A sigmoid activation is applied to make sure the range of output is in [0, 1]. Eq. (8) shows whole neural network matrix operations which is indicated by nn.

$$nn\left(\overrightarrow{u}, \overrightarrow{t}, \overrightarrow{s}\right) = sigmoid\left(\sum_{i=0}^{d} u_{ei} t_{ei} s_{ei}\right), [0, 1] \tag{8}$$

4. Experimental evaluation

In this section, an overview of the dataset, which applied in the proposed method, will be provided. The dataset used in this research is real world. This dataset is widely experimented by social recommendation systems. In the following, the number of baseline algorithms will be introduced to compare with the proposed approach against them. Next, the used parameter settings are described. Finally, the metrics hired to evaluate our proposed algorithm will be discussed.

4.1. Dataset

One of the most important parts of any recommendation system is data collection. If it was done in the accurate and regular way, data analysis can be done with great speed and accuracy [55]. In this system, among the published valid datasets, the widely used Hetrec2011-Delicious-2 k dataset such as [5657] was employed in the experiments, which contains 53,388 tags, 1867 users, and 69,226 sources collected from Delicious.com and pub-

Table 3
User contacts.

UserID	UserID
22	102
1432	54
298	8
. . ...	. . ...
. . ...	. . ...

Table 4
User–tag- source.

UserID	TagID	SourceID
22	7434	4
1432	765	8128
54,643	98,195	87
. . ...	. . ...	. . ...
. . ...	. . ...	. . ...

Fig. 3. Word cloud of tags.

lished in [51]. In this dataset, users are connected in a social network created from Delicious interactions [58]. The sample data of this relations shown in Table 3. In the data set each user has their own tags, sources, and tag assignments [58]. Table 4 show the sample data.

In this article, the trained and test data as 80% and 20% of the total dataset were determined. Recommendations are produced based on the known information in the training set, and then to evaluate the performance of the proposed system test set is used.

All experiments implemented an Intel(R) Core i7 computer with 2.67 GHz CPU and 16.00 GB RAM. Google Colab has been used to train the most of the models and the AmpliGraph library has been used to implement the knowledge graph representation learning techniques.

In order to make the information of the dataset clearer, the visualization of word cloud of tags and the knowledge graph are presented in Figs. 3, 4. In Fig. 3, each tag is picturized with its frequency. The generated knowledge graph in Fig. 4 is plotted with Gephi software. This graph includes 111,987 nodes and 746,404 edges. To have a great depth of knowledge of the dataset, the contents of the tag data and construction graph are visualized.

4.2. Parameter settings

Parameters of the approach has been selected using the grid search and the parameters that have been searched are learning rate and the embedding size of the knowledge graph embedding method (pre-trained vectors). Best parameters that have been selected for this work are 0.001 for adam optimizer learning rate and 128 for vector size.

In total, our approach has 14,334,336 parameters that are pre-trained using knowledge graph embedding. All of these parameters fine-tuned on the down-stream task and none of them freezed during training.

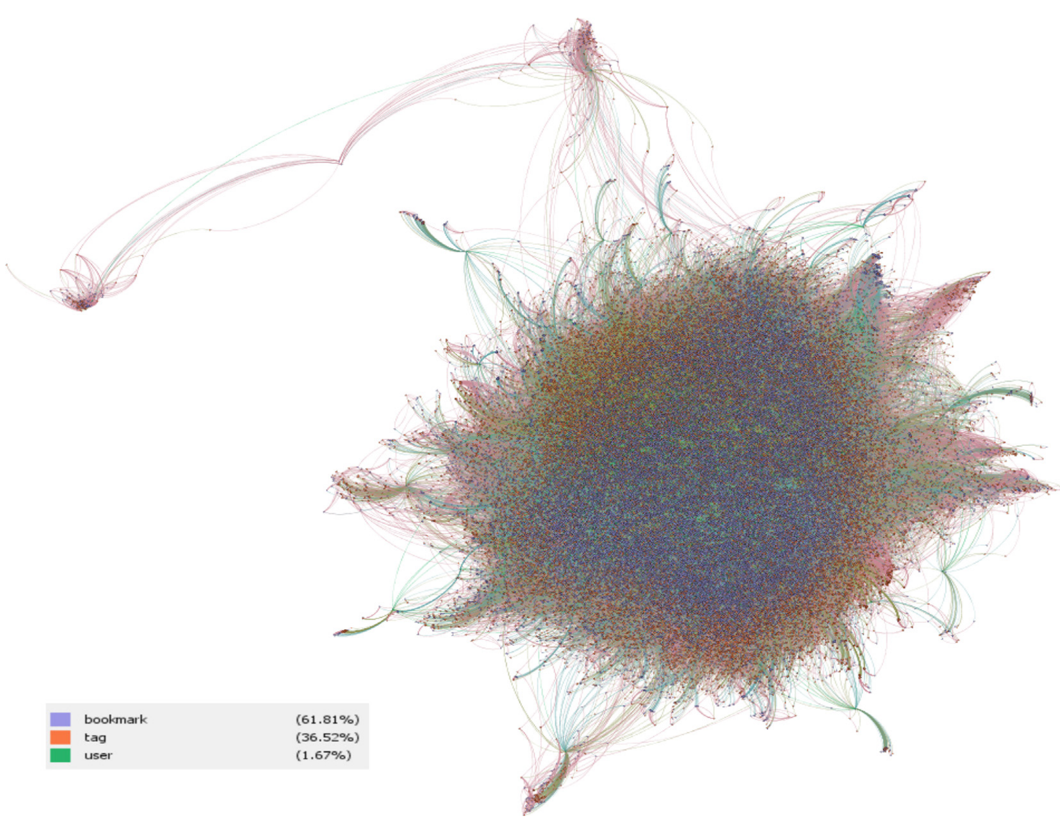


Fig. 4. The constructed knowledge graph from dataset.

4.3. Evaluation metrics

The evaluation metrics used in the literature are top-k precision, recall, and F1-score. The same metrics were also used to make the results comparable to previous works. This metrics are the most popular and powerful metrics to evaluate quality of recommender systems [53]. The acceptable range for all three criteria is between 0 and 1. One of the challenges for researchers is to design approaches that can increase these values. These metrics are described in Eqs. (9), (10), and (11) according to [59].

$$P = \frac{|lr \cap lt|}{|lr|} \tag{9}$$

$$R = \frac{|lr \cap lt|}{|lt|} \tag{10}$$

$$F1 = 2 \times \frac{p \times R}{p + R} \tag{11}$$

In Eqs. (9) and (10), lr is the recommended items list and lt is the list of items being tested.

4.4. Baselines

To evaluate the efficiency of the proposed method for sources, users and, tags recommendations, the baseline algorithms were divided into three groups. While one class includes previous studies that recommend source, the other reveals the researches proposed models for user and tag recommendation. The following models recommend source:

- CCS method: The basis of this method is the use of cosine similarity based on clustering. In other words, to model users and sources as attributes vector based on the hierarchical clustering method has been used. Then content-based filtering based on cosine similarity has been used to provide recommendations [60].
- ACF method: In this method, the automatic encoder is used in the content-based method. An automated encoder is used to obtain summary introductions from user profiles along with recommendations based on collaborative filtering. This method has shown with different tests on collaborative filtering and with different number of hidden layers that deeper architectures can provide better performance if the depth of the neural network is properly set [56].
- CCF or CF based on clustering: This method is similar to CCS but here CF method based on user is used to provide recommendations [61].
- PMF method: This technique is also based on collaborative filtering. It uses a user rating matrix. This method works with the assumption that similar sets of items, if ranked by users, means that those users have similar preferences [62].
- KGAT: this is state-of-the-art knowledge-based model, performs knowledge-aware attentive graph convolution in KG for high-order modeling of relation [63].

In the succeeding paragraphs, the baselines for user recommendation are listed:

- Tag-co: In this algorithm used the cosine similarity. It calculates user similarity based on tags [64].
- Item-co: This method similar to Tag_co, with the difference that in this method calculates user similarity based on items [64].
- Tri-graphs: This method provides a recommendation algorithm with diffusion based on bipartite graphs. Then it uses the integrated algorithm on the graphs and calculates the similarity of the users by the graphs [60].
- FURG: Based on the user's interest, a user similarity graph is used to recommend a friend. First, in this method, LDA is applied to identify topics of interest to users in order to reduce the problems of tag redundancy and data sparseness. Then, an interest-based user similarity graph is constructed by various views of users' favorite topics, shared collections, and co-annotated tags. Finally, it provides interest-based recommendations by graph mining [64].

In the following paragraphs, the baselines for tag recommendation are listed:

- CFA: It Uses a sparse auto-encoder to latent representations of user profiles and applied user-based CF for recommendation [56].
- DSPR: In this method, MLPs are used to extract user and item representations. Here, MLPs with common parameters were suitable for tag-based feature processing [57].
- DeepFM: factorization machines [65] and deep neural network combined for feature learning. Then recommendations of the tags are done [66].
- PinSage: this technique deploys GraphSage [67] on industrial application. Then extracts node representations through non-spectral graph convolution [27].

4.5. Evaluation results

In this section, the efficiency of the proposed model will be investigated on the three different recommendation items: source, user and tag according to recall, precision and F1-score metrics. It should be noted that top-k in evaluation metrics shows number of relative items that are used for validation of approach. In addition to this, the effect of the number of embedding dimensions will be examined on the performance. Then, in order to make more judgments about our solution, the clusters created by embeddings of users, tags, and sources are going to be visualized.

4.5.1. Source recommendation results

The three evaluation criteria, recall, precision and F1-score show that the proposed method have been significantly improved, according to the known algorithms. The results of source recommendation based on different criteria have shown in Table 5, 6 and 7. According to the results provided in Tables 5-7, it can be

Table 5
Source Recommendation Results (Recall in %).

Models		R@5	R@15	R@30
CCF		0.439	1.051	1.499
ACF		0.590	1.209	1.917
CCS		0.938	2.271	3.774
PMF		1.302	2.851	4.988
KGAT		12.121	18.080	22.654
Proposed method(TransE)	64	12.3062	18.1397	23.4488
	128	13.0062	18.7049	22.7944
	256	11.1136	17.6240	23.7954
Proposed method(ComplEx)	64	13.0350	19.1087	23.3158
	128	13.0450	19.1630	23.5312
	256	**13.2771**	**19.2323**	**23.8975**
Proposed method(ConvE)	64	12.0131	17.0943	22.0237
	128	12.7751	18.3415	23.4005
	256	11.8413	17.6472	21.8941

Table 6
Source Recommendation Results (Precision in %).

Models		P@5	P@15	P@30
CCF		0.913	0.757	0.597
ACF		1.120	0.909	0.736
CCS		2.397	1.903	1.564
PMF		9.157	7.467	6.784
KGAT		14.01	13.88	11.98
Proposed method(TransE)	64	15.3333	14.0033	13.6533
	128	16.6897	14.0011	13.7778
	256	15.5556	14.7963	13.5852
Proposed method(ComplEx)	64	15.6667	14.1111	13.3704
	128	15.3514	14.7027	13.6396
	256	**16.7846**	**14.9487**	**13.6610**
Proposed method(ConvE)	64	16.0120	14.7667	13.5556
	128	15.6667	14.5556	13.2302
	256	16.0000	14.6800	13.2500

Table 7
Source Recommendation Results (F1 in %).

Models		F1@5	F1@15	F1@30
CCF		0.593	0.880	0.854
ACF		0.791	1.038	1.064
CCS		1.349	2.070	2.205
PMF		2.280	8.791	8.803
KGAT		12.998	15.704	15.672
Proposed method(TransE)	64	13.6540	15.8053	17.2580
	128	14.6195	16.3704	17.1746
	256	12.9647	16.0868	17.2959
Proposed method(ComplEx)	64	14.2302	16.2340	16.9950
	128	14.1045	16.6391	17.2693
	256	**14.8262**	**16.8221**	**17.3843**
Proposed method(ConvE)	64	13.7272	15.8455	16.7819
	128	14.0739	16.2307	16.9035
	256	13.6101	16.0274	16.5090

concluded that presenting data in form of this knowledge graph, and using collaborative neural network make source recommendation more accurate. Additionally, it is obviously that in all most cases ComplEx of our approach using 256 embedding sizes provides best results on top-k precision, Recall, and F1-score.

4.5.2. User recommendation results

Tables 8-10 illustrate the obtained results for user recommendation. The results demonstrate that our solution could significantly improve the performance in terms of recall, precision and F1-score.

Table 8
User Recommendation Results (Recall in %).

Models		R@5	R@10
Tag-co		6.3152	9.7991
Item-co		6.9220	9.8252
Tri-graphs		7.0133	11.7367
FURG		7.9035	12.0442
Proposed method(TransE)	64	13.8918	19.2987
	128	13.2872	18.8954
	256	**13.9439**	**19.3883**
Proposed method(ComplEx)	64	12.3077	18.6760
	128	13.0839	18.2517
	256	12.5152	19.3939
Proposed method(ConvE)	64	12.7366	18.8205
	128	12.5909	19.0027
	256	13.0545	18.9798

Table 9
User Recommendation Results (Precision in %).

Models		P@5	P@10
Tag-co		10.3696	8.0450
Item-co		11.3658	8.0664
Tri-graphs		11.3658	9.6668
FRUG		14.5581	11.0927
Proposed method(TransE)	64	15.0000	13.0012
	128	15.0321	13.0021
	256	**16.0210**	**14.1200**
Proposed method(ComplEx)	64	14.3333	13.0313
	128	14.0100	13.0000
	256	14.0110	13.0100
Proposed method(ConvE)	64	14.0010	13.701
	128	14.0110	13.012
	256	15.0010	14.0901

Table 10
User Recommendation Results (F-Measure in %).

Models		F@5	F@10
Tag-co		7.8497	8.8354
Item-co		8.6039	8.8593
Tri-graphs		8.6741	10.6016
FRUG		10.2450	11.5488
Proposed method(TransE)	64	14.4246	15.5360
	128	14.1059	15.4043
	256	**14.9105**	**16.3400**
Proposed method(ComplEx)	64	13.2435	15.3512
	128	13.5311	15.1846
	256	13.2209	15.5731
Proposed method([ConvE]	64	13.3389	15.8578
	128	13.2630	15.4468
	256	13.9602	16.1735

Table 11
tag Recommendation Results (Recall in %).

Models		R@10	R@20
CFA		3.69	3.98
DSPR		12.19	13.49
Deep FM		12.41	13.87
PinSage		12.12	13.25
Proposed method(TransE)	64	14.1884	16.0783
	128	14.7153	**17.0409**
	256	15.5551	16.0366
Proposed method(ComplEx)	64	14.3212	16.4979
	128	**15.9978**	16.8458
	256	14.9218	16.9050
Proposed method(ConvE)	64	15.7249	16.6398
	128	15.8115	16.5984
	256	15.5551	16.6483

There are three major reasons why our system can suggest users in accurate manner. Firstly, our constructed knowledge graph relies on user-user, user-tag and tag-source interactions. Secondly, knowledge graph embedding techniques infer and analyze these connections when they learn items' representations. Finally, using the knowledge graph embedding vectors along with neural collaborative filtering could provide better understandings of items specifically users. In other words, learning users' features along with sources and tags guaranties better user recommendation. Furthermore, it is clear that, user recommendation employing TransE (with 256 embedding sizes) as graph representation learning, can engender better results on top-k precision, recall, and F1-score.

Table 12
Tag recommendation results (precision in %).

Models		P@10	P@20
CFA		3.51	4.36
DSPR		12.12	14.69
Deep FM		11.90	12.71
PinSage		14.16	10.27
Proposed method([TransE]	64	16.3051	13.3051
	128	15.3750	14.1406
	256	15.0159	12.8730
Proposed method([ComplEx]	64	16.2648	14.2085
	128	**16.5500**	**14.2106**
	256	16.3651	13.1429
Proposed method([ConvE]	64	16.2899	13.2464
	128	15.7532	13.6623
	256	15.9322	13.5424

Table 13
Tag recommendation results (f1 in %).

Models		F1@10	F1@20
CFA		0.5865	0.5265
DSPR		2.5443	2.2772
Deep FM		1.0322	1.0101
PinSage		13.9365	13.981
Proposed method(TransE)	64	15.1733	14.5608
	128	15.0379	14.8387
	256	15.2807	14.2817
Proposed method(ComplEx)	64	15.2312	15.2679
	128	**16.2692**	**15.4164**
	256	15.6102	14.7884
Proposed method(ConvE)	64	16.0024	14.7504
	128	15.7823	14.9879
	256	15.7414	14.9356

4.5.3. Tag recommendation results

The tag recommendation results can be shown in Tables 11-13. It is obviously that in most cases ComplEx of our approach using 128 embedding sizes provides best results on top-k precision and F1-score measures. TransE with 128 embedding sizes on top 20 recall measure, imply best performance.

The main reason that how our system can precisely make suggestions is that the relation between tag-source in our created graph contributes to understand the differences between concepts and our system can better distinguish sources of ambiguous tags. Moreover, the user-user interaction enables our system to make recommendations for new users who just follow their friends and there are no histories about their tagging behaviors.

4.5.4. Visualization task

In this section, the extracted embedding was used to cluster users, tags and sources. For this goal, t-SNE dimensionality reduction and K-Means clustering method with 10 cluster on the embeddings obtained from ComplEx with size 64 were applied. Since lower dimensions cannot offer significant features than high dimensional feature space, the size 64 for the embedding dimensions was considered to evaluate the performance of the proposed method in a low dimensional feature vector. The results of these clusters shown in Figs. 5-7. The important goal of this investigation is to discover how to do clustering with created user, tag and source embeddings. The embeddings derived from our proposed learning model handles the problem of high dimensionality in an effective manner, hence, more accurate clusters are formed. A great clustering of tags leads to more precise recommendations. By looking at how users tag sources, the recommendation system is able to precisely recognize users' preferences and eventually do the specified recommendation.

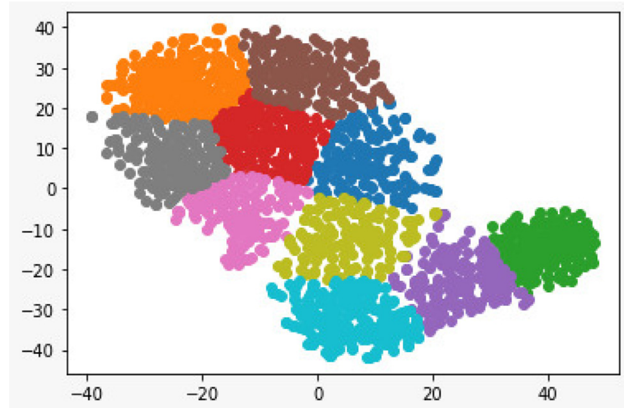


Fig. 5. User clusters: t-SNE dimensionality reduction / K-Means Clustering with 10 clusters.

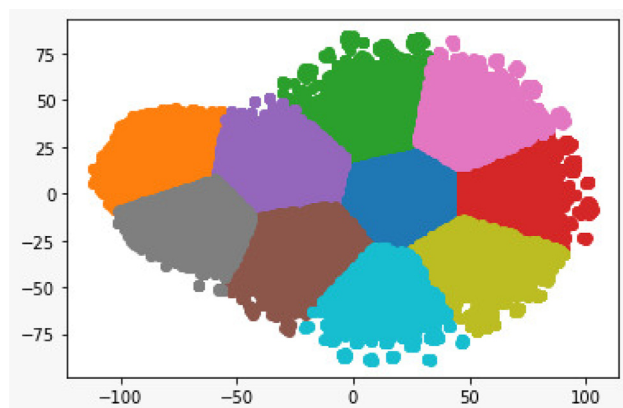


Fig. 6. Tag clusters: t-SNE dimensionality reduction / K-Means Clustering with 10 clusters.

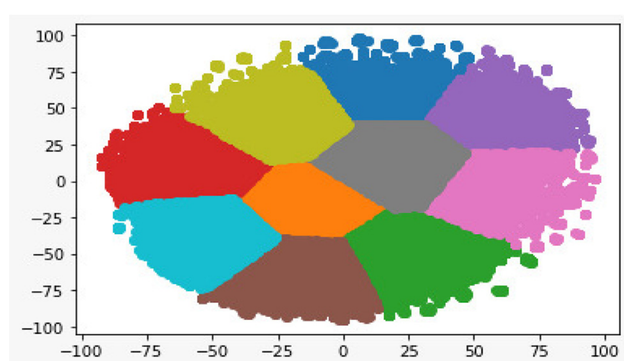


Fig. 7. Source clusters: t-SNE dimensionality reduction / K-Means Clustering with 10 clusters.

For more investigation, the community detection task on the knowledge graph were carried out. These communities are detected using Louvain algorithm. The modularity value is 0.413. Normally, regardless of the semantic relationship, and only from the knowledge graph and communications of the nodes, communities have obtained, that is shown in Fig. 8. Also, Fig. 9 indicates the size of each community.

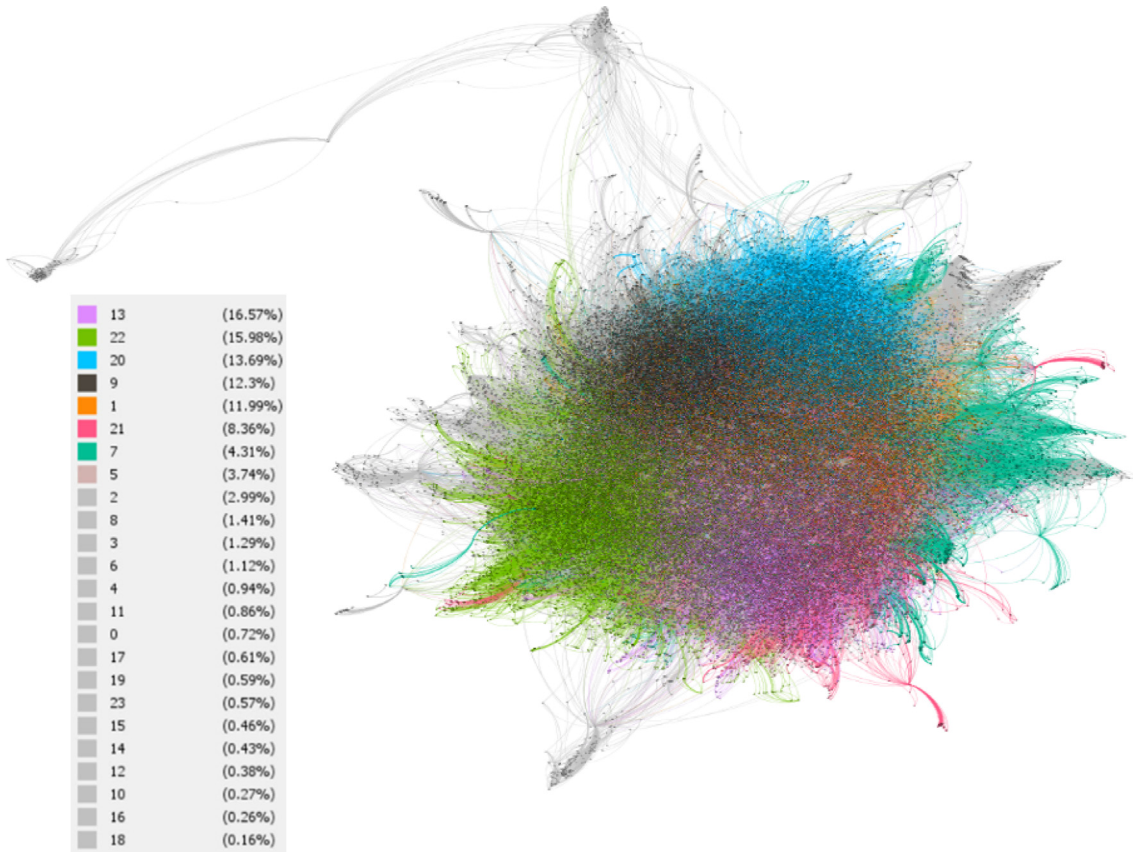


Fig. 8. Communities detected from the knowledge graph.

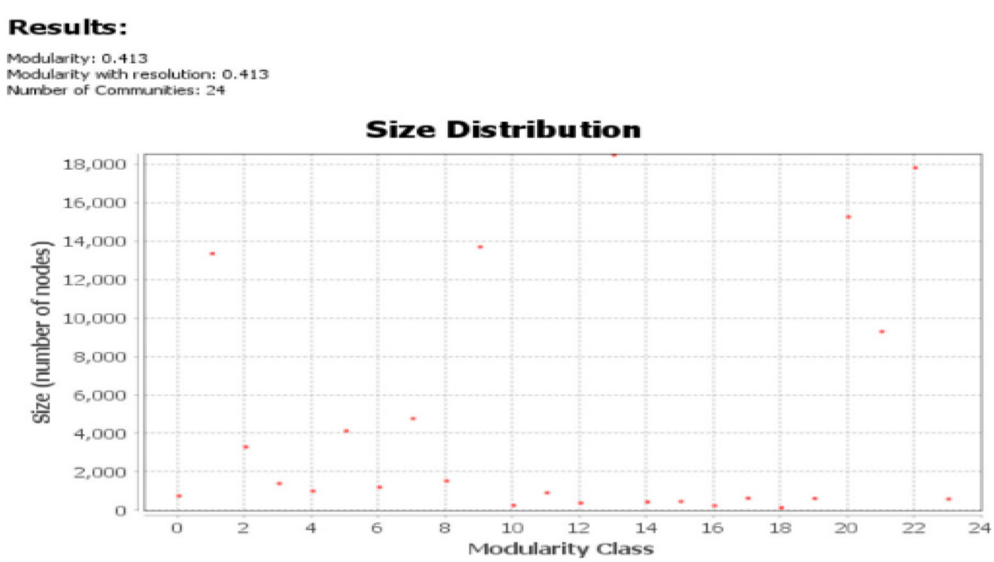


Fig. 9. Size of communities.

Two main observations can be conducted from the figure. Firstly, by applying the community detection algorithm on the constructed knowledge graph, a group of tags, users, and sources (bookmarks) that are similar to each other can be generated in communities. The second monitoring point is that the communities illustrate how ambiguity and redundancy can be addressed with the help of multi-hop neighbors in our constructed knowledge graph.

5. Conclusion

In this paper, the researchers are motivated to formulate the recommendation task as a classification problem by constructing a knowledge graph from user, tag, and source entities. The key idea behind using embeddings is to accurately find similar items. Fundamentally, the hypothesis declares that a knowledge graph constructed from different types of items aids in comprehending how traits are related to one another. Moreover, neural collaborative filtering assets to learn accurate representations of items. To capture statistical dependencies and provide more contextual semantic information, the problem was formulated in terms of knowledge graph construction. The connections in the knowledge graph contributes to understand the differences between concepts. Moreover, a part of graph constructs based on the user-user interaction that make it possible to suggest tags and sources to new users with no histories about their tagging behaviors. Additionally, these interactions enable this model to deal with the cold-start problem for new tag or source. Broadly, the researchers argue how the model can ease ambiguity and redundancy issues that arise as a result of the lack of contextual semantics for tag and source recommendation.

The proposed method uses simple matrix operations and learns over them with no need for extra parameters on model Utilization of pre-trained vectors from knowledge graph embedding ensures that the pre-trained vectors are more adapted to the structure of graph. These embeddings are semantic representations of the graph for each node regardless of its type. The size of these vectors is all the same for any entity type (User, Tag, and Source).

Additionally, the simplicity of our approach in combining pre-trained vectors in the architecture makes it much easier to use any other knowledge graph embedding and keep the training time much smaller. On the other hand, instead of using a very huge model with complexities on different sides (from number of layers to types of layers), only a simple neural collaborative neural network that includes vector operations on entity vectors is utilized which makes model to fine-tune embeddings for the downstream task faster and with less data. All of these novelties in the methodology helps the approach to overcome much complex methods. Binarization of model output and creating negative samples in order to create a balanced dataset also makes outputs more robust to data and concept drift.

In the future, the researchers aim to formalize the problem as a dynamic knowledge graph. They reckon that dynamic knowledge graph construction is a trendy solution that may raise interesting questions. So, they are eager to model the sequential dependencies over the user-tag, user-source and source-tag interactions in a sequence. In other words, in the future work, they will investigate a dynamic knowledge graph-based approach that recommends items in a sequential form of interactions.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

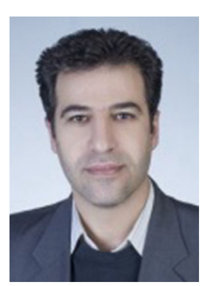
[1] Ahmadian S, Ahmadian M, Jalili M. A deep learning based trust-and tag-aware recommender system. Neurocomputing 2022;488:557–71.
[2] Chen YC, Hui L, Thaipisutikul T. A collaborative filtering recommendation system with dynamic time decay. J Supercomput 2021;77:244–62.
[3] Ni J, Huang Z, Cheng J, Gao S. An effective recommendation model based on deep representation learning. Inf Sci 2021;542:324–42.
[4] Hou YE, Gu W, Dong W, Dang L. A Deep Reinforcement Learning Real-Time Recommendation Model Based on Long and Short-Term Preference. Int J Comput Intell Syst 2023;16(1):4.
[5] Sun Z, Guo Q, Yang J, Fang H, Guo G, Zhang J, et al. Research commentary on recommendations with side information: A survey and research directions. Electron Commer Res Appl 2019;37:100879.
[6] Mısır M, Sebag M. Alors: An algorithm recommender system. Artif Intell 2017;244:291–314.
[7] Wang R, Jiang Y, Lou J. ADCF: Attentive representation learning and deep collaborative filtering model. Knowl-Based Syst 2021;227:107194.
[8] Jamali M, Ester M. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. 2009, June: p. 397–406.
[9] Chen X, Jia S, Xiang Y. A review: Knowledge reasoning over knowledge graph. Expert Syst Appl 2020;141:112948.
[10] Guo Q, Zhuang F, Qin C, Zhu H, Xie X, Xiong H, et al. A survey on knowledge graph-based recommender systems. IEEE Trans Knowl Data Eng 2020.
[11] Zhang W, Gu T, Sun W, Phatpicha Y, Chang L, Bin C. Travel attractions recommendation with travel spatial-temporal knowledge graphs. In: International conference of pioneering computer scientists, engineers and educators. Singapore: Springer; 2018. p. 213–26.
[12] Pla Karidi D, Stavrakas Y, Vassiliou Y. Tweet and followee personalized recommendations based on knowledge graphs. J Ambient Intell Hum Comput 2018;9(6):2035–49.
[13] Yan C, Chen Y, Zhou L. Differentiated fashion recommendation using knowledge graph and data augmentation. IEEE Access 2019;7:102239–48.
[14] Wang X, Wu H, Hsu CH. Mashup-oriented API recommendation via random walk on knowledge graph. IEEE Access 2018;7:7651–62.
[15] Christoforidis G, Kefalas P, Papadopoulos A, Manolopoulos Y. Recommendation of points-of-interest using graph embeddings. In: In 2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA). IEEE; 2018. p. 31–40.
[16] Hu S, Tu Z, Wang Z, Xu X. A POI-sensitive knowledge graph-based service recommendation method. In: In 2019 IEEE International Conference on Services Computing (SCC). IEEE; 2019. p. 197–201.
[17] Deng, W., Zhu, P., & Ma, J. (2018, June). Enhancing group recommendation by knowledge graph. In 22nd Pacific Asia Conference on Information Systems (PACIS 2018) (p. 214).
[18] Cui P, Wang X, Pei J, Zhu W. A survey on network embedding. IEEE Trans Knowl Data Eng 2018;31(5):833–52.
[19] Oramas S, Ostuni VC, Noia TD, Serra X, Sciascio ED. Sound and music recommendation with knowledge graphs. ACM Trans Intell Syst Technol (TIST) 2016;8(2):1–21.
[20] Lu C, Laublet P, Stankovic M. Travel attractions recommendation with knowledge graphs. In: European Knowledge Acquisition Workshop. Cham: Springer; 2016. p. 416–31.
[21] Hogan A, Blomqvist E, Cochez M, d'Amato C, Melo GD, Gutierrez C, et al. Knowledge graphs. ACM Comput Surveys (CSUR) 2021;54(4):1–37.
[22] Sun Z, Yang J, Zhang J, Bozzon A, Huang L-K, Xu C. Recurrent knowledge graph embedding for effective recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems. 2018, p. 297–305.
[23] Fan H, Zhong Y, Zeng G, Ge C. Improving recommender system via knowledge graph based exploring user preference. Appl Intell 2022:1–13.
[24] Makarov I, Kiselev D, Nikitinsky N, Subelj L. Survey on graph embeddings and their applications to machine learning problems on graphs. PeerJ Comput. Sci. 2021;7.
[25] Dettmers T, Minervini P, Stenetorp P, Riedel S. Convolutional 2d knowledge graph embeddings. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1) 2018, April.
[26] Wang H, Zhang F, Xie X, Guo M. Dkn: Deep knowledge-aware network for news recommendation. In: Proceedings of the 2018 World Wide Web Conference, ser. WWW '18. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2018, p. 1835–44.
[27] Ying R, He R, Chen K, Eksombatchai P, Hamilton WL, Leskovec J. Graph convolutional neural networks for web-scale recommender systems. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. 2018, July: p. 974–83.
[28] Trouillon T, Welbl J, Riedel S, Gaussier É, Bouchard G. Complex embeddings for simple link prediction. In: International conference on machine learning. PMLR; 2016. p. 2071–80.
[29] Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O. Translating embeddings for modeling multi-relational data. Adv Neural Inf Proces Syst 2013;26.
[30] Shokeen J, Rana C. A study on features of social recommender systems. Artif Intell Rev 2020;53(2):965–88.
[32] Dai Y, Wang S, Xiong NN, Guo W. A survey on knowledge graph embedding: Approaches, applications and benchmarks. Electronics 2020;9(5):750.
[33] Xu M. Understanding graph embedding methods and their applications. SIAM Rev 2021;63(4):825–53.
[34] Chen Y, Zhou M, Zheng Z, Chen D. Time-aware smart object recommendation in social internet of things. IEEE Internet Things J 2019;7(3):2014–27.
[35] Wang H, Wang Z, Hu S, Xu X, Chen S, Tu Z. DUSKG: A fine-grained knowledge graph for effective personalized service recommendation. Futur Gener Comput Syst 2019;100:600–17.
[36] Wang H, Zhang F, Zhao M, Li W, Xie X, Guo M. May). Multi-task feature learning for knowledge graph enhanced recommendation. In: In The world wide web conference. p. 2000–10.
[37] Mezni H, Benslimane D, Bellatreche L. Context-aware service recommendation based on knowledge graph embedding. IEEE Trans Knowl Data Eng 2021.
[38] Wang Q, Mao Z, Wang B, Guo L. Knowledge graph embedding: A survey of approaches and applications. IEEE Trans Knowl Data Eng 2017;29(12):2724–43.
[39] Wang Z, Zhang J, Feng J, Chen Z. Knowledge graph embedding by translating on hyperplanes. in Twenty-Eighth AAAI conference on artificial intelligence, 2014.

[40] Lin Y, Liu Z, Sun M, Liu Y, Zhu X. February). In Twenty-ninth AAAI conference on artificial intelligence: Learning entity and relation embeddings for knowledge graph completion; 2015.
[41] Ji G, He S, Xu L, Liu K, Zhao J. July). Knowledge graph embedding via dynamic mapping matrix. In: In Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers). p. 687–96.
[42] Yang B, Yih WT, He X, Gao J, Deng L. Embedding entities and relations for learning and inference in knowledge bases; 2014. arXiv preprint arXiv:1412.6575.
[43] Hu B, Shi C, Zhao WX, Yu PS. July). Leveraging meta-path-based context for top-n recommendation with a neural co-attention model. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. p. 1531–40.
[44] Tiddi I, Schlobach S. Knowledge graphs as tools for explainable machine learning: A survey. Artif Intell 2022;302:103627.
[45] Chen Y. Convolutional neural network for sentence classification. University of Waterloo); 2015. Master's thesis.
[46] Huang J, Zhao WX, Dou H, Wen JR, Chang EY. June). Improving sequential recommendation with knowledge-enhanced memory networks. In: In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. p. 505–14.
[47] Zhang Y, Ai Q, Chen X, Wang P. Learning over knowledge-base embeddings for recommendation; 2018. arXiv preprint arXiv:1803.06540.
[48] Wang H, Zhang F, Hou M, Xie X, Guo M, Liu Q. February). Shine: Signed heterogeneous information network embedding for sentiment link prediction. In: In Proceedings of the eleventh ACM international conference on web search and data mining. p. 592–600.
[49] Yang D, Guo Z, Wang Z, Jiang J, Xiao Y, Wang W. In: A knowledge-enhanced deep recommendation framework incorporating GAN-based models. IEEE; 2018. p. 1368–73.
[50] Dong Y, Chawla NV, Swami A. metapath2vec: Scalable representation learning for heterogeneous networks. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. 2017, August, pp. 135–144.
[51] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space; 2013. arXiv preprint arXiv:1301.3781.
[52] Ye Y, Wang X, Yao J, Jia K, Zhou J, Xiao Y, et al. November). Bayes embedding (bem) refining representation by integrating knowledge graphs and behavior-specific networks. In: Proceedings of the 28th ACM international conference on information and knowledge management. p. 679–88.
[53] Colombo-Mendoza LO, Valencia-García R, Rodríguez-González A, Alor-Hernández G, Samper-Zapater JJ. RecomMetz: A context-aware knowledge-based mobile recommender system for movie showtimes. Expert Syst Appl 2015;42(3):1202–22.
[54] Chicaiza J, Valdiviezo-Diaz P. A comprehensive survey of knowledge graph-based recommender systems: Technologies, development, and contributions. Information 2021;12(6):232.
[55] Nikzad-Khasmakhi N, Balafar MA, Feizi-Derakhshi MR. The state-of-the-art in expert recommendation systems. Eng Appl Artif Intel 2019;82:126–47.
[56] Zuo Y, Zeng J, Gong M, Jiao L. Tag-aware recommender systems based on deep neural networks. Neurocomputing 2016;204:51–60.
[57] Xu Z, Chen C, Lukasiewicz T, Miao Y, Meng X. October). Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. p. 1921–4.
[58] Shokrzadeh Z, Feizi-Derakhshi M.-R., Balafar MA, Bagherzadeh-Mohasefi J. Graph-Based Recommendation System Enhanced with Community Detection; 2022. arXiv preprint arXiv:2201.03622.
[59] Zhang J, Peng Q, Sun S, Liu C. Collaborative filtering recommendation algorithm based on user preference derived from item domain features. Physica A 2014;396:66–76.
[60] Shepitsen A, Gemmell J, Mobasher B, Burke R. October). Personalized recommendation in social tagging systems using hierarchical clustering. In: Proceedings of the 2008 ACM conference on Recommender systems. p. 259–66.
[61] Xu Z, Yuan D, Lukasiewicz T, Chen C, Miao Y, Xu G. Hybrid deep-semantic matrix factorization for tag-aware personalized recommendation. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), p. 3442–6. IEEE; 2020, May.
[62] Mnih, A., & Salakhutdinov, R. R. (2007). Probabilistic matrix factorization. Advances in neural information processing systems, 20.
[63] Wang X, He X, Cao Y, Liu M, Chua TS. KGAT: Knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. p. 950–8.
[64] Wu BX, Xiao J, Chen JM. Friend recommendation by user similarity graph based on interest in social tagging systems. In International Conference on Intelligent Computing; 2015, August. p. 375–86. Springer, Cham.
[65] Rendle S. Factorization machines. In 2010 IEEE International conference on data mining, 2010, December, p. 995–1000, IEEE.
[66] Guo H, Tang R, Ye Y, Li Z, He X. DeepFM: a factorization-machine based neural network for CTR prediction, 2017. arXiv preprint arXiv:1703.04247.
[67] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. In: Advances in neural information processing systems. p. 30.

Zeinab Shokrzadeh was born in Tabriz, Iran, in 1975 and received her B.S. degree from Isfahan University in computer engineering. She received her M.S. degree in computer engineering from Islamic azad university sience and research branch. Her candidate Ph.D. student is in Islamic azad university Urmia branch from Department of Computer Engineering. She is a lecturer in Department of Software Engineering, shabestar University.

Mohammad-Reza Feizi-Derakhshi received his B.S. in Software Engineering from the University of Isfahan. Then he received his M.S. and Ph.D. in Artificial Intelligence from the Iran University of Science and Technology. He is currently a faculty member at the University of Tabriz. His research interests include: natural language processing, optimization algorithms, social network analysis and intelligent databases.

Mohammad-Ali Balafar is associate professor at university of Tabriz. His research area includes AI and Computer Vision. He has published more than 30 journal papers in respected journals.



Jamshid Bagherzadeh Mohasefi was born in Urmia, Iran, in 1974 and received his B.S. degree from Sharif University in computer engineering. He received his M. S. degree in computer engineering from Sharif University, Tehran in 2005. His Ph.D. degree was from IITD. He is a faculty member at urmia univetsity in Department of Software Engineering urmia.