Building Semantic Based Recommender System Using Knowledge Graph Embedding

Miriyala Kartheek*, G P Sajeev†‡
Dept of Computer Science and Engineering, Amrita Vishwa Vidyapeetham, Amritapuri, India
‡Dept of Electronics & Communication, Govt Engineering College Kozhikode, India
Email: *miriyala.karthik@gmail.com, †sajeevgp@am.amrita.edu, ‡gpsajeev@gmail.com

2021 Sixth International Conference on Image Information Processing (ICIIP) | 978-1-6654-3361-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICIIP53038.2021.9702632

***Abstract*—Recommendation systems are information filtering mechanisms used in E-commerce, media and entertainment industry. It essentially facilitate the customers for a better user experience by processing the content user-specific. This is known as personalization. However, though leveraged by machine learning algorithms existing recommendation systems, still suffers from the problem of *cold-start* and *sparcity*. These problems could be resolved by using knowledge graphs since it gives a semantic explanation of recommendations. Also, graph learning method overcomes the problems of manual feature extraction and is effective for feature learning in predicting tasks. In this research, we develop a semantic based recommender through link prediction in a knowledge graph. We apply graph embedding techniques for extracting the semantics of explicable recommendations. The proposed method is validated by building a knowledge graph using the *MovieLens* dataset. We observed that factorization based scoring functions such as *HolE* and *DistMult* provides better semantic recommendations.**

***Index Terms*—Recommendations, Knowledge Graph, Link Prediction, Graph Embedding, Personalization.**

I. Introduction

Recommendation systems are one of the most important streams of Artificial Intelligence that helps in enhancing user experience by recommending web content, products and services. In recent years, due to the exponential growth of E-commerce users and products [13], it is difficult for users to find the right product or content in less amount of time. Recommendation engines use user-item interactions like watching history and product purchase history to meet personalized recommendations [20].

Among many recommender algorithms, collaborative filtering (CF) [8], [21] is the popular one. It filters information from *item-user* interactions and makes the recommendations based on *user-item* common preferences. On the other hand, content based recommendation system [18], [19] uses meta information of items and users for making recommendations. Also, there exist hybrid methods by combining CF and content of users such as images, social networks and text content [27]. Nevertheless, all these methods suffer from sparsity and cold-start problems.

Knowledge graphs can handle representation of real world data that explicitly represent the knowledge. Knowledge graphs are been used by many companies, like Google Knowledge Graph [22], Facebook [16], Linked In [11]. Knowledge graph contain nodes and edges where each node is an entity and edge is a representation of relation between entities.

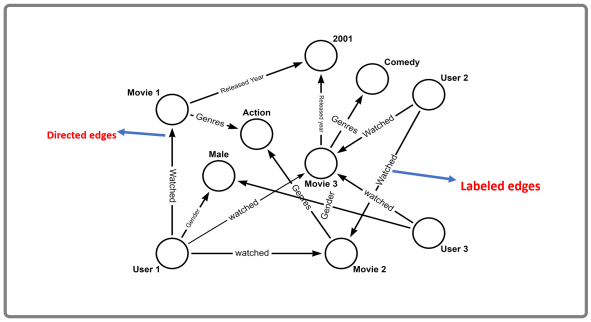

Figure 1: Example of a Knowledge Graph.

RDF-style triplets (h, r, t) is a data structure that uses to represent Knowledge Graph, where (h, t) are entities and r is the relation between entities, Fig. 1 shows entities and their relationships (directed). However, the explosion of data forms millions of entities and relations, where it becomes hard for traditional graph structures to manipulate knowledge graphs. The two main drawbacks of traditional knowledge graphs are

- Computational efficiency is low for mapping relations between entities, hence need a new algorithm.
- Large scale of knowledge suffers from sparsity due to a large number of nodes and relations, resulting in inaccurate inferential relations of entities.

Knowledge graph embedding can tackle the problem of representing entities and relationships to a lower dimension feature space without loosing graph properties. Besides, it calculates the relationships of entities in a lower dimensional feature space [7], and thus effectively solves sparsity issue and computational complexity. Knowledge graph embedding method learns semantic representations of entities and their relations and also explore new facts from existing facts, known as link prediction.

In general, knowledge graph utilizes distributed representation technology to raise the issues of traditional graph representation. Advantages of this approach are

- All elements, entities and relations of knowledge graphs are embedded to low feature dimensions, hence, overcomes sparsity.
- Representation learning connects heterogeneous objects by using unified feature space, makes fusion and calculations of different types of information.

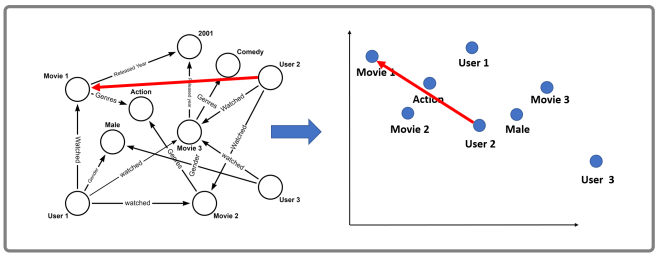

Figure 2: Illustration of recommendations by Predicting a link in Embedding Space (lower dimension feature space)

Graph embedding representation works better than traditional one-hot encoding, thereby improving semantic computing. Link prediction is a knowledge graph completion task that predicts missing relations from existing relations to build complete graph. There exist many link prediction techniques, viz. path based methods, decomposition models and embedding models.

In path based methods, a path from x to y on sequence of edges is considered. However, computational complexity is high for this method, especially when larger steps are needed for computing the optimal space. In decomposition model, entity relationships are encoded into tensors [5]. Based on potential semantic information, involves many parameters result in low efficiency and solubility. To solve the above mentioned issues embedding models are proposed. In this technique, knowledge graph is transformed into a lower feature space, without losing underlying semantics [14].

Knowledge graph embedding models for link prediction significantly advanced the state of art in past few years. To address the issues from traditional Recommendation systems, we propose a recommendation engine which adds semantic explanation for the recommendations by link prediction graph embedding models.

II. Related Work

Recommendation systems are one of the information filtering systems that provide related suggestions to user, predicting ratings for a given item or product by user. The customer behaviour is analyzed and then product recommendation is made. It has a wide range of applications, such as in the field of query searching, entertainment, social tags etc. Basically, there are two major types of recommendation systems, which are content-based and collaborative recommendation systems both have its own applications.

Recommendation systems encounter some common problems like scalability, long term and short term preferences, diversity of recommendations and other privacy issues. Recommendation systems can handle data overload by personalization to users.

In recent years, various recommendation system approaches are being developed mainly utilizing collaborative [1], content-based [18] or hybrid approach [4]. GroupLens employed collaborative approach locates news articles from a massive dataset [2].

Amazon improves recommendations by topic diversification algorithms. Hybrid approach joining of content and collaborative filtering methods can overcome limitation of Cold Start problem but sparsity still exists and lack of explainability of recommendations. Graph embedding can handle both the problems, it will add semantic explanation to the recommendation by link prediction which we are using in our approach.

Knowledge graph embedding can be categorized into three types translation based models, factorization based models and neural network based models. Scoring function measures the correctness of the triplet (h, r, t). In translation based models like TransE [3], TransH [25] and TransM [9], relation r translate head entity to the tail entity. Factorized models DistMult [26] and Complex [24] captures more semantic information than Translation models [28]. Using Convolutional neural networks or attention mechanisms, neural network based models gets better embeddings. Considering the computational complexity aspect, in this paper, translation TransE [3], RotatE [23] and factorization DistMult [26], HolE [15] models are utilized for recommendations.

III. The Proposed Recommender System

The embedding models represent knowledge graph to lower dimension by preserving the graph information and link prediction (recommendations) can perform by feature dimension vectors. In this work, recommendation engine works by predicting the missing links in a knowledge graph. To achieve efficient link predictions from a knowledge graph by graph embedding, we generate a Knowledge base. Further, for representing a fact in Knowledge base, a scoring function of triplet form is utilized. A negative generator and a loss function are utilized for increasing the correctness and for optimizing the score, respectively.

A. Knowledge base

Knowledge base is used for information retrieval, text understanding and question answering. Knowledge base is a set of triplets (h, r, t), where h is a head entity, t is a tail entity and r relation of (h, t) is the representation of a multi relational data. Mainly characteristic and user-item interactions data is relational. Creating a knowledge base schema from the relational data helps to form triplets.

Example 1. Characteristic information like User meta information such as age, gender, occupation and location can form triplets by keeping h as a user $headentity$, r relation (age, gender, occupation, location) and t $tail$ as the values of characteristic information.

B. Scoring function

Scoring function gives a triplet (h, r, t) a score value of correctness. Multiple triplets form a knowledge base where each triplet is a fact. For example a triplet (h, r, t) $(user1, watched, movie100)$ is a fact, similarly many facts

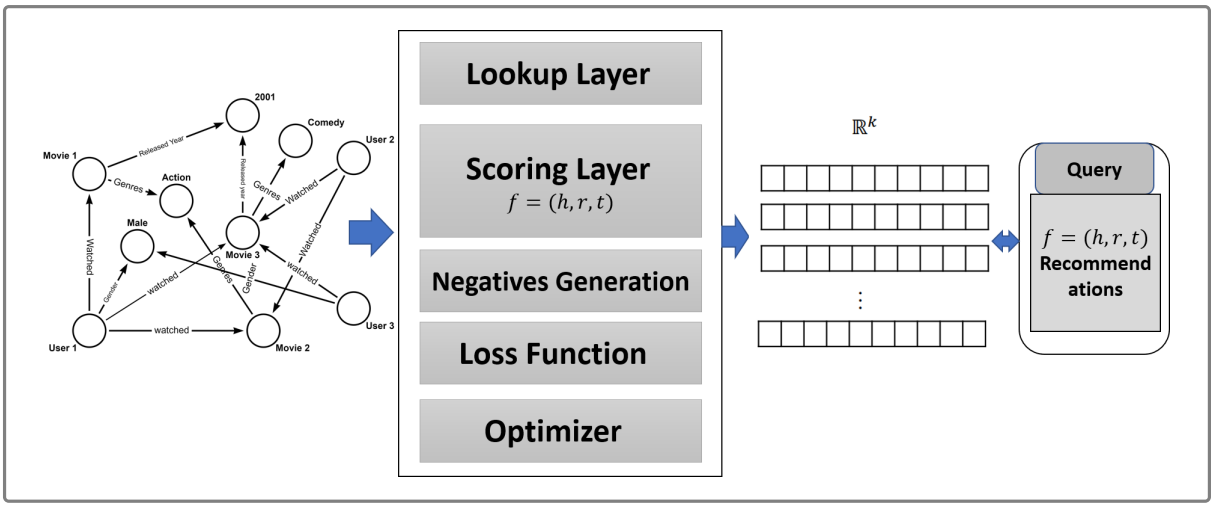

Figure 3: A Glance of Graph embedding training process.

are obtained from knowledge base. Based on the facts, recommendations should be given to $user1$ and the query will be $(user1, watched, ?)$. So accordingly for $tailentity$ there will be many movies available, scoring function will give the score for all possible movies, higher the score, higher will be the chance for the triplet to be true.

There are many scoring functions proposed by researchers, among them translation based TRansE [3](1), RotatE [5](2) and factorization based DistMult [26](3), HolE [15](4) methods are utilized in this paper to measure the correctness of the recommendations.

Translation-based Scoring Functions

$$f_{TransE} = -||(h + r - t)|| \tag{1}$$

$$f_{RotatE} = -||(h \circ r - t)||^2 \tag{2}$$

Factorization-based Scoring Functions

$$f_{DistMult} = (h^\intercal diag(r) t) \tag{3}$$

$$f_{HolE} = \frac{2}{n} f_{ComplEx}(h, r, t) \tag{4}$$

C. Negative generation

During the training of the knowledge graph embedding to generate a embedding space R for a node to distinguish true fact and false fact, negative samples and positive samples are introduced. Negatives $TableI$ are drawn by, from the Local Closed World assumption of a knowledge graph.

- e = user1, user2, user3, user4, movie1, movie2, movie3, movie4
- r = watched
- $t \in Groundtruth$ = (user1 watched movie1)

Negatives are drawn on both $headentity$ and $tailentity$ by relations form ground truth entities. By increasing correctness score of a triplet from a scoring function by optimizing loss function, feature embedding for corresponding entities are generated.

Table I: Illustration of synthetic negatives

head(h)	relation (r)	tail(t)
User 1	Watched	Movie 2
User 1	Watched	Movie 3
User 2	Watched	Movie 1
User 3	Watched	Movie 1
User 4	Watched	Movie 2

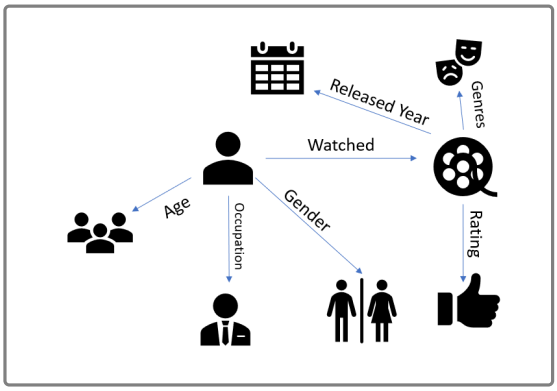

Figure 4: Knowledge Graph for Movie Recommendations

D. Loss function/Optimizer

Initially, all entity and relation embedding vectors of dimension R are initialised randomly. From the training set T, by sampling method, a subset T_{batch} is fed into the model. For each triplet of mini-batch, synthetic negative triplets are generated. Further, all generated negative set N_{batch} and training set T are incorporated as the new training batch X_{batch}. Finally, all parameters of entity and relation embedding vector parameters are updated by minimizing the loss function. There are two types of loss functions, margin based loss function

Algorithm 1: Training process for embeddings

Input: Training set of triplets T, embedding dimension k
Output: Entity embedding e and Relation Embedding r

Initialisation : Entity embedding e and Relation Embedding r with dimension k.
for T_{batch} **do**
  Initialize set of X_{batch}.
  for all T_{batch} **do**
    corrupted triplet for T_{batch}
    update X_{batch}
  end for
  updated e,r embedding by minimizing loss function.
end for

and logistic loss function, which indicates the category of the triplet (h, r, t), calculated by Stochastic gradient descent or Adam optimizer.

IV. Performance Evaluation

The proposed model is evaluated through experiments [12]. For this purpose we have developed a recommendation software using Python language. For creating a Knowledge base and analysing the data, the software library *Pandas* [17] is utilized. We use the open source library, *AmpliGraph* [6], for training and testing the proposed model.

A. Dataset

Experiments are conducted on Movielens dataset [10] to measure effectiveness of recommendations from graph embedding model. Dataset contains 1,000,209 ratings of 3,900 movies by 6,040 users. By using Schema Fig.4 knowledge base is created Table II $\#(h)$: head entities, $\#(t)$: tail entities, $\#(r)$: relation entities, $\#entities$: unique entities, gives meta information of the knowledge base.

For a given true test triplet (h, r, t), tail entity is replaced with every entity e from the set. The Scoring function provides score for each (h, r, e) corrupted triplet. After scoring, to get most relevant entity for $(h, r, ?)$, rankings are given to scores. So based on a query, recommendations are generated by the ordered score results. Lower the rank, higher the recommendation relevancy. If the rank for true test triplet (h, r, t) is low, it indicates that the model is well-trained. The process is also suitable for $(?, r, t)$. Illustration of ranking for the true test triplet is given in Table III. In Table III for illustration true test triplet is (User1 Watched Movie1) and query triplet is (User1 Watched **?**).

B. Evaluation Metrics

By ordering scored ranks, better the ranks for true triplets better the model trained. For Evaluating the performance of models used Mean Rank MR(5), Mean Reciprocal Rank MRR(6) and $Hits@N$(7).

Mean Rank MR is arithmetic mean of all ordered score rankings, lower the MR better the results. MR lies between 0 and ∞ is sensitive to outliers.

$$MR = \frac{1}{|N|}\sum_{i=1}^{N} R_i \tag{5}$$

Mean Reciprocal Rank MRR is average of inverse of the score rankings, in other words, inverse of the harmonic mean of ranks. MRR lies between 0 and 1, higher the value MRR, better the results.

$$MRR = \frac{1}{|N|}\sum_{i=1}^{N} \frac{1}{R_i} \tag{6}$$

$Hits@N$ is the proportion of predictions with a rank which is equal to, or less than a certain threshold. $Hits@N$ lies between 0 and 1, closer to 1 better the results.

$$Hits@N = \sum_{i=1}^{N} = 1 \; if \; R_i <= N \tag{7}$$

Table II: Movielence $1M$ Dataset Knowledge base

	$\#Triplets$	$\#(h)$	$\#(r)$	$\#(t)$	$\#entities$
Data	2028829	9873	3663	3784	13536
Train	2003829	9870	3663	3784	9896
Validation	5000	2697	1279	1301	3974
Test	20000	4770	2249	2327	6996

Table III: Ranking on score for a test triplet

Entity(h)	Relation(r)	Entity(t)	Score(f)	Ranking
User1	watched	Movie3	0.90	1
User1	watched	Movie2	0.85	2
✓User1	watched	Movie1	0.83	3
User1	watched	Movie100	0.70	4
User1	watched	Movie9	0.68	5
User1	watched	Movie25	0.65	6
User1	watched	Movie325	0.60	7

C. Results

From Movielens knowledge base, four embedding models are implemented to measure the effectiveness of recommendations with evaluation metrics MR, MRR and $Hits@N$. MR is the mean of ranks (position). MRR is the mean of the reciprocal ranks (inverse of ranks). $Hits@N$ is correct entities proportion ranked top N.

The MR value is highest for *TransE* model and it shows a slight improvement in *DistMult* method than that of *ComplexE*. However, MR value is slightly less in *HolE*, as can be seen from Table IV. For the metric MRR *HolE*, gives the highest values of 0.181. Note that, *ComplexE* and *DistMult* gives more or less a similar performance. For $Hits@100$ metric *HolE*, with a value of 0.416, outperforms other methods.

Table IV: Results

Embedding space $k = 300$, epochs 100					
Method	MR	MRR	$Hits@1$	$Hits@10$	$Hits@100$
TransE [3]	1568.19	0.142	0.079	0.245	0.360
ComplexE [24]	1081.87	0.162	0.095	0.270	0.360
DistMult [26]	1021.16	0.168	0.097	0.279	0.393
HolE [15]	1001.13	0.181	0.108	0.297	0.416

V. Conclusion

In this research, we have presented a recommender system by knowledge graph embedding. In general, traditional recommendation systems face the problems of cold-start, multi-relational, lack of explainability and sparsity. However, we infer that semantically meaningful recommendations could be produced using knowledge base, for better explainability. The experimental results indicate that factorization based embedding methods like DistMult and HolE generates better recommendations based on $Hits@N$.

The recommendation engine proposed in this paper works for multi-directional and mitigates the multi-relational issues. It is capable of giving recommendations for users as well as the items. Each entity in knowledge base is projected to a lower dimension vector of size k. Note that, using these lower dimensional embedded vectors, various analytical methods such as segmentation, prediction and classification could be performed. For instance, applications like grouping users have similar interests, demographic segmentation and genres segmentation could be performed in segmentation.

References

[1] A Merve Acilar and Ahmet Arslan. A collaborative filtering method based on artificial immune network. *Expert Systems with Applications*, 36(4):8324–8332, 2009.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[4] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[5] Kai-Wei Chang, Wen-tau Yih, Bishan Yang, and Christopher Meek. Typed tensor decomposition of knowledge bases for relation extraction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1568–1579, 2014.

[6] Luca Costabello, Sumit Pai, Chan Le Van, Rory McGrath, Nick McCarthy, and Pedro Tabacof. AmpliGraph: a Library for Representation Learning on Knowledge Graphs, March 2019.

[7] Yuanfei Dai, Shiping Wang, Neal N Xiong, and Wenzhong Guo. A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics*, 9(5):750, 2020.

[8] P Devika, RC Jisha, and GP Sajeev. A novel approach for book recommendation systems. In *2016 IEEE international conference on computational intelligence and computing research (ICCIC)*, pages 1–6. IEEE, 2016.

[9] Miao Fan, Qiang Zhou, Emily Chang, and Fang Zheng. Transition-based knowledge graph embedding with relational mapping properties. In *Proceedings of the 28th Pacific Asia conference on language, information and computing*, pages 328–337, 2014.

[10] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[11] Qi He, B Chen, and Deepak Agarwal. Building the linkedin knowledge graph. *LinkedIn Blog*, 2016.

[12] M Karthik. Semantic based recommendations from knowledge graph embedding. https://github.com/karthikmiriyala/karthikmiriyala-Knowladge-Graph-Embedding-for-recommendations-.git, September 2021.

[13] Prajyoti Lopes and Bidisha Roy. Dynamic recommendation system using web usage mining for e-commerce users. *Procedia Computer Science*, 45:60–69, 2015.

[14] Fengyuan Lu, Peijin Cong, and Xinli Huang. Utilizing textual information in knowledge graph embedding: A survey of methods and applications. *IEEE Access*, 8:92072–92088, 2020.

[15] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[16] Natasha Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-scale knowledge graphs: lessons and challenges. *Communications of the ACM*, 62(8):36–43, 2019.

[17] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[18] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[19] T Lekshmi Priya and Harikumar Sandhya. Matrix factorization for recommendation system. In *Advances in Artificial Intelligence and Data Engineering*, pages 267–280. Springer, 2021.

[20] GP Sajeev and PT Ramya. Effective web personalization system based on time and semantic relatedness. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1390–1396. IEEE, 2016.

[21] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.

[22] Amit Singhal. Introducing the knowledge graph: things, not strings. *Official google blog*, 5:16, 2012.

[23] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[24] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR, 2016.

[25] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

[26] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[27] Lina Yao, Quan Z Sheng, Aviv Segev, and Jian Yu. Recommending web services via combining collaborative filtering with content-based features. In *2013 IEEE 20th International Conference on Web Services*, pages 42–49. IEEE, 2013.

[28] Liu Zhiyuan, Sun Maosong, Lin Yankai, and Xie Ruobing. Knowledge representation learning: a review. *Journal of Computer Research and Development*, 53(2):247, 2016.