Enhancing Recommender Systems using Knowledge Graphs

INFO320 Essay

Anton Nydal

March 2023

1 Introduction

In 2020 Qingyu Guo, Fuzhen Zhuang, Chuan Qin, and Hengshu Zhu surveyed knowledge graph-based recommender systems and outlined three types of methods: embedding-based, path-based, and unified methods Guo et al. (2020). In this essay I will dissect examples from each category to discuss the intuition behind each category, and discuss how each category seeks to improve recommendations and mitigate the cold start and data sparsity problem.

1.1 Recommender Systems

The task of a recommender system (RS) is to give expert-advice in the absence of human experts. Typically, RS sort sets of items according to how well they match the preference or need of a given user. Such systems are a subtle part of modern life in more than one way. All content-based applications, e.g TikTok, YouTube, and Facebook, continually make judgements about what content to present to the users, advertisers identify relevant ads for a given time and person, and the news and streaming platforms use information about your interaction history and other users to present content you are likely to enjoy. These are some of the most common use cases, but in reality RS can substitute expert advice in any domain.

There are multiple approaches to RS. Collaborative filtering describe RS that rely on information from similar users to make recommendations, while content-based filtering recommend items based their similarity to other items enjoyed by the user. Combining the approaches yields a third, hybrid approach. No all-purpose RS exists and all approaches have their drawbacks. The problems of cold start and data sparsity has proven a challenge to most types of RS.

1.2 Cold Start and Data Sparsity

The cold start problem relates to the difficulty an RS experiencing when handling new users and items. A new item will not be recommended in a system

where interactions from similar users form the basis of new recommendations because the item will have recorded user interactions. Similarly, it becomes impossible to make personalized recommendations to new users because they have no history from which preferences are induced. The data sparsity problem arises when users tend to only interact with a small percentage of all available items. When overlapping interests among users are rare, the RS must adapt accordingly to perform well. This is especially problematic for collaborative filtering techniques. Guo et al. (2020) surveys how knowledge graph-based recommenders overcome these issues.

1.3 Knowledge Graphs

A Knowledge Graph (KG) is a type of Heterogeneous Information Network (HIN). In contrast to homogeneous information networks where entities belong strictly to one of two categories - entity or relation - HINs have multiple types of entities and relations. In a HIN, two entities, for instance Jacob and Mercedes, could exist in the same graph despite their different types, "human" and "automotive brand" respectively. Type different relations can also and co-exist in a HIN. For example, the Jacob-entity and Mercedes-entity could be connected via an a "employer"-relation, all the while Jacob is connected to Thomasin through a "father"-relation. In short, KGs are a type of HIN consisting of multiple interconnected "node-relation-node"-formatted triples.

Retroactively fitting a knowledge graph for a recommender system context requires little work. Because a prerequisite of RS are 1) items to recommend and 2) users to whom recommendations are made, a chosen entity type must be classified as user. Once an entity type is considered the "user type", the KG can be split into three sub-graphs. Each sub-graph will contain only triples in one of the three formats: user-relation-user, user-relation-item, or item-relation-item. This produces a user-graph, an item-graph and a user-item graph, a useful partition for embedding-based methods discussed in Guo et al. (2020)

2 Embedding-based methods

The intuition behind embedding-based methods is to identify and exploit key item similarities that the machine would otherwise ignore. As vehicles, the words "car" and "train" are semantically more similar than the words "car" and "whisper", though this is not obvious to a computer, to whom words lack semantic content. For a machine to take semantic content into consideration, items must be represented as vectors - a set of numbers describing a position in multi-dimensional space. Embedding is a method that push the vectors of semantically similar items towards each other in multi-dimensional space. This enables distance measuring to get a numerical value describing semantic similarity between two items.

Though there are many, the methods for generating knowledge graph embeddings (KGE) generally follow the same process:

1. Generate random values for all entity and relation vectors
2. Iterate the following until a stopping condition is reached:
3. Sample a batch of triples and for each triple in the batch create a corrupted triple. A triple is corrupted by substituting parts of it
4. The sampled triples and the corrupted triples are used to train the embedding by minimizing a loss function when assigning a probability that a given triple is in fact present in the knowledge graph
5. At the end, the learned embedding have extracted semantic content from the triples and can provide the closest n neighbours of an entity or relation Dai et al. (2020)

There are multiple instances of embedding algorithms, though they tend to differ mainly in how they meassure loss. Despite variation, the idea remains a constant: penalise low scores for true triples and high scores for false triples. Furthermore there are multiple variants of optimization techniques, whom minimize loss in different ways. Choice of algorithm matters, but they all result in n-dimensional representations of entities and relations from the graph. TransR is mentioned later as an instance of embedding algorithms.

Guo et al. (2020) Categorizes embedding-based methods for recommendation depending on how they handle the initial knowledge graph.

3 Embedding user-item interactions

One strategy is to embed the full graph, including user-item interactions. Here user preferences are captured relying on triples such as "item - belongs_to - user", "user - also_bought - item". In this way the embedding, through training, can give probability scores about whether a given triple, for example "Tom - rated_highly - Top Gun" is true Guo et al. (2020). The intuition is that the embedding procedure captures and preserves a semantic representation of user preferences well enough to make accurate predictions for new users.

3.1 Collaborative Knowledge Base Embedding

Collaborative Knowledge Base Embedding (CKE) for Recommender Systems addresses the data sparsity problem by including auxiliary information and leveraging heterogeneous data. The method uses three components to extract structural, visual, and textual item representation respectively. The components use TransR to embed the items' structural representation, and deep learning techniques to embed textual and visual presentation. The three embeddings are aggregated alongside an offset vector from the user-item interaction matrix to create an enriched item representation. To compare how well items match the preference of a user, compare the inner product of the user's vector and the item vector Zhang et al. (2016).

3.2 Deep Knowledge-aware Network

Deep Knowledge-aware Network (DKN) considers the click history of a user and candidate news as input and leverages knowledge-aware convolutional neural networks (KCNN) to predict click-through rate for the given user and candidate news. Candidate news and clicked news passes through the KCNN and the words in the news' headline are passed to a CNN where the combination of layers fuse word-embeddings, the entity embedding of words found in the item-graph, and the immediate neighbours of those entities. A traditional CNN layer followed by max-over-time pooling make up the last two layers of the KCNN. Using a KCNN, the final item representation capture meaning of its' words and relevant knowledge from the knowledge graph.

Using an attention network the DKN model determine user embeddings by aggregating the enriched representation of each item in their click-history. Recent clicks are given more weight, whereas old clicks are given less weight - the domain dictates that yesterday's news are less interesting than today's! When item representations have been enriched, and the user embeddings determined, the DKN model feed user embedding and candidate news embedding to a fully connected CNN layer to determine click-probability. Wang et al. (2018)

3.3 Further improvement with generative adversarial networks

Yang et al. (2018) suggests using Generative Adversarial Networks (GANs) to improve methods that enhance item and user representation. To make the knowledge transfer generative adversarial network (KTGAN) model, Yang and his colleagues first enhance the embedding of items - in their case movies - by incorporating embeddings from the Word2Vec and Metapath2Vec models. In this model user embeddings are the aggregate result of their favored movie's vectors. Then, a generator and a discriminator help to further refine representations. Tasking the generator with improvising the favorite movie of a user according, and the discriminator with distinguishing relevant from irrelevant user-move pairs, the KTGAN model enhances representations by minimizing loss for the generator and discriminator. This method is interesting as introduces a component whose goal is not to enrich, but to increase the precision of existing representations.

3.4 Summary of embedding-based methods

Embedding-based models create low-ranking representations of the knowledge graph and combines different types of side-information to further enrich the representation of items and users. The CKE model generate and combine embeddings for textual, visual, and structural content to create a representation that captures multiple implicit item features, The DKN model uses a KCNN to embed word-level information as well as information gathered from the knowledge-graph to create a knowledge-aware item representation, in addition to an at-

tention network that create user embeddings by aggregate embeddings from click-history emphasising recent clicks, whereas the KTGAN model use information from Word2Vec and Metapath2vec to refining item representations via the use of a generative adversarial network and a discriminator. All methods in this category utilize parts of the knowledge graph to extract representations which are later enhanced using various sources of side information.

4 Path-based methods

The intuition of path-based methods is to overcome, or mitigate, cold start and data sparsity by leveraging connectivity patterns in the knowledge graph. In general these methods rely on connectivity similarity of users and/or items to enhance recommendations Guo et al. (2020). To measure how similar the paths of two entities are, PathSim is used alongside a scoring function Sun et al. (2011).

Meta-path is a term that is often used in this context. It describes a distinct path in a graph. In a hypothetical graph, "Person - Married_to - Person" describe the meta-path of all married couples.

4.1 Hete-MF and CF

Hete-MF is a knowledge graph-based recommender system that uses connectivity patterns and matrix factorization to refine user and item representation, which in turn alleviates the data sparsity problem Yu et al. (2013). Whilst embedding-based methods use embedding algorithms, Hete-MF extracts a number of meta-paths and calculates item-item similarity in each path using weighted non-negative matrix factorization (NMF) to create low-rank representations of users and items. In Hete-MF NMF uses item-item similarity to regularize item representations, so that items with similar connection patterns will also be similar in their final representation. The final representations contain information about graph connectivity, and as discussed in the section for embedding-based methods, representations containing more information improves recommendations.

Alternatives to Hete-MF, such as Hete-CF include user-user similarity, item-item similarity and user-item similarity as regularization terms, which outperforms Hete-MF. These types of methods utilize path connectivity to enrich user and or item representation. One drawback of these methods is that they require domain knowledge to define meta paths Guo et al. (2020).

4.2 RuleRec

RuleRec is suggested as a solution to the problem of fine-tuning parameters. In short, RuleRec consists of two modules that complement each other: a rule learning module and a recommendation module. The learning module links items to entities in the knowledge graph, then learns the weight for each rule.

The recommendation module considers the rules and weights and use matrix factorization to generate recommendations. Basing recommendations on rules make recommendations more explainable and overcome the issue of having to define relevant meta-paths Ma et al. (2019).

4.3 Summary of path-based recommendations

Path-based models are similar to embedding-based models in that they both aim to enrich the representation of items and/or users. Path-based models generally integrate matrix factorization - or versions of it - with meta-paths to enrich representation. Examples of this is Hete-MF and Hete-CF. However, tuning paramters for these methods requires domain-specific knowledge. To combat this, models like RuleRec attempts to mine connectivity patterns, which are then used to enrich item representation. Though embedding-based models and path-based models differ in some key ways, they are not mutually exclusive. Unified methods is the last category outlined in the survey.

5 Unified methods

Unified methods attempt to leverage the intuition of both path- and embedding-based methods to improve recommendations. Unified methods are based on the idea of propagating embeddings through the graph, meaning entity vectors refined by a unified method will capture aspects of neighbouring entities' vectors as well. Propagation happens through aggregation, which boils down to combining numbers, or in this case, combining vectors.

Some terms that are relevant to these methods are multi-hop neighbours and ripple sets. The first term describes items that are reachable by N amount of hops. In the tiny graph "Thomasin - daughter_of - Jacob - employed_by - Mercedes" Jacob is a 1-hop neighbour, and Mercedes is a 2-hop neighbour, of Thomasin. Thomasin is her own 0-hop neighbour. A ripple set is a group of entities that are the head of all triples that are (K-1) multi hops away from a user or item. Each user and item has a ripple set for K = 1, 2, 3 ... n. Unified methods are divided into two groups.

5.1 Propagating user preferences

The idea of this group is to refine the user's representation based on their interaction history. First, embeddings are created for items. Second, for each user, extract ripple sets for k=1, 2, 3, ..., H. Filter out triples where there are no observed interaction between the user and the head of the triple. Concocting the remaining triples alongside a bias yields a representation of the user.

RippleNet expands on user preference propagation by exploring the potential interest of a user. Like ripples in water, the user preference aggregate outwards iteratively and automatically. Interestingly, while the initial center of the ripple is the user, RippleNet recursively set h-hop neighbours as new "ripple centers".

This way, the user preference is propagated in multiple iterations, spreading further and increasing its reach.

5.2 Propagate item representations

This group of unified methods focus on refining item representation by aggregating the embeddings of an item's multi-hop neighbours. The survey mention multiple aggregators like sum, concat, neighbour, and bi-interaction, though there are more.

Knowledge Graph Convolutional Network (KGCN) is a method for refining item representation. It samples a batch of H-hop neighbors, and aggregate their embeddings with the embeddings the (H-1)-hop neighbors. Following this pattern, the KGCN method work inward, towards the candidate item. By the end, the candidate item representation contains aggregated information from its original representation and information from neighbours.

5.3 Summary for unified methods

KGCN and RippleNet are similar in that they rely on information from neighbors to improve recommendations. The connectivity of the graph is captured by the links through which semantic embeddings are propagated.

6 Conclusion

Embedding-based methods overcome data sparsity and cold start by incorporating additional information about the items, such as semantic content. Path-based methods utilize the structure and interconnectedness of knowledge graphs to capture structural information, which in turn enriches representation of items and users. Lastly, unified methods propagate embeddings through connected entities in the graph, which results in item representations that contain both semantic content and information from neighbours.

References

Dai, Y., Wang, S., Xiong, N. N., and Guo, W. (2020). A Survey on Knowledge Graph Embedding: Approaches, Applications and Benchmarks. *Electronics*, 9(5):750. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., and He, Q. (2020). A Survey on Knowledge Graph-Based Recommender Systems. arXiv:2003.00911 [cs, stat].

Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., Ma, S., and Ren, X. (2019). Jointly Learning Explainable Rules for Recommendation with

Knowledge Graph. In *The World Wide Web Conference*, WWW '19, pages 1210–1221, New York, NY, USA. Association for Computing Machinery.

Sun, Y., Han, J., Yan, X., Yu, P. S., and Wu, T. (2011). PathSim: meta path-based top-K similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, 4(11):992–1003.

Wang, H., Zhang, F., Hou, M., Xie, X., Guo, M., and Liu, Q. (2018). SHINE: Signed Heterogeneous Information Network Embedding for Sentiment Link Prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 592–600. arXiv:1712.00732 [cs, stat].

Yang, D., Guo, Z., Wang, Z., Jiang, J., Xiao, Y., and Wang, W. (2018). A Knowledge-Enhanced Deep Recommendation Framework Incorporating GAN-Based Models. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1368–1373. ISSN: 2374-8486.

Yu, X., Ren, X., Gu, Q., Sun, Y., and Han, J. (2013). Collaborative Filtering with Entity Similarity Regularization in Heterogeneous Information Networks.

Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W.-Y. (2016). Collaborative Knowledge Base Embedding for Recommender Systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 353–362, New York, NY, USA. Association for Computing Machinery.