An Information Retrieval Framework for Contextual Suggestion Based on Heterogeneous Information Network Embeddings

Dominic Seyler*
IBM Research
dseyler2@illinois.edu

Praveen Chandar†
IBM Research
praveenr@spotify.com

Matthew Davis‡
IBM Research
matthew.davis@invitae.com

ABSTRACT

We present an Information Retrieval framework that leverages Heterogeneous Information Network (HIN) embeddings for contextual suggestion. Our method represents users, documents and other context-related documents as heterogeneous objects in a HIN. Using meta-paths, selected based on domain knowledge, we create graph embeddings from this network, thereby learning a representation of users and objects in the same semantic vector space. This allows inferences of user interest on unseen objects based on distance in the embedding space. These object distances are then incorporated as features in a well-established learning to rank (LTR) framework. We make use of the 2016 TREC Contextual Suggestion (TRECCS) dataset, which contains user profiles in the form of relevance-rated documents, and demonstrate the competitiveness of our approach by comparing our system to the best performing systems of the TRECCS task.

1 INTRODUCTION

Recent advances in HIN research [7] allow us to embed a HIN into a multi-dimensional vector space [6]. We present a method that facilitates these embeddings in an LTR framework for the problem of contextual suggestion. In this problem domain, we have contextual queries and user profiles that contain preference rankings of objects of interest. In our approach, we model users and objects in the same HIN and define meta-paths specific to the problem domain, then generate graph embeddings by randomly sampling the HIN conditioned on the meta-paths. Once users and their interests are projected into the resulting embedding space, we derive features that rank the objects of interest according to a contextual query.

Meta-paths enable us to capture human intuition by introducing a set of semantic constrains on the HIN. For example, if user U_1 tags a relevant document D_1 using word w, then another user U_2 that has used w to tag a different document D_1 might also find D_1 relevant. This information is inherently contained in the graph but hidden. A domain expert can utilize meta-paths to express her knowledge about this latent structure. The generation of the statistical representation of the graph is then guided by conditioning the sampling of the graph along nodes that follow the meta-path.

In this work we show how meta-paths can be utilized in an Information Retrieval setting. By representing users and the objects of interest (i.e. documents) in the same vector space we can make use of the distances between objects as features in a ranking function. The ranking function is then learned from a small sample of relevance judged documents (in our case the relevance judgments from previous years of the TRECCS task). Once trained, the ranking function can be utilized to re-rank a set of retrieved documents, thereby incorporating the latent information of the HIN.

To demonstrate a concrete application of the retrieval framework we have selected the trip recommendation problem of the TRECCS task [3]. The TRECCS dataset provides user profiles composed of a set of objects ranked by relevancy to the user along with the search intent (e.g. planning a trip). The goal is to return a ranked list of attractions that might be interesting to the user. As the objects of interest are represented by text documents, we investigate how document-based nodes can be broken up into fine-grained node types to improve the retrieval performance significantly. Further, we experimentally show that we can reduce sparsity in the graph by limiting the number of nodes prior to training, which results in significant performance improvements.

To summarize, the contributions of this paper are to: (1) Define a general IR framework for the ranking of heterogeneous objects based on HIN embeddings. (2) Identify node types and graph topology specific to the TRECCS task. (3) Specify meta-paths to encode domain knowledge in the embedding space. (4) Show how fine-grained modeling of document-based nodes can improve performance. (5) Compare how different feature selection methods can reduce graph size and sparsity, while improving performance.

2 PROBLEM FORMULATION

The task we are addressing is the problem of contextual suggestion. There, a user has previously rated a list of documents, which make up the user profile, and the goal is to recommend a new set of documents to the user for a contextual query. We focus on the TRECCS task [3] where the IR system is assisting a user in planning a trip to a target city. The input to the system is a list of requests (R) and user profiles (U), where user profiles are a list of rated attractions (preferences), gender and age.

$$input = \{R, U = \{info, pref\}\}$$

$$R = \{group, season, trip\_type, duration, location\}$$

$$info = \{gender, age\}$$

*Current affiliation: University of Illinois at Urbana-Champaign.
†Current affiliation: Spotify Research.
‡Current affiliation: Invitae.

SIGIR '18, July 8–12, 2018, Ann Arbor, MI, USA

ACM ISBN 978-1-4503-5657-2/18/07…$15.00
https://doi.org/10.1145/3209978.3210103

$$pref = \left\{(attraction, rating, tags)^{1,\ldots,k}\right\}$$

The output is a ranked list of attractions not in the preference list, ordered by their posterior probability conditioned on the user profile and request:

$$output = \{P(attraction|R, U)|\forall attraction \notin pref\}$$

Our approach addresses TRECCS's second phase, which is the main phase of the track. There, the system is given a list of attractions that were retrieved in the first phase of the track. Each participating system is then required to re-rank this list and output it to the user. A list of ratings is given by the task providers to determine the optimal ranking, which is required for IDCG computation. To evaluate performance of the system we follow the TRECCS task's specifications by using normalized discounted cumulative gain (NDCG@5), precision (P@5), and mean reciprocal rank (MRR).

Along with the request and user profile information the TRECCS task provides a collection of all attractions in the task (attractions that are part of the user profile and that are part of the retrieval corpus). The collection is a list of 1,235,844 attractions with meta-information (i.e. attraction id, city, URL and title). In addition, the task provides a web crawl of all attraction's web pages that are part of the TRECCS collection. The web crawls covers 77.93% percent of attractions or 956,437 web pages. These web pages are a mix of travel recommendation portals (20% foursquare[1], 16% yelp[2], 5% tripadvisor[3]) and attraction's home pages (59%).

3 APPROACH

In this section, we present our HIN-embedding-based framework for contextual suggestion. Networks which consider type information have been shown to outperform homogeneous network-based approaches when measuring similarity of objects [9]. Choosing meta-paths over these networks facilitates the information propagation between subgraphs composed of heterogeneous node types, which contain inherent contextual information. We propose to model users and their preference context as heterogeneous nodes in the graph. User defined meta-paths guide the creation of an embedding space, where distances between objects can be interpreted as similarity features that inform a ranking function.

3.1 Graph Modeling

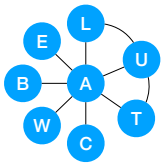
When applying the HIN framework to the domain of contextual suggestion we are faced with a number of choices: i) What are the node types in the graph? ii) Which topology will the node types adhere to? iii) Which meta-paths are most effective? We now elaborate on the modeling of the HIN that is used to incorporate the specific user profile information.

Node Types. We break down the contextual suggestion problem into semantic concepts, from which we derive the node types in the HIN. As part of user contexts, the TRECCS task provides information about users, locations, attractions, and user endorsements. We model each of these as different node types. In addition, TRECCS provides a collection of attractions, from which we utilize the business name as another node type. Along with the collection of attractions, TRECCS provides a web crawl of all attraction home pages, from which we derive words and named entities. Categories are extracted from the travel recommendation portals provided in the web crawl. Following Table 1, we eventually identify eight different node types that satisfactorily describe our problem setting.

Table 1: Node Types.

Node Type	Description
$\mathcal{U}$	User
$\mathcal{L}$	Location
$\mathcal{A}$	Attraction
$\mathcal{T}$	User tags/endorsements
$\mathcal{B}$	Token in attraction's business name
$\mathcal{W}$	Token on attraction's homepage
$\mathcal{C}$	Category tags from attraction's profile page
$\mathcal{E}$	Named entities in attraction's profile page

Figure 1: Network Topology.



Network Topology. Only a subset of node interactions are likely to be relevant to the problem, thus criteria for a modeling an interaction between nodes must be carefully selected. For the TRECCS task, we decided that attractions are the "hub" through which all node types connect to each other. Attractions link to users ($\mathcal{U}$) who rated them, to the location ($\mathcal{L}$) they are at, to the endorsements ($\mathcal{T}$) that were given by a user, to their business name ($\mathcal{B}$), to the words ($\mathcal{W}$) and named entities ($\mathcal{E}$) appearing on their web pages, and to the categories ($\mathcal{C}$) they belong to. Furthermore, users link to the location that they plan a trip to and the endorsements they made. Figure 1 depicts the network topology described.

Meta-Paths. To facilitate information propagation and guide the creation of the network embedding, our method requires the specification of meta-paths. Objects that have many path instances following a given meta-path will be closer in the embedding space than objects that have no such connection [6]. Furthermore, since nodes have types, it is possible to embed the similarity of objects of different types into the resulting vector space. Since we are interested in similarities of users and attractions, we decide to use them as start and end node types of all meta-paths. Table 2 lists all identified meta-paths and their semantic meaning. To create the shared embedding space, we leverage [6] as the graph embedding method of choice.

3.2 Learning to Rank Framework

We derive features from the vector representations of graph objects and the LTR framework we employed to learn a document ranking function. The vector space incorporates latent information about similarities of objects following different meta-paths. We leverage cosine distance as the similarity measure between the objects. We acknowledge that it would be worth investigating other similarity functions as well. However, the focus of this work is centered around investigating the usefulness of HINs for this problem. We

[1] http://foursquare.com
[2] https://www.yelp.com/
[3] https://www.tripadvisor.com

Table 2: Meta-Paths and Their Semantics.

Meta-Path	Semantics
$\mathcal{A}-\mathcal{U}$	Attractions were rated by a user.
$\mathcal{A}-\mathcal{T}-\mathcal{U}$	Attractions were tagged/endorsed by a user.
$\mathcal{A}-\mathcal{T}-\mathcal{A}-\mathcal{U}$	Attractions share tags/endorsements with other attractions that were rated by a user.
$\mathcal{A}-\mathcal{B}-\mathcal{A}-\mathcal{U}$	Attractions share business tokens with other attractions that were rated by a user.
$\mathcal{A}-\mathcal{W}-\mathcal{A}-\mathcal{U}$	Attractions share words on web page with other attractions that were rated by a user.
$\mathcal{A}-\mathcal{C}-\mathcal{A}-\mathcal{U}$	Attractions belong to the same category as other attractions that were rated by a user
$\mathcal{A}-\mathcal{E}-\mathcal{A}-\mathcal{U}$	Attractions mentioning the same entities as other attractions that were rated by a user

therefore argue that using a well established similarity measure, such as cosine, is sufficient at this point. Equation 1 defines the similarity for two nodes n_1, n_2 with vector representations $v^M_{n_1}, v^M_{n_2}$ for meta-path M.

$$similarity(n_1, n_2|M) = cos(v^M_{n_1}, v^M_{n_2}) = \frac{v^M_{n_1} * v^M_{n_2}}{||v^M_{n_1}||_2||v^M_{n_2}||_2} \quad (1)$$

Since each of the meta-paths capture different semantics (see Table 2) we decided to learn a parameter for each meta-path separately. Thus, the feature function f is defined over all meta-paths $M_i, i \in \{1...N\}$ for nodes n_1, n_2 as shown in Equation 2.

$$f(n_1, n_2) = \{similarity(n_1, n_2|M_i)\}, \forall i \in \{1...N\} \quad (2)$$

For the contextual suggestion task, we measure similarity between users $\mathcal{U}$ and attractions $\mathcal{A}$. In particular, the tasks requires ranking a set of candidate attractions $a_i \in A^{candidate}$ for a pair of request $r_i \in R$ and user profile $u_i \in U$. We then define our feature vector F for attraction a_i as in Equation 3.

$$F(a_i|r_i, u_i) = f(a_i, u_i), \forall a_i \in A^{candidates} \quad (3)$$

For training and testing, we calculate all feature vectors for all tuples of candidate, request, and user, and pair them with the relevancy information. We obtain the relevance judgments for the training phase from the 2015 TRECCS task. Then, we employ LambdaMART [10] to learn the ranking function and optimize for NDCG@5. Once the ranking function is learned, we score all candidate attractions of the 2016 TRECCS requests.

4 EVALUATION

4.1 Experimental Setup

As mentioned previously, we employ learning to rank and choose LambdaMART as our ranker. For this we use the RankLib 2.8 [4] implementation of LambdaMART and optimize for NDCG@5. We select 10% of the training data for validation. We set number the number of trees to 1000, the number of leafs for each tree to 10, the threshold candidates for tree splitting is set to consider all features. We set the minimum number samples each leaf has to contain to 1. Early termination is exercised after 100 iterations with no improvement on the validation data.

Due to the stochastic nature of the algorithm we expect some variance in the results. In order to provide fair and comparable results we report the average performance over ten identical runs. In addition, we report the variance over these runs in parentheses.

[4]https://sourceforge.net/p/lemur/wiki/RankLib/

Table 3: Fine-grained Representations of Attractions.

Node Types	NDCG@5
$\{\mathcal{U}, \mathcal{L}, \mathcal{A}\}$	.2400(±.0005)
$\{\mathcal{U}, \mathcal{L}, \mathcal{A}, \mathcal{T}\}$	.2565(±.0010)
$\{\mathcal{U}, \mathcal{L}, \mathcal{A}, \mathcal{T}, \mathcal{B}\}$	.2932(±.0006)
$\{\mathcal{U}, \mathcal{L}, \mathcal{A}, \mathcal{T}, \mathcal{B}, \mathcal{W}\}$	.2986(±.0003)
$\{\mathcal{U}, \mathcal{L}, \mathcal{A}, \mathcal{T}, \mathcal{B}, \mathcal{W}, C\}$	.3081(±.0004)
$\{\mathcal{U}, \mathcal{L}, \mathcal{A}, \mathcal{T}, \mathcal{B}, \mathcal{W}, C, \mathcal{E}\}$	**.3206(±.0003)**

4.2 Fine-grained Representation of Attractions

We now experimentally investigate whether a more fine-grained representation of attractions leads to better recommendations of attractions that are not previously rated by a user. This ties directly to how information propagates between different node types through the choice of different meta-paths. In general, one would expect that additional information in the network (i.e. node types) facilitates information propagation and thereby improving the overall retrieval performance. We show this in our experiments by incrementally adding new node types to the HIN. We then train and evaluate the HIN embeddings based on the meta-paths for each node type following Table 2. Table 3 presents the results.

The TRECCS dataset inherently is made up of three node types (users, locations, attractions). Using these node types as the most basic information network, we find that our approach performs with an NDCG@5 of 0.24. Adding user tags to the graphs increases the performance slightly to 0.2565. We can now observe a trend that with each additional node type more information is introduced in the HIN that was not present before, thereby increasing the overall performance. The best performance of 0.3206 is reached when all node types are present. This result might not surprise, but it shows that domain knowledge can guide the integration of additional information sources into the HIN, which greatly enhances the retrieval performance.

4.3 Reduction of Graph Sparsity

Since the number of nodes for certain types is quite large (the total $\mathcal{W}$ subgraph has over 1.5 million tokens), we run into various sparsity-related issues. For one, a high number of sparsely connected nodes prevents us from achieving a sufficient amount of

Table 4: Comparison of Different Feature Selection Methods and Cut-Off Points for Sparsity Reduction in the HIN.

Top-k	TFIDF	$\mathcal{X}^2$	MI
10	**.3309(±.0004)**	.2900(±.0003)	.3176(±.0005)
50	.3157(±.0004)	.3183(±.0004)	.2982(±.0003)
100	.3246(±.0003)	.3105(±.0005)	.3159(±.0005)
500	.3294(±.0001)	.2937(±.0005)	.2996(±.0007)
1000	.3163(±.0006)	.3135(±.0006)	.3079(±.0002)

sampling depth, since our algorithm is limited by an non-infinite number of samples per iteration. Secondly, dealing with web data introduces a certain amount of noise, which misguides the sampling of the space and negatively influences the resulting embeddings.

To counteract sparsity we compare multiple feature selection strategies, which allows us to reduce the graph to the most "expressive" nodes before training the embedding. We specifically investigate the case where nodes are derived from a natural language text, as was done for our $\mathcal{W}$ node types. We choose three common features selection strategies, namely (1) average TFIDF[8] weight, (2) chi-squared ($\mathcal{X}^2$) statistics between each non-negative feature and class and (3) mutual information (**MI**).

Results for this experiment are presented in Table 4 for different Top-k cut-off values. From the table we can see that a small top-k (i.e., 10) is able to not only reduce the size of the graph significantly but also improve performance simultaneously. We also find that unsupervised feature selection methods (i.e., TFIDF) seem to outperform the supervised features selection methods (i.e., $\mathcal{X}^2$ and **MI**) for almost all values of k.

One might wonder why the unsupervised method seems to almost consistently outperform the supervised methods. An explanation might be connected to the locality of the negative examples that are used by both methods. Since TFIDF statistics are computed on the entire document collection it might be better at estimating the overall amount of information a token contributes. On the other hand, supervised methods only consider documents with relevance judgments, which make up only a fraction of the collection. We leave the question of how supervised methods can be extended to incorporate more (unlabeled) documents for future work.

4.4 Comparison to other Systems

To put our work into context, We follow the TRECCS evaluation methodology and compare our system to the track's contestants as baselines. [4] addresses the TRECCS task by performing document analysis using a weighted kNN classifier and a query that was expanded using Rated Rocchio. To improve results they crawl additional information from travel portals such as Foursquare and Yelp. [5] makes use of word embeddings for both user profiles and candidate places. Here objects are represented by the sum of their word vectors. Their work ignores the inherent heterogeneity of the network. This work also extracts additional information from travel portals. [1] trains a binary SVM classifier for predicting the appropriateness of a venue. The approach also relies on external data sources which were manually created using crowdsourcing. [11] uses user-provided ratings and collaborative filtering to infer the missing rankings for attractions. This work also extracts additional information from travel portals. [2] uses word embeddings to build a neural document and a neural category preference model to re-rank attractions. This is the only top-performing system that does not utilize additional information sources other than those provided by TRECCS. As we can see our system is only outperformed by the debugged version of [4], which uses additional information from the web that was not provided by the TRECCS task.

Table 5: Comparison of TRECCS 2016 Task Runs. The Best Performance for the Five Highest-Ranked Teams are Shown. Systems are Ordered by NDCG@5 (Highest to Lowest).

System	NDCG@5	P@5	MRR
DUTH_knn (debugged) [4]	.3388	.4690	.6697
This work	**.3309**	**.4476**	**.6475**
Laval_batch_3 [5]	.3281	.5069	.6501
USI5 [1]	.3265	.5069	.6796
bupt_pris_2016_cs.2_.4_max [11]	.2936	.4483	.6255
UAmsterdamDL [2]	.2824	.4448	.5924

5 CONCLUSION AND FUTURE WORK

We presented an IR framework for contextual suggestion on HIN embeddings. Using the TRECCS task, we instantiated this framework and showed how domain knowledge can be encoded using meta-path guided embeddings as semantic and efficient representations. Furthermore, we showed how feature selection reduces graph sparsity and improves embedding quality.

Our method performs among the top of the state-of-the-art methods for the TRECCS task. The framework also proved effective on a person name disambiguation dataset, whose results were not shown due to spacial constraints. For future work we are planning to further investigate the generalization of our method on other datasets (e.g. MovieLens) and how supervised features selection strategies can benefit from training on unlabeled documents.

REFERENCES

[1] Mohammad Aliannejadi et al. 2016. Venue Appropriateness Prediction for Contextual Suggestion. In *TREC*.
[2] Seyyed Hadi Hashemi et al. 2016. Neural Endorsement Based Contextual Suggestion. In *TREC*.
[3] Seyyed Hadi Hashemi et al. 2016. Overview of the TREC 2016 contextual suggestion track. In *TREC*.
[4] Georgios Kalamatianos and Avi Arampatzis. 2016. Recommending Points-of-Interest via Weighted kNN, Rated Rocchio, and Borda Count Fusion. In *TREC*.
[5] Jian Mo et al. 2016. Word embeddings and Global Preference for Contextual Suggestion. In *TREC*.
[6] Jingbo Shang et al. 2016. Meta-Path Guided Embedding for Similarity Search in Large-Scale Heterogeneous Information Networks. *arXiv preprint arXiv:1610.09769* (2016).
[7] Chuan Shi et al. 2017. A survey of heterogeneous information network analysis. *IEEE Trans. Know. and Data Eng.* (2017).
[8] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* (1972).
[9] Yizhou Sun et al. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *VLDB* (2011).
[10] Qiang Wu et al. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* (2010).
[11] Danping Yin et al. 2016. Beijing university of posts and telecommunications (BUPT) at TREC 2016: A rating model based on tags for contextual suggestion. (2016).