Feature Selection on Heterogeneous Graph

Chun Guo  
School of Informatics and Computing  
Indiana University Bloomington  
Bloomington, IN, USA, 47405  
chunguo@indiana.edu

Xiaozhong Liu  
School of Informatics and Computing  
Indiana University Bloomington  
Bloomington, IN, USA, 47405  
liu237@indiana.edu

ABSTRACT

Heterogeneous graph based information recommendation have been proved useful in recent studies. Given a heterogeneous graph scheme, there are many possible meta paths between the query node and the result node, and each meta path addresses a hypothesis-based ranking function. In prior researches, meta paths are manually selected by domain experts. However, when the graph scheme becomes complex, this method can be inefficient. In this study, we propose feature generation tree, a novel feature selection method for heterogeneous graph mining based recommendation algorithms, which adds graph structure information into the original “feature selection for ranking” algorithm and saves a fair amount of time for feature computation. In our preliminary experiment, the proposed method outperforms the original “feature selection for ranking” algorithm in both efficiency and effectiveness.

Keywords
Feature Selection, Heterogeneous Graph Mining.

INTRODUCTION

Heterogeneous graph, in contrast to homogeneous graph, refers to an information network in which multiple types of nodes interact via different types of relationships. For example, in the music domain, we can model the context-rich heterogeneous graph as demonstrated in Figure 1. In this graph, 6 types of nodes (Artist, Album, Song, Playlist, Genre, and User) are connected via 16 types of relationships, e.g. Song is linked to the album it is published in via the includes relationship. If a user commented on a music track, there will be a comment relationship between the user node and the music node.

Figure 1. Scheme of a Heterogeneous Music Graph

Meta path based mining algorithms on heterogeneous graph have been proved useful in many recommendation tasks (Liu, Yu, Guo, & Sun, 2014, Guo & Liu, 2015). Meta paths are path patterns that describe how two types of nodes are connected on a heterogeneous graph. Different meta paths encode different semantics regarding the relatedness between the two types of nodes. Under the learning to rank framework, multiple meta paths can be combined together to achieve better recommendation performance than each individual one. In most past studies, the meta paths to be included in the learning to rank algorithm were manually picked by domain experts (Liu, Yu, Guo, & Sun, 2014, Liu, Yu, Guo, Sun, & Gao, 2014). In Guo and Liu’s work (2015), the top performing ones among a pool of meta paths automatically mined from the graph schema were used as the feature set. However, different meta paths, each treated as a unique feature in the learning to rank algorithm, might have dependencies between each other. Thus, more sophisticated feature selection method should be used to attain the most effective learning to rank model.

Comparing with machine learning, the objective for ranking is different. Rather than categorizing every instance into the correct class, ranking puts more focus on selecting the top-
in the meta-path based ranking functions. On the heterogeneous graph, meta-path defines how the query node and result node are connected, which is an abstraction of the many path instances that follow the same pattern. For the same recommendation task (e.g. recommend music tracks to a user), there are usually multiple possible meta-paths on the heterogeneous graph. For example, $U \xrightarrow{c} S$ is a simple meta-path, which denotes all the songs (S) that the user (U) has commented on. A meta-path can be represented as:

$A_Q \xrightarrow{R_1} A_1 \xrightarrow{R_2} A_2 \ldots \xrightarrow{R_{L-1}} A_{L-1} \xrightarrow{R_L} A_R$

where $A_Q$ is the query node type, and $A_R$ is the recommended node type. In the case of music recommendation, $A_Q$ is the user node, and $A_R$ is the music node.

Most previous research used manually selected meta-paths to solve different data mining tasks. However, there is no guarantee these selected meta-paths are the best performing ones. In Guo and Liu (2015), all possible meta-paths within a given length $L$ for a specific recommendation task were automatically mined from the heterogeneous graph scheme. When $L$ gets larger, more meta-paths will be generated, but the meta-path based features can be noisier.

Learning to Rank and Feature Selection
Each meta path provides different information regarding the connection between the query node and the result node. We use a learning to rank model $\Phi(P_1, P_2, \ldots, P_t)$ to combine multiple meta path based ranking features for better recommendation. Feature selection decides which meta paths to include in $\Phi$. The goal of feature selection is to find $t$ meta paths from all meta paths within a given length of $L$ that makes good recommendation under the learning to rank framework.

Feature Selection for Ranking
Our Feature Selection Method is inspired by the feature selection for ranking method proposed by Geng, Liu, Qin, and Li (2007). The authors evaluated the features for document ranking based on their importance and similarity. The importance is determined by standard information retrieval evaluation metric, and similarity is determined by the Kendall’s tau of two ranking list. In each iteration of the algorithm, the feature with the highest importance score is selected. Then the importance of all other features yet to be selected are punished by a value proportional to its similarity with the selected feature. The algorithm runs $t$ iterations to select $t$ features with maximum importance and minimum similarity.

Feature Generation Tree
As mentioned earlier, the calculation of meta path based ranking results is very time consuming, especially when the meta path gets long. Thus, we propose a feature selection method called feature generation tree, which tries to select the best meta paths by analyzing their internal structure. Before touching on the feature generation tree, we first introduce some notations.

Given a meta path

$MP = \dot{A}_Q \xrightarrow{R_1} \dot{A}_1 \xrightarrow{R_2} \dot{A}_2 \ldots \xrightarrow{R_{L-1}} \dot{A}_{L-1} \xrightarrow{R_L} \dot{A}_R$

of length $L$. If we cut the meta path into two parts at $R_{l+1}$,

the left part that starts with $\dot{A}_Q$ is a sub path of $MP$ which can be written as $SP = \dot{A}_Q \xrightarrow{R_1} \dot{A}_1 \xrightarrow{R_2} \dot{A}_2 \ldots \xrightarrow{R_l} \dot{A}_l$,

where $l$ is the length of the SP, and $l < L$. Meta path $MP$ has $(L-1)$ sub paths in total, each of length 1, 2, ..., $L-1$.

The assumption behind our feature selection method is: (1) Good sub paths lead to good sub paths of longer length, and eventually good meta paths; (2) Good sub paths are the ones that perform well finding node instances that are highly relevant to $v_R$ (node instance belonging to the result
node type $A_R$), and have smaller similarity with other sub paths ending with the same node type.

For example, in the music recommendation case, the meta path $U \xrightarrow{pl} A_r \xrightarrow{pln} Al \xrightarrow{} S^j$ has two sub paths: $U \xrightarrow{pl} Ar (SP_1)$, and $U \xrightarrow{pl} Ar \xrightarrow{pln} Al (SP_2)$. Following our assumption, if $SP_1$ locates good artists related to the ground truth songs, $SP_2$ has a higher chance of locating good albums related to ground truth songs. In the end, the meta path itself has higher chance of performing well in the Learning to Rank model.

In order to select the good sub paths, we apply the method proposed in Geng, Liu, Qin, and Li (2007) every time the pool of sub paths to be selected are expanded with sub paths of longer length. Following their notations, in each iteration of the algorithm:

1. We select the sub path with highest importance:

   $i_{select} = \arg \max \{imp(SP_j)\}$

2. Decrease the importance of each sub path $SP_j$ in the rest of the pool by an amount proportional to its similarity with $SP_{select}$:

   $imp(SP_j) = imp(SP_j) - 2c \cdot sim(SP_{select}, SP_j)$

   , where $c$ is a constant.

3. Remove $SP_{select}$ from the pool and select the next most important sub path based on the updated importance scores.

Nodes within two hops away from the ground truth song nodes are considered pseudo relevant nodes and used to evaluate the importance of sub paths.

Kendall’s tau is used to calculate the importance of each sub path and the similarities between sub paths. Importance is calculated as Kendall’s tau between the sub path’s ranking list and the pseudo ground truth ranking list. Similarity is Kendall’s tau between the ranking lists of two sub paths.

Given a pool of meta paths mined automatically from the graph scheme. Our goal is to find a subset of size $t$ that performs well in the learning to rank framework. We first evaluate sub paths of length 1 and select the $n$ best of them for each type of end node ($n \ll t$). Then we move on to evaluating sub paths of length 2. Please note that only the sub paths that contain the selected sub paths of length 1 are considered. After sub paths of length 2 are selected, we move on to sub paths of length 3 and so forth, until we’ve completed selection for sub paths of length $L-j$. In this way, we avoid calculating the intermediate and final ranking results for the meta paths that contain the sub paths discarded along the way.

**EXPERIMENT**

We tested our feature selection method in a music recommendation task. The heterogeneous graph scheme is demonstrated in Figure 1. For the task of recommending song to user, a total number of 259 meta paths that have a maximum length of 4 can be automatically extracted from the scheme. The goal is to select a good subset of 10 meta paths to build the learning to rank model.

**Data**

In order to build the heterogeneous music graph, we collected metadata and user generated data from xiami.com, a popular music social website in China. The website stores user-contributed and editor-curated metadata about different music entities (artist, album, song, etc.). It also allows users to build personal profile, record their listening history, create customized playlist, and socialize with other users within the community. For this study, the experiment dataset contains 56,055 artists, 43,086 albums, 1,233,651 songs, 633 genres, 677,275 users, and 305,916 playlists. We also collected a large number of relationships among different objects. There are 15,929,369 edges in total created on the graph.

For evaluation, we randomly selected 1,000 users from the dataset whose listening history contains more than 50 different songs. We set up this filter to make sure there are enough training/test instances for each user. Then 40% of the “play” relationships originating from each of the 1000 selected users are removed for evaluation purpose.

**Baseline**

We have two baselines in this experiment. The first is a naïve baseline where we simply select the top 10 performing individual meta paths. The second baseline is directly applying “feature selection for ranking” method proposed by Geng, Liu, Qin, and Li (2007) at meta path level.

**Preliminary Results**

Table 1 shows the experiment results for the top 5 performing meta paths and the learning to rank models. It is clear that all three learning to rank models outperform the best performing individual meta path. Different meta paths provide different information about the connection between the user and his/her preferred music. Combining them together helps make a better recommendation. Feature generation tree outperforms both baselines for MAP@5, MAP@10, and Precision@5. However, both feature selection methods performed worse than the naïve baseline on NDCG. Since Kendall’s tau only considers the number of concordant/discordant pairs, it does not take the order of the whole ranking list into full account. Errors occurring at

---

1 One of the largest online music streaming services (MSS) in China. Xiami.com has more than 20 million of active users.
any part of the list get equal penalty (Yilmaz, Aslam, & Robertson, 2008). It does not guarantee that the selected feature have a higher NDCG. In our future experiments, we will try substituting Kendall’s tau with NDCG in the importance calculation. However, we also notice that the feature selection for ranking method directly applied at meta path level has the worst performance among the three. One of the reasons is we randomly picked the parameters in this preliminary experiment. With more parameter tuning, we expect the evaluation measures to go up.

CONCLUSION
In this study, we proposed a feature selection method for heterogeneous graph mining based recommendation algorithm. In addition to considering the importance of and similarity between features, it leverages the graph structure to reduce the time required for feature computation and selection. In our preliminary experiment, the proposed algorithm outperformed the original “feature selection for ranking” algorithm. It did not beat the naïve baseline on NDCG measures, and we propose to fix the problem by substituting Kendall’s tau with NDCG in future calculation of feature importance. Other future directions are: parameter tuning, better ways to generate pseudo ground truth, and combing collaborative filtering into the learning to rank model to further improve recommendation performance.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>MAP@ 5</th>
<th>MAP@ 10</th>
<th>NDCG @5</th>
<th>NDCG @10</th>
<th>P@5</th>
<th>P@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual meta path</td>
<td>$U \xrightarrow{p_{l}} A_r \xrightarrow{p_{ln}} A_l \xrightarrow{i} S^?$</td>
<td>0.0017</td>
<td>0.0024</td>
<td>0.084</td>
<td>0.0852</td>
<td>0.1114</td>
<td>0.1135</td>
</tr>
<tr>
<td>Individual meta path</td>
<td>$U \xrightarrow{p_{l}} S \xleftarrow{p_{ln}} A_r \xrightarrow{p_{ln}} A_l \xrightarrow{i} S^?$</td>
<td>0.0014</td>
<td>0.0021</td>
<td>0.071</td>
<td>0.0717</td>
<td>0.096</td>
<td>0.0963</td>
</tr>
<tr>
<td>Individual meta path</td>
<td>$U \xrightarrow{p_{l}} S \xleftarrow{i} A_l \xleftarrow{p_{ln}} A_r \xrightarrow{p_{ln}} S^?$</td>
<td>0.0011</td>
<td>0.0017</td>
<td>0.064</td>
<td>0.0676</td>
<td>0.0894</td>
<td>0.0932</td>
</tr>
<tr>
<td>Individual meta path</td>
<td>$U \xrightarrow{b} A_l \xrightarrow{i} S^?$</td>
<td>0.0011</td>
<td>0.0016</td>
<td>0.0478</td>
<td>0.0414</td>
<td>0.0758</td>
<td>0.0648</td>
</tr>
<tr>
<td>Individual meta path</td>
<td>$U \xrightarrow{p_{l}} A_r \xrightarrow{p_{ln}} S^?$</td>
<td>0.0009</td>
<td>0.0015</td>
<td>0.0389</td>
<td>0.042</td>
<td>0.0576</td>
<td>0.0614</td>
</tr>
<tr>
<td>Learning to Rank</td>
<td>top 10 performing meta paths</td>
<td>0.0019</td>
<td>0.003</td>
<td>0.117</td>
<td>0.122</td>
<td>0.1528</td>
<td>0.1599</td>
</tr>
<tr>
<td>Learning to Rank</td>
<td>10 features selected by “feature selection for ranking”</td>
<td>0.002</td>
<td>0.0031</td>
<td>0.1112</td>
<td>0.1163</td>
<td>0.15</td>
<td>0.1544</td>
</tr>
<tr>
<td>Learning to Rank</td>
<td>10 features selected by feature generation tree</td>
<td>0.0021</td>
<td>0.0034</td>
<td>0.1131</td>
<td>0.1185</td>
<td>0.154</td>
<td>0.159</td>
</tr>
</tbody>
</table>

Table 1. Evaluation Results.

REFERENCES


