# CS224W Project Report:
# SimRank Based Restaurant Recommendation System

Yang Du, Shijie Liu, You Zhou

**Abstract**

*This project aims to design a restaurant recommendation system by analyzing the link structure of the network constructed from a set of user to business rated reviews. In particular, we will use SimRank to calculate the similarity between a pair of users and a pair of businesses, and use the calculated similarity to predict the rating for user business pairs. We trained SimRank on the yelp academic dataset, evaluated our approach against collaborative filtering and achieved higher link and rating prediction accuracy than collaborative filtering.*

## 1. Introduction

Recommendation systems have always been an area of active research in artificial intelligence these days. Most of the businesses nowadays like Amazon and Yelp utilizes recommendation systems to cater to various user needs and increase sale. For example on Amazon, while a user searches for an item, similar items will be displayed and the user has great potential to browse them and make purchase.

While traditionally methodologies like machine learning and collaborative filtering has been used to build recommendation systems, not much research has focused on finding similar structures in the graph representation of the products. Thus it comes to our mind to find similar items in the product universe by looking at a graph-structured base similarity metric called SimRank.

SimRank is a similarity metric 'that measures similarity of the structural context in which objects occur, based on their relationships with other objects' [1]. In our project, the main purpose is to develop a restaurant recommendation system based on a set of user to business reviews. We will do this in several steps. Firstly we will collect our data, construct the network and do some initial filter to get the main graph that we will do analysis on. After that we will calculate the SimRank similarity score between pair of nodes in our main graph. As the SimRank computation is expensive and resource consuming, we will use several heuristics and optimization to get an estimate of the SimRank scores within reasonable speed without losing too much accuracy. After that for a pair of user and business, we will develop an algorithm to predict the user's rating for that business. Similar to collaborative filtering, we will have two approaches namely prediction based on item-item similarity and prediction based on user-user similarity. We will evaluate our prediction by training the similarity scores on 70% of the total data and test the prediction rating on 30% of the total data. We will also implement item-item and user-user collaborative filtering so as to compare our accuracy with collaborative filtering.

## 2. Related Work

### 2.1. SimRank: A Measure of Structural-Context Similarity (Glen Jeh et al., 2002)

In [1], Jeh and Widom presented a metric based on a simple and intuitive graph-theoretic model that serves to estimate the similarity between pairs of nodes in the graph. The similarity between two pairs of nodes $a$ and $b$ in a directed graph is

$$s(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_i(b))$$

Simply put, it is the average of the similarity of the cross product pair of all the notes that points to nodes $a$ and $b$ respectively, times the decay constant $C$, which serves as a hyper parameter.
SimRank can also be applied to bipartite graphs as well. In this case there are two sets of similarity scores in the two sets of nodes respectively.

Moreover Jeh and Widom also presented the algorithm to calculate the similarity measures by using fix-point iteration, and proposed some pruning ideas to make the computation faster. The paper also presented an interpretation of the SimRank similarity measure as the meeting distance of two random surfers in the graph. The authors tested their algorithm on two datasets and concluded that the meature they propose 'confirms the applicability of the algorithm in these domains, showing significant improvement over simpler co-citation measures' [1].
This paper gives us the basic formulation of SimRank and provides a viable starting block for the computation of SimRank.

### 2.2. Accuracy Estimate and Optimization Techniques for SimRank Computation (Dimitry Lizorkin et al., 2008)

In [2], Lizorkin et al. analyzed the accuracy of fix point SimRank computation and presented three major areas of improvement for the fix-point iteration algorithm. The first method is selecting a set of essential node pairs based on some criteria to compute the similar scores iteratively, and leaving the rest of the pairs and not computing their similarity scores until the last iteration. The second method is to adopt a kind of dynamic programming notion to store the value of the partial sums to avoid repetitive computation, since for a node $a$, the same partial sum will be used in the computation of similarity with a number of other nodes. In this way we can reduce the time complexity of computation from $O(n^4)$ to $O(n^3)$. The last suggestion is to have a threshold value for similarity which is different each round, and similarity scores below this threshold value are treated as 0. It can be proved that the difference of similarity value to the true value each round will be below some constant number if choosing the threshold values correctly.
The paper experimented on generated graphs and the Wikipedia corpus, and compared the time of computation between the original method and method with the optimizations. It concluded that while the original method is $O(n^4)$, their method with optimization can achieve $O(n^3)$, while maintaining an accuracy smaller than 0.1. The authors also noted that accuracy can increase with smaller value of the decay factor, as they set $C = 0.6$.

This paper gives us some analysis of the accuracy and computation speed of the original SimRank computation in [1], and provides us with a way to speed up the computation.

### 2.3. GroupLens: An Open Architecture for Collaborative Filtering of Netnews (Paul Resnick et al., 1994)

GroupLens[4] is a collaborative filtering system for netnews. The paper introduced the system in detail and provides an architecture for evaluating the similarity between two users and news articles respectively. After calculating the similarities the project introduced a way to predict the rating of a user to a news article by computing the average deviation of a neighbour's rating from that neighbor's user-specific mean rating. This is done to take into account that users may be rating on different scales. GroupLens's deviation-from-mean approach can be shown as

$$R_{u,b} = b_{u,b} + \frac{\sum_{v \epsilon U} sim(u,v) * (R_{v,b} - b_{vb})}{\sum_{v \epsilon U} sim(u,v)}$$

Where $U$ is the set of users that are most similar to $u$ and have rated the article $b$. The project did not explicitly present their performance on the prediction scores but later research has proved it to be a very reliable score prediction architecture and it is commonly used.
GroupLens provides us with a way to predict the rating of a user to a business. We can evaluate our approach by comparing the prediction score with similarity scores calculated by collaborative filtering.

## 3. Data Collection and Graph Formulation

We are using the Yelp Academic Dataset, which consists of business and user data around 30 universities. To focus our scope on restaurant recommendation, we performed an initial filtering by keeping only businesses of category food or restaurant.
We set the users as a set of nodes, the businesses as another set of nodes, and the reviews as the edges linking from the users to the businesses. In this way we will construct a directed bipartite graph connecting the users to the businesses. Currently, a review is only present in our model if the review has a star rating higher than three. We chose to do this because we care about whether a user will enjoy a restaurant. We want to distinguish between good reviews (like) and bad reviews (dislike). A user in favor of a restaurant is now reflected by the presence of a connecting

edge. In the next step, we plan to incorporate both positive and negative reviews into the graph by assigning edge weights as the ratings of the corresponding review.
We then dropped all users who've rated less than three restaurants. Considering our goal of customizing recommendations for each user, the amount of data we have for those users are simply too little and thus not representative of the users' actual interest. We then calculated the largest weakly connected component of this graph to use as our model in concern. Note here, we are using 70% of the edges in this WCC as our training set, and the remaining 30% as test set.
By running the SimRank similarity algorithm on this WCC, we can obtain the similarity scores between a pair of users and a pair of businesses.
Below is a visualization of the constructed graph. Due to the scale of our complete graph, we randomly sampled 20% of the nodes before visualization. The green nodes represent the businesses and the pink nodes represent the users. (TODO:yd322)

## 4. Methodology

### 4.1. Data Preprocessing

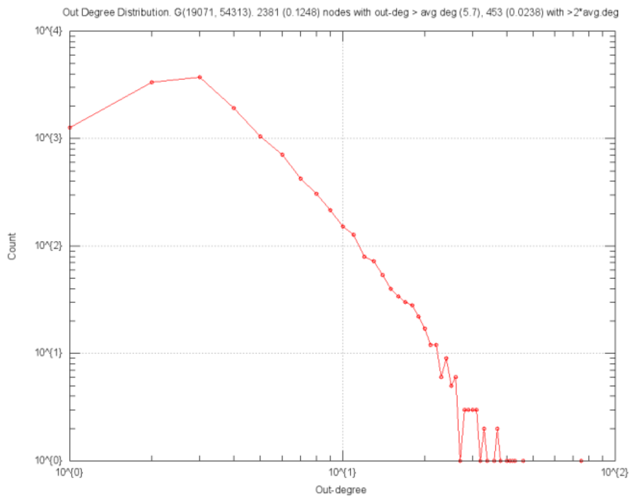
As the initial graph has hundreds of thousands of nodes and it is memory intensive to store the similarity score for every pair of nodes, we have to do some pruning to get rid of some of the irrelevant nodes. As we are doing recommendation, a lot of the nodes and edges can be removed.

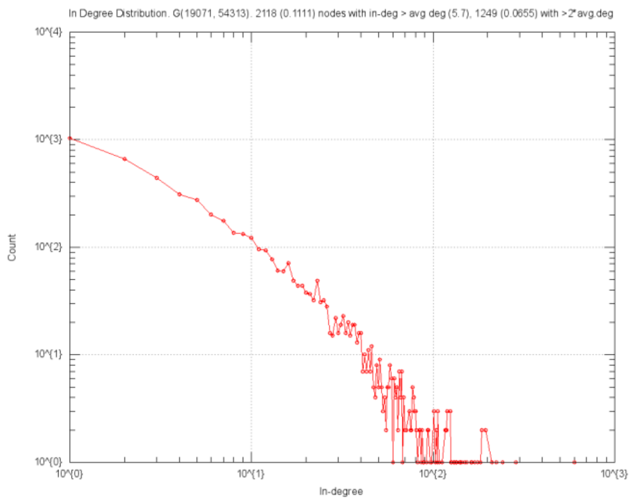
Firstly we notice that business nodes with low indegree and user nodes with low outdegree are very likely to be noise. This is because if a user has only rated 1 or 2 businesses, we do not have the sufficient information on what kind of business this user prefers and should not include that in our training set. The same can be applied to businesses. If a business has only been rated by 1 or 2 users, this business does not have enough information for us to evaluate and thus we should not train our similarity scores on them. Moreover as a recommendation system, we probably do not want to recommend businesses that has not been rated a lot of people. Hence in our initial graph, we removes business nodes with only 1 or 2 indegree, user nodes with only 1 or 2 outdegree, together with all the edges associated with them.

Secondly as we want to do recommendation, we want to find businesses that a user actually likes. Hence it is not ideal to include edges that have low ratings, which denotes that the user actually dislike the business. Thus in our graph we removed edges that have edge weights 1,2 or 3 and reserved edges with rating 4 or 5, so as to only operate on a graph with edges indicating that the user likes the business.

Lastly we notice that although most of the nodes are inside the biggest weakly connected component, there are some anomalies which do not. As the similarity scores between nodes in different components are always 0 and there are only a few nodes that are not in the largest weakly connected component, we only retain nodes that are inside the largest weakly connected component.

After these three filters our training set is now a directed, weakly connected bipartite graph with 4839 business nodes, 13721 user nodes and 54313 edges, which we computed the SimRank scores on.





## 4.2. SimRank Similarity Computation

We use the bipartite graph SimRank calculation presented by Jeh and Widom. In particular, as we have a directed bipartite graph from the users to the businesses, we let a pair of users be $u_a, u_b$ and a pair of businesses be $b_m, b_n$, then the basic SimRank computation is the power iteration of the two equations for each pair of users and businesses:

$$s_1(u_a, u_b) = \frac{C_1}{|O(u_a)||O(u_b)|} * \sum_{i=1}^{|O(u_a)|} \sum_{j=1}^{|O(u_b)|} s_2\left(O_i(u_a), O_j(u_b)\right)$$

$$s_2(b_m, b_n) = \frac{C_2}{|I(b_m)||I(b_n)|} * \sum_{i=1}^{|I(b_m)|} \sum_{j=1}^{|I(b_n)|} s_1\left(I_i(b_m), I_j(b_n)\right)$$

Where $s_1$ is the similarity between users and $s_2$ is the similarity between businesses. $O(u)$ denotes the set of business nodes that the current user node is pointing to while $I(b)$ denotes the set of user nodes that points to the current business node. Moreover the $C_1$ and $C_2$ are constants which we plan to use 0.5 for both. We need to compute the first equation between every pair of user nodes and the second equation between every pair of business nodes. Hence the total time of this computation can be $O(n^4)$ where $n$ is the number of nodes in the graph. This would be too time and resource consuming for our tens of thousands of nodes. Hence we need to do some optimization.

## 4.3. Optimization with Partial Sums

We used the optimization introduced in [2]. The main idea behind this optimization is that for a user node $u$ and a business nodes $b$, the quantity $\sum_{j=1}^{|I(b)|} s_1(u, I_j(b))$ is used in several computation of the business to business similarity while the quantity $\sum_{i=1}^{|O(u)|} s_2(O_i(u), b)$ is used in several computation of the user to user similarity. Hence for each pair of user and business, we can precompute the partial sums $P(u,b) = \sum_{j=1}^{|I(b)|} s_1(u, I_j(b))$ and $Q(u,b) = \sum_{i=1}^{|O(u)|} s_2(b, O_i(u))$ for every pair of user and business. Then our power iteration formulas become

$$s_1(u_a, u_b) = \frac{C_1}{|O(u_a)||O(u_b)|} \sum_{i=1}^{|O(u_a)|} Q(u_b, O_i(u_a))$$

$$s_2(b_m, b_n) = \frac{C_2}{|I(b_m)||I(b_n)|} \sum_{i=1}^{|I(b_m)|} P(I_i(b_m), b_n)$$

We note that in our precomputation, we need to do a summation over neighbors for every pair of user and business nodes, and in our power iteration, we reduces one loop and thus we can achieve a run time of $O(n^3)$ without losing any accuracy and be able to run with reasonable speed.

## 4.4. Collaborative Filtering Similarity

Collaborative filtering is another approach to calculating the similarity between a pair of businesses and users. In particular for two businesses, we can use the Pearson (correlation)-based similarity measure to get the similarity value:

$$sim(a,b) = \frac{\sum_{u \in U}(R_{u,a} - \overline{R_a})(R_{u,b} - \overline{R_b})}{\sqrt{\sum_{u \in U}(R_{u,a} - \overline{R_a})^2}\sqrt{\sum_{u \in U}(R_{u,b} - \overline{R_b})^2}}$$

Where $U$ is the set of users who have rated both businesses, $\overline{R_a}$ is the average rating for restaurant $a$ and $\overline{R_b}$ is the average rating for restaurant $b$. Moreover the similarity between a pair of users can be calculated similarly as:

$$sim(u,v) = \frac{\sum_{b \in B}(R_{u,b} - \overline{R_u})(R_{v,b} - \overline{R_v})}{\sqrt{\sum_{b \in B}(R_{u,b} - \overline{R_b})^2}\sqrt{\sum_{u \in U}(R_{v,b} - \overline{R_v})^2}}$$

Where $B$ is the set of businesses that have been rated by both users, $\overline{R_u}$ is the average rating for user $u$ and $\overline{R_v}$ is the average rating for user $v$.
We will use our SimRank similarities to compare with the CF similarities in terms of the two tasks described below.

### 4.5. Rating Prediction

After calculating these two similarity measures we can use them to predict the rating of a user to a business. We will have two methods, business-business based and user-user based. In particular for business-business based rating prediction, let $S$ be the set of businesses that are most similar to $b$ (This is the K-nearest neighbor approach) and the user has rated we can predict the rating to be

$$R_{u,b} = b_{u,b} + \frac{\sum_{c \epsilon S} sim(b,c) * (R_{u,c} - b_{uc})}{\sum_{c \epsilon S} sim(b,c)}$$

Where $b_{u,b} = \mu + \sigma_u + \sigma_c$ is the baseline rating, $\mu$ denotes the overall average rating, $\sigma_u$ denotes the rating deviation of user $u$ (i.e. average rating of the user minus $\mu$) and $\sigma_b$ denotes the rating deviation of business $b$.
Similarly for user-user based rating prediction, we can find the set of users $U$ that are most similar to $u$ and have rated the business, and calculated the prediction rating to be

$$R_{u,b} = b_{u,b} + \frac{\sum_{v \epsilon U} sim(u,v) * (R_{v,b} - b_{vb})}{\sum_{v \epsilon U} sim(u,v)}$$

Note that as we are using KNN, the cardinality of the set $S$ is a parameter that we can vary.
We can evaluate both approaches by comparing our predicted rating with the label ratings in the test set, and compute average deviation from the true rating, i.e.

$$Average\ Deviation = \frac{\sum |R^{predict} - R^{real}|}{number\ of\ test\ ratings}$$

## 5. Results and Findings

Using the parameters $C_1 = C_2 = 0.6$, we trained the scores using fixed point iteration for 10 iterations. Below is a table of the similarity scores between several businesses:

| | F | H | M | P |
|---|---|---|---|---|
| F | 1 | <0.0001 | **0.3039** | <0.0001 |
| H | <0.0001 | 1 | <0.0001 | **0.5999** |
| M | **0.3039** | <0.0001 | 1 | <0.0001 |
| P | <0.0001 | **0.5999** | <0.0001 | 1 |

The letters stand for four businesses: Two in West Lafayette: **M**arco's Pizza West Lafayette and **F**uzzy's Taco Shop. Two in Chapel Hill: **H**ot Dog & Brew and **P**anera Bread Chapel Hill. Note here each pair is from the same area. And we can observe from the matrix that the similarity scores are higher within the same area than across areas. Also, note here, since we are using the parameter C = 0.6, a similarity score of 0.5999 is nearly a perfect score. It makes sense since both **H**ot Dog & Brew and **P**anera Bread Chapel Hill are very affordable lunch places serving food such as Hot Dogs and Sandwiches. The simRank between **M**arco's Pizza West Lafayette and **F**uzzy's Taco Shop is not as high. They serve slightly different food: one pizza and the other Taco.
However, since they are located in the same area, and both serve good household food, their Simrank score is still pretty high.

We also calculated the collaborative similarity scores, and used the two similarity measures to predict the ratings in the test set. Note that in our SimRank similarity matrices, only nodes that are in the WCC have the similarity with other nodes. Hence we did an initial filter of the test set and only retains ratings such that both the user and the business are in the WCC. This leaves us with 66888 test examples. The average deviation $\frac{\sum |\mu - R^{real}|}{number\ of\ test\ ratings}$=1.041. Apart from the SimRank similarity based and CF similarity based rating prediction introduced above, we also evaluated uniform rating prediction as baseline where we set the similarity between any pair of items to be 1, and thus if we are using KNN, the rating is just the average of the ratings by K random users from the set $S$. We evaluated both business-business based approach and user-user based approach using 3NN, 5NN, 10NN respectively, and the result can be summarized below:

| K | 3 | 5 | 10 |
|---|---|---|---|
| Uniform Baseline | 0.6008 | 0.6031 | 0.6136 |
| CF | 0.4411 | 0.4407 | 0.4415 |

| | | | |
|---|---|---|---|
| Similarity | | | |
| SimRank Similarity | 0.2253 | 0.2102 | 0.2141 |

Table 1. Business-Business approach average deviation of rating prediction for Uniform Baseline, CF Similarity and SimRank Similarity

| K | 3 | 5 | 10 |
|---|---|---|---|
| Uniform Baseline | 0.7046 | 0.6725 | 0.6485 |
| CF Similarity | 0.5007 | 0.4941 | 0.4944 |
| SimRank Similarity | 0.2757 | 0.2786 | 0.2714 |

Table 2. User-User approach average deviation of rating prediction for Uniform Baseline, CF Similarity and SimRank Similarity

## 6. Observation and Discussion

Firstly we observe that unlike uniform baseline, our CF and SimRank similarity based score prediction is not very sensitive to the change of number of nearest neighbors K. This is expected because as the similarity measures indicate the likeliness between businesses and users, hence the performance will not change much once we predict the rating according to the few most similar businesses or users to the one being evaluated.

Another observation is that business-business based score prediction usually performs better than user-user based approach, for both CF and SimRank similarity based score prediction. This is mainly because users usually have more complex taste, as a user may be interested in a range of different types of restaurants, while businesses are much simpler.

Most importantly, from the average rating deviation we can see that the SimRank similarity score can perform much better result than CF similarity, at least in the task of score prediction. We believe that this is because SimRank similarity is a more complex similarity measure. In the computation of CF similarity, for example when computing similarity between two users, we only care about their rating on items that have been rated by both users while ignoring the items that has been rated by one user but not the other, though it may be similar to another item rated by the other user. But in SimRank similarity, we consider the cross product of businesses rated by each user and use the similarity between each pair of businesses. Our results show that SimRank similarity is a better similarity measure than CF based similarity, at least for the task of score prediction.

However our SimRank based rating prediction approach also has a few limitations. Firstly the computation is costly in both space and time. In order to calculate the similarity we need to keep a matrix that stores the similarity score between every pair of objects which is space consuming. Also even with the partial sum optimization our computation time is still in the order $O(n^3)$, which is costly for huge networks. For our dataset we spent around 15 hours training the 10 iterations. Moreover due to the way we formulate the graph and predict the ratings, we basically converted the original ratings to 'likes' (as we kept edges with rating 4 or 5). In this way there is some information loss in our computation. Ideally we can incorporate the ratings into our SimRank computation, by for example computing another set of similarity scores on a network of 'dislikes', i.e. only keeping edges with ratings 1 or 2. The intuition is that two businesses are similar to each other if they are disliked by similar users. Another way to incorporate the ratings is to have a weighted version of the SimRank computation:

$$s_1(A,B) = \frac{C_1}{\sum_{i=1}^{|O(A)|}\sum_{j=1}^{|O(B)|} w_1\left(I_i(a),I_j(b)\right)} * \sum_{i=1}^{|O(A)|}\sum_{j=1}^{|O(B)|} w_1\left(O_i(A),O_j(B)\right) * s_1(O_i(A),O_j(B))$$

$$s_2(a,b) = \frac{C_2}{\sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|} w_2\left(I_i(a),I_j(b)\right)} * \sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|} w_2\left(I_i(a),I_j(b)\right) * s_1(I_i(a),I_i(b))$$

Here the weights $w_1$ and $w_2$ are the weights associated with a pair of nodes, and we propose to be a function of the nodes weights and edges weights. For example we can set the weight as the inverse of the absolute difference between the two node weights, under the idea that restaurants with great difference in average ratings are less relevant to the similarity of the current pair and should be penalized. Thus the similarity becomes a weighted average of the similarity scores of the cross product of the neighbors of the two nodes, times the decay constant. There are all possible extensions to our project.

## 7. Conclusion

In this project we computed the SimRank similarity scores between businesses and users in the Yelp Academic Dataset, by formulating a network with users and businesses as nodes, reviews as edges. We used the partial sum optimization to enable our computation to have reasonable speed. We developed an algorithm to predict the user-business ratings and compared with approaches based on collaborative filtering. We concluded that our SimRank similarity based rating prediction can achieve much better result than collaborative filtering, at least for the task of rating prediction.

## References

[1] G. Jeh and J. Widom. SimRank: a measure of structural-context similarity. In KDD '02: Proceedings of the eighth ACM SIG-KDD international conference on Knowledge discovery and data mining, pages 538-543. ACM Press, 2002.

[2] D. Lizorkin, P. Velikhov, M. Grinev and D. Turdakov. Accuracy Estimate and Optimization Techniques for SimRank Computation. In VLDB '08: Proceedings of the 34th International Conference on Very Large Data Bases, pages 422—433, 2008.

[3] Mitsuru Kusumoto, Takanori Maehara, and Ken-ichi Kawarabayashi. 2014. Scalable similarity search for SimRank. In Proceedings of the 2014 ACM SIG-MOD International Conference on Management of Data (SIGMOD 2014), pages 325–336, 2014.

[4] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work, pages 175–186, 1994