

Community Detection & Network Analysis for Beer Recommendations

Nicole Crawford and Soo Cho

{nicolecr, soocho}@stanford.edu

CS224W Project — Fall 2015

1 INTRODUCTION

Recommendation systems with expert-level results are difficult to implement. Often, there are many different features that can be taken into account, and it is hard to know what, if anything, is relevant. Using reviews from BeerAdvocate, we suggest that a network structured such that beers are connected if they have words from their reviews in common will create valuable recommendations. We propose that communities constructed from review features could partition the graph in such a way that these communities could be used to provide strong recommendations for new items. Secondly, we construct a bipartite network of beers and reviewers to build a strong recommendation system using Belief Propagation and Collaborative Filtering.

2 RELEVANT WORK

1. From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise Through Online Reviews (McAuley, Leskovec) [1]

In the paper “From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise Through Online Reviews,” McAuley and Leskovec analyze the results of a time-sensitive recommendation system on a network of beer reviews. They compare the results of an algorithm that takes into account the evolution a user’s tastes versus a static algorithm that makes recommendations as though all reviews occurred at the same point in time.

However, while McAuley and Leskovec do compare their proposed algorithm to a static algorithm that uses communities for recommendations, they do not compare their algorithm to one that has pre-constructed communities based on a particular feature of the network. We propose that whether strong

communities based on particular beer features should be studied in further detail.

2. Evaluation of Item-Based Top-N Recommendation Algorithms (G. Karypis) [2]

This paper examines item-based recommendation algorithms. It states the disadvantages of user-based Collaborative Filtering method: the computational complexity of this method grows linearly with the number of users, which is cumbersome in large-scale data. The author explores item-based recommendation techniques, which analyze the user-item matrix to find relationships between different items, and use these relations to predict users’ preferences and find recommendations for them. He shows that these algorithms are up to 28 times faster than the user-based methods, and the prediction quality was up to 27% better.[2]

Although the author makes a strong point about the item-based recommendations, it is important to note that recommenders scale with the number of users or items they must deal with, so there would be various scenarios in which each type can perform better than the other. Moreover, the item-based recommendation model still has $O(n^2m)$ runtime where n is the number of items and m is the number of users. This is because we would need to calculate $m(m-1)$ similarities, each taking up to n computations. Thus, this method is still not good enough for large-scale system. We thus explore the Belief Propagation Model, which can be applied using a bipartite network of users and items. This model has a linear time complexity, which makes it very attractive in large-scale dataset.

3 DATA

Our data comes from BeerAdvocate (<http://snap.stanford.edu/data/web->

BeerAdvocate.html), an online community of beer reviews. Each detailed review contains the beer’s name, id, style, alcohol by volume (ABV), the reviewer’s username, as well as the review score for four different categories: aroma, palate, taste, and overall. This dataset included 1,586,614 reviews for 66,055 beers across approximately 104 styles.

4 METHOD

4.1 Community Detection

4.1.1 Network Construction

We used three different community detection algorithms: Newman’s modularity, Clauset-Newman-Moore (CNM), and the Louvain method. We first construct a bipartite network represented as an adjacency matrix with weighted edges between beers (node type b) and review

$$\frac{1 + \log(t_{i,j}) \times \log \frac{N}{df_i}}{\sqrt{w_1^2 + w_2^2 + \dots + w_M^2}}$$

words (node type w). The edge weights are the tf-idf scores of the words in each beer’s set of reviews. The equation for tf-idf weighting used here is:

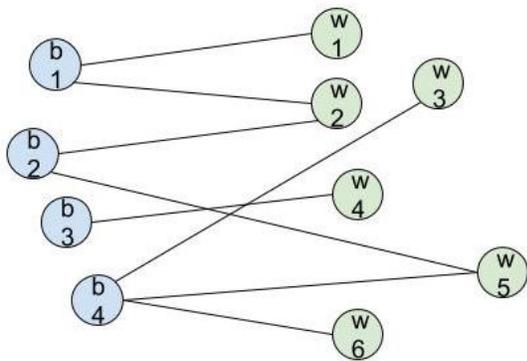


Figure 1: Example of Beer-Word Bipartite Network

We implement a comparative-level threshold such that all beer-word pairs with tf-idf score under 0.35 are removed from the bipartite graph. The resulting adjacency matrix is multiplied by its transpose to create an adjacency matrix where the nodes are beers, and the edges are weighted with the scores of the review words the two connected beers have in common. The resulting network has 7,767 reachable nodes and 44,212 edges.

4.1.2 Modularity of Style Matrix

One of the differentiating beer features is style, in that each beer in the dataset belongs to exactly one style. We propose creating an adjacency matrix of styles from the beer adjacency matrix where an edge i,j exists with weight w if there are w edges between beers of style i and beers of style j in the beer adjacency matrix. We use Newman’s modularity equation,

defined as $Q = \sum_i (e_{ii} - a_i^2)$ on the style matrix to see whether beers of the same style have more textual similarity in their reviews than beers of differing styles.[3]

4.1.3 Clauset-Newman-Moore

As a comparison to the modularity of the beer network where the communities are pre-constructed, we use the snap.py Clauset-Newman-Moore (CNM) algorithm, an algorithm that partitions the network into communities such that the modularity is optimized, on the beer network. The goal of using this function is to find whether other communities based on the textual similarity of the beers can be made, and if so, what the defining feature of these communities are.

4.1.4 Louvain Method

To further compare our style modularity to other communities that can be formed, we use NetworkX’s Louvain method on the network. This algorithm was used to see if different methodology for forming communities would result in vastly different partitioning or network modularity.

4.2 Review Rating Prediction

4.2.1. Baseline Model Used for Comparison

The BeerAdvocate Review dataset provided by beeradvocate.com contains the reviews of beers from wine enthusiasts and professionals. This dataset is used as a baseline model for comparison. Furthermore, because the dataset is too large having 66,054 beers total, for the purpose of this analysis, we have down-sampled the data when we apply the algorithms that we only consider beers that have received greater than or equal to 5 reviews. This narrows the sample size of the beer down to 19,793 beers total.

4.2.2. Belief Propagation Model

In this paper, we implement the Belief Propagation algorithm, which is an iterative probabilistic, message-passing algorithm, for beer recommender systems by using a network of beers and reviewers. Recommendations for each active user are iteratively computed through probabilistic message passing. It has been used for finding marginal distributions of the unobserved nodes, conditioned on the observed ones. By using a factor graph, we can obtain a qualitative representation of how the reviewers and beers are related on a graphical structure. An advantage of the Belief Propagation model is that it computes the recommendations for each user with linear complexity and without requiring a training period.[4]

For our beer review data, we applied Belief Propagation to a network of reviewers and beers. We aimed to make statistical inference about the ratings of users for unseen beers based on past data evidence. In Figure 2, we create a bipartite graph G on the set of reviewers and beers. Each reviewer reviews one or more beers, with a rating value between 0 and 5, and each beer is reviewed by one or more reviewers. Thus, we can draw two graphs, one for the reviewers and one for the beers, and create a bipartite graph to express the relationship between the two. The edge between a reviewer node and a beer node exists if and only if a reviewer in the reviewer node has reviewed a beer in the beer node where the edge connects to.

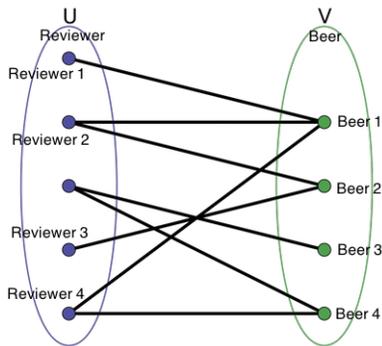


Figure 2: bipartite graph

The Belief Propagation algorithm is primarily composed of two parts. The first part of the algorithm is that each beer node that contains the value of an oracle beer rating receives the

rating that each reviewer indicates and then sends the offset of the reviewer back from the rating. In our bipartite graph, each beer node receives a number of reviews from a set of reviewers that review that beer. Thus, the beer nodes have directed incoming edges, which come from the reviewer nodes. Then, the reviewer nodes would be the transmitting nodes and the beer nodes would be the receiver nodes. In Figure 2, if the arrows of edges from u were to point to a node in v , which is a beer graph, then it would signify that the beer nodes are the receiver nodes.

The second part of the Belief Propagation algorithm computes the average offsets for the reviewer nodes and then offset their original rating by the computed average offset. In other words, the offset that is computed in the first part is sent back to the reviewers. This offset calculation uses the oracle value of each beer o_j , which was obtained by calculating the average of all the input values x_{ij} a beer j receives from reviewer i who reviews it.

Then, we subtract the oracle value of each beer from the rating of that beer by a reviewer i to obtain the offset value for the reviewer i , which we denote as $x_{ij}-o_j$. The offset value is calculated for every reviewer to calculate his average offset. The average offset encountered by reviewer i is equal to $p_i = |K_i|^{-1} \sum_j (y_{ji})$ where $j \in K_i = \{j: j_i \text{ in } G\}$. Then, the rating x_{ji} is adjusted to $x_{ji}-p_i$. This process is repeated until convergence: that is, until $p=[p_1, p_2, \dots, p_n]$ has a norm close to zero. The offset for reviewer i is thus the sum of the offsets p_i at each step of the iteration.

From Figure 3, which shows a difference between the original rating and the predicted rating from Belief Propagation vs. number of beers, we can see that a majority of the beers seems to have a difference of less than 1.

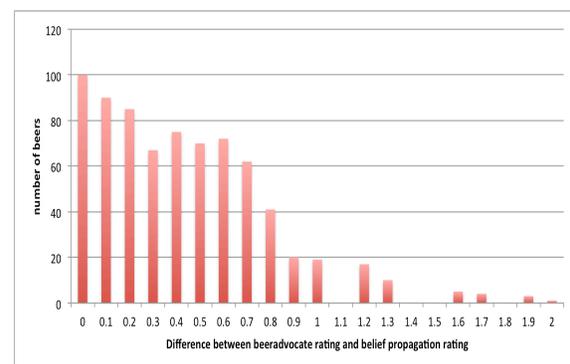


Figure 3: Histogram for BP model**4.2.3. User-Based Collaborative Filtering**

We then apply a method called Collaborative Filtering to build a full recommendation system for the beer reviewers. Among many ways of implementing collaborative filtering algorithms, we first implement a user-based filtering, which builds predictions based on the network of beer reviewers.

User-based Collaborative Filtering algorithm is implemented in a following way: we first aggregate information about which beers the user has reviewed and their review scores for each beer they reviewed. For each item the user has reviewed, our algorithm then chooses the top 5 neighbors that best match the user. In order to choose the top neighbors, we calculate a similarity score between the user and the others using the Pearson Correlation measure, and then pick the top 5 neighbors that have the highest Pearson correlation value with the user.[5] The basic idea behind those measures is that the more the users have similar tastes, the more they are next to each other in the preferences search space. Below is the formula for finding similarity between each user:

$$sim(i, j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_i)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_j)^2}}$$

where i and j are beer reviewers we compare and R indicates the review ratings of each user.

Then, based on this similarity score, we find two results: first, for each user, we recommend top 5 reviewers whom the user is most strongly connected to in terms of beer preference. Secondly, for each user, we produce a prediction of which beer the user might like by taking the weighted average of all the ratings. Thus, we can predict what rating a user will give to an unseen beer.

4.2.4. Item-Based Collaborative Filtering

Similarly, we implement item-based Collaborative Filtering method for the beer review sample in the following way: First, we analyze the user-item matrix to identify relationships between different items, and then use these relationships to indirectly compute recommendations for users.[5]

In the algorithm, like the user-based algorithm, the similarities between different items in the dataset can be calculated by using one of the similarity measures such as cosine-based, Pearson-based, and adjusted cosine similarity. For the beer review data, we will use Pearson-based similarity measure, which calculates how much the ratings by common reviewers for a pair of beers deviate from average ratings for those beers.[5]

Once we develop a model using Pearson-based similarity measure, we can predict the rating for any reviewer-beer pair by using the idea of weighted sum. First, we take all the beers similar to our target beer, and from those similar beers, we pick beers that the active reviewer has rated. We weight the reviewer's rating for each of these beers by the similarity between that and the target beer. Finally, we scale the prediction by the sum of similarities to get a reasonable value for the predicted rating. [5]

$$P_{u,i} = \frac{\sum_{\text{all similar items, } N} (s_{i,N} * R_{u,N})}{\sum_{\text{all similar items, } N} (|s_{i,N}|)}$$

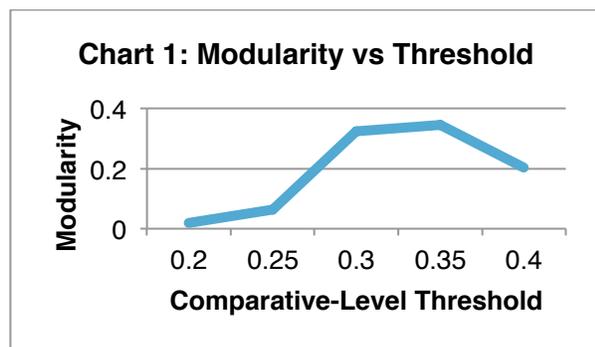
5 RESULTS**5.1 Community Detection****Table 1: Community Statistics**

Algorithm	Modularity	Number of Communities	Average Cluster Size
Style Matrix (all styles)	0.17	104	635.14
Style Matrix (10 styles)	0.35	10	2117.40
Louvain	0.95	1575	4.00
CNM	0.93	1583	4.91

Out of all the algorithms used, only the modularity of the style matrix using the top 10 styles had significant with an adequate modularity. The range of modularity is -1 to 1, where 1 is an extremely modular network, and 0 is random. With a score of 0.35, the style matrix of the top 10 styles is somewhat modular. Both the Louvain and CNM methods resulting in extremely small clusters for the dataset, meaning that these communities are not statistically significant. Additionally, the style matrix with all

styles used did not result in significant clustering, leading to the use of only the top 10 styles when looking for significant communities.

For the top 10 styles matrix modularity, the comparative-level threshold affects the results, as seen in chart 1. Having a threshold of 0.35 sets the modularity as optimal.



Further analysis conducted was on the words in reviews of the beers belonging to the top 10 styles to see if there were particular words that belong to each style and lead to the level of modularity of the network.

For each word in the reviews for the beers of each style, we took the count of the word in the style over the total count of the word in the dataset. The top word for each style can be seen in table 2, below.

Table 2: Unigram Likelihood of Top Word in Style

Style	Word	Probability
American IPA	48s	0.96
American Pale Ale (APA)	tepee	0.93
American Amber / Red Ale	sigs	0.94
American Porter	perseus	0.96
Fruit / Vegetable Beer	ephemeres	0.96
Hefeweizen	heinnieweisse	0.875
American Double / Imperial IPA	dipai	0.96
English Bitter	hsb	0.91

Euro Pale Lager	tatra	0.93
American Stout	babayaga	0.94

All the words in the table are a specific sub-style of that beer, or the name of a popular beer within that style. For example, *tepee*, the top word for the APA community in the network, is likely referring to the Stone Tepee Pale Ale. Therefore, we see that using the review words for beers does result in a modular network, as the beers connected are usually of the same, or similar style.

5.2 Rating Prediction

We created each test dataset by 80%/20% split of the full data into training and test dataset. We evaluate the rating prediction accuracy of 3 algorithms in terms of Root Mean Square Error (RMSE) metrics over the predicted ratings. We used the training data to predict the test data's ratings.

$$RMSE = \sqrt{\frac{1}{|K|} \sum_{i \in U, j \in I} (G_{ij} - \hat{G}_{ij})^2}$$

where $|K|$ refers to the number of ratings to be predicted in the test data, G_{ij} is the actual rating provided by user i for the item j in the test data, and \hat{G}_{ij} is the predicted rating by our algorithms.

The root mean-square error for the Belief Propagation method is 0.86, while that of the user-based collaborative filtering is 0.83, and that of the item-based collaborative filtering is 0.91. Among the three algorithms we used, RMSE was lowest using user-based collaborative filtering (see Table 3).

Table 3: RMSE of Algorithms Used

Methods	RMSE
Item-based collaborative filtering	0.91
User-based collaborative filtering	0.83

Belief propagation	0.86
---------------------------	------

6 CONCLUSION

First, the beer network with edges connecting textually similar beers is somewhat modular with respect to beer style; the words connecting the beers within a style usually have to do with a popular name or sub-style of beer within that particular style.

Second, we implemented the Belief Propagation model, which was used on a bipartite network of reviewers and beers, and compared it with user-based and item-based Collaborative Filtering models to make statistical inference about the user ratings for unseen beers based on the past observations. The Belief Propagation algorithm provides linear complexity, which makes it very attractive for large-scale systems. However, when compared with other popular methods that do not involve using a network, it did not lead to a better prediction of beer ratings.

7 REFERENCES

- [1] M. Julian, and J. Leskovec. “From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise Through Online Reviews.” *WWW*, (2013).
- [2] G. Karypis, “Evaluation of Item-Based Top-N Recommendation Algorithms”, (2001).
- [3] M. Newman and M. Girvan. “Finding and evaluating community structure in networks.” *Phys Rev E* **69**, 026113, (2004).
- [4] E. Ayday, A. Einolghozati, and F. Fekri, “BPRS: Belief Propagation Based Iterative Recommender System”, (2012).
- [5] X. Su, T. M. Khoshgoftaar, “A survey of collaborative filtering techniques”, (2009).