ALETHEIA
The Alpha Chi Journal of Undergraduate Scholarship

Volume 2 | Issue 2 | 2017

Combining Content Information with an Item-Based Collaborative Filter

Daryl Bagley

Harding University
Arkansas Eta Chapter

Vol. 2(2), 2017

Article Title: Combining Content Information with an Item-Based Collaborative Filter
DOI: 10.21081/AX0126
ISSN: 2381-800X
Key Words: algorithms, anime, collaborative filtering, content analysis, recommendation
This work is licensed under a Creative Commons Attribution 4.0 International License.
Author contact information is available from the Editor at editor@alphachihonor.org.

Volume 2 | Issue 2 | Fall 2017

Article Title: Combining Content Information with an Item-Based Collaborative Filter
DOI: 10.21081/AX0126
ISSN: 2381-800X

ALETHEIA
The Alpha Chi Journal of Undergraduate Scholarship

Combining Content Information with an Item-Based Collaborative Filter

Daryl Bagley

Harding University
Arkansas Eta Chapter

Abstract

This paper describes an attempt to use content information to improve an item-based collaborative filter for recommending anime. A recommendation engine is a system with the goal of determining how a user would rate an item in the database based on how he/she has rated previous items. Recommendation engines are classified based on the additional information they use to make predictions. Content information and social information are two common sources of information used by a recommender. Content-based recommenders use information about the items in the database and look at the similarity between items that the user has seen and the item it is trying to evaluate. Collaborative filters look at ratings from other users and find similarities between the user's ratings and other users' ratings in either a user-to-user comparison or an item-to-item comparison. This paper looks at a content-based filter, a user-based collaborative filter, and an item-based collaborative filter implemented to work in the domain of anime and compares that to a hybrid implementation that uses both content and collaborative information. As expected, this project found that the hybrid implementation showed an improvement over the item-based collaborative filtering implementation. However, item-based collaborative filtering was shown to perform worse in the domain than user-based collaborative filtering, which was the best implementation for the domain.

Key words: algorithms, anime, collaborative filtering, content analysis, recommendation

Introduction

Currently, there exist over 7,000 anime, including TV series, original animation videos, movies, and original net anime, with hundreds of new anime being produced each year [16]. For simplicity's sake, anime is often defined as animation made in Japan [31, 38]. "Anime is a form of visual culture and media, as well as a type of popular entertainment, a commercial product, an object of interest, and sometimes an obsession" [36]. Advances in information technology give individuals access to more information and media than ever before, creating an issue in recommendation and discovery: how does an anime fan decide what to watch next [18]? In the past, people relied on recommendations via word of mouth or published through the media [41], but since the early 1990s algorithms have been created that allow software to be able to fill this void, providing personalized recommendations based on large amounts of data [24, 33, 42].

The term "recommender system" describes "any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options" [8]. The goal of a recommender system is to determine items or products that may interest the users of the system [33]. In this paper, the terms "recommendation engine," "recommendation system," and "recommender system" have the same meaning and will be used interchangeably. The individual for whom the recommendation is made is known as the "active user," and the item for which a prediction is made is known as the "active item."

The goal of this project was to design and test a solution for the task of accurately predicting a rating for any given item in the data set. The design of a recommendation engine depends on the domain and the characteristics of the available data [33]. While large companies such as Netflix are able to gather vast and detailed amounts of data for large numbers of users, including when viewers pause, fast forward, or rewind; what devices they are using to stream; and what time of day they are watching [22], it is not always possible to make recommendations based on large amounts of implicit data, nor should users be required to give out large amounts of information to get accurate recommendations. In the anime industry, this is especially true, as a single individual may use several streaming sites, preventing any one service from getting an accurate portrayal of the user's preferences. Thus, the data used in this project is limited to what can be pulled from user-submitted sources and does not contain any metadata on the users. This allows users to remain completely anonymous. Instead, it contains lists of items (anime) for each user, along with the user's ratings (on a scale of 0.5 to 5.0, at intervals of .5) for the anime on the list.

The results of this project may be generalized beyond anime and applied to other areas with similar data sets. Furthermore, this project attempts to explain a viable but less common algorithm in addition to confirming existing knowledge of common algorithms.

The Domain

"A domain of recommendation is the set of items that the recommender will operate over, but may also include the set of aims or purposes that the recommender is intended to support" [11]. Typical domains include books, movies, TV, music, products, and documents. "One lesson that has been learned over the past years of recommender systems research is that the application domain exerts a strong influence over the types of methods that can be successfully applied" [10]. One of the unique aspects of this research is that the algorithms will be applied to an uncommon domain. The domain of anime is comparable to a combination of the domain of movies and the domain of TV but has some notable differences. Because the choice of a recommendation algorithm is dependent upon the domain, even the small differences exhibited by this domain can have a large impact on the applicability of a given algorithm.

Domains have six characteristics that can influence the choice and success of an algorithm: heterogeneity, risk, churn, interaction style, preference stability, and scrutability [11]. The domain used in this project has mid to low heterogeneity, meaning that the items in the database, while quite varied, all have similar purposes and satisfy similar goals. The domain has low risk, as a user does not lose much by accepting a recommendation. The domain also has low churn, as most items that are available at some point remain available for long periods of time. The domain has an explicit interaction style, as users are specifically searching for something to watch, and the primary means of determining the user's opinion on an item is through explicit ratings. The domain has fairly high preference stability, as users are not likely to experience drastic changes in opinion over short periods of time. Finally, the domain does not require much scrutability, meaning a user does not need much of an explanation for how the prediction was computed.

The Algorithms

The characteristics of the domain place certain constraints on the kinds of knowledge that a recommender system can use, while the availability and quality of knowledge influences what recommendation technologies a recommender can profitably use [11]. There are three broad categories of knowledge (individual, social, and content) that can be further expanded into subtypes of knowledge, and algorithms are classified based on what categories of knowledge they use [3, 11, 17].

While all algorithms use personal knowledge (knowledge about the active user), they vary in what subcategories they use. Because of the domain and available sources of knowledge, this project is restricted to the use of only opinion information from the active user. This eliminates many algorithms, leaving only collaborative and content-based algorithms as viable options. Collaborative algorithms rely on social knowledge (information about users other than the active user), while content-based algorithms rely on item features, a specific form of content knowledge.

To reach the aforementioned goals, several algorithms were implemented and compared against each other. These algorithms were user-based collaborative filtering, item-based collaborative-filtering, content-based filtering, and a hybrid algorithm involving the combination of content information with an item-based collaborative filtering algorithm. The implementation of common algorithms gives the hybrid algorithm suitable controls to be compared against, allowing for a better understanding of its performance.

User-based Collaborative Filtering

Collaborative filtering algorithms are by far the most researched and mature recommendation technologies [8, 24]. “Collaborative filtering makes predictions about an individual’s interests based on the interests of similar people” [6]. Collaborative filtering systems rely solely on a large database of past data for many users [33]. Neighborhood-based collaborative filtering is characterized by giving a recommendation based on the weighted combination of the ratings of a subset of users similar to the active user [33].

Collaborative filtering algorithms can be grouped into two classes: memory-based algorithms, in which user data is stored in memory to directly compare different users, and model-based algorithms, in which user data is analyzed and a model is derived for making predictions [7]. Memory-based algorithms, also known as neighborhood-based algorithms [28, 40], find users in the database with similar rating patterns to the active user, known as neighbors, and base the prediction on the ratings those users assigned to the active item. In contrast, model-based algorithms attempt to model the user-item interactions by determining factors representing latent characteristics of the users in the system [15]. While research has shown that state-of-the-art model-based algorithms are superior [27], this project implements a memory-based algorithm, for a couple key reasons.

Memory-based algorithms are simpler than model-based approaches: “In their simplest form, only one parameter (the number of neighbors used in the prediction) requires tuning” [15]. Compounding this is the fact that memory-based algorithms are efficient, requiring no costly training phase [15]. These factors make model-based algorithms easier to implement and more intuitive, which is why this type of algorithm was chosen for this project.

The algorithm computes a prediction for the active anime by comparing the active user to all the users in the database who have rated the active item to determine which are the most similar. Then the ratings given by the nearest neighbors are combined into a weighted average, which forms the prediction given by the algorithm. The weight used for the weighted average is the similarity. Preliminary testing showed that the optimal number of neighbors for the algorithm is 11. This means that only the ratings of the 11 users that are most like the active user will be used to determine the prediction.

Data. To implement a collaborative filtering algorithm, it is necessary to acquire a database of user ratings. This data was collected from Hummingbird.me, a free website that allows its users to keep track of what they have watched. The API provided by Hummingbird gives applications access to information about users’ libraries, which includes a list of anime rated by the owner of the library. Random users were pulled from the Hummingbird API, and their library information was saved to a text file to be read in at the beginning of the algorithm.

It was decided that the number of ratings in the database should be close to 300k, with the actual number being 299,968. Because anime is such a niche domain, the number of anime fans are relatively few but are quite devoted. Consequently, the domain for this project uses data where the number of items is much larger than the number of users. This is in contrast to the typical data set, which involves many more users than items [24]. There are 2698 users in the database, having rated, on average, 111 items each, and 7788 items in the database, having been rated 38.5 times each, on average. Thus, the sparsity of the data set is 0.9857, which is standard for a typical data set [24].

Similarity – Pearson correlation coefficient. The most common way of determining similarity (w) between the active user (a) and a given user (u) is through the Pearson coefficient, defined as:

$$w_{a,u} = \frac{\sum_{i\in I}(r_{a,i} - \bar{r}_a)(r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i\in I}(r_{a,i} - \bar{r}_a)^2 \sum_{i\in I}(r_{u,i} - \bar{r}_u)^2}}$$

where I is the set of items rated by both users, $\overline{r}_u$ is the average rating given by user u, and $r_{u,i}$ is the rating given to item i by user u [4, 40]. This equation compares the ratings of each user on all the items the users have in common. The output is a value between -1 and 1, with -1 representing complete disagreement, 1 representing complete agreement, and 0 representing no relation. The Pearson correlation coefficient also accounts for the fact that different users may be using different rating scales.

There have been other similarity measures tested in the past, including Spearman rank correlation, Kendall’s τ correlation, mean squared differences, entropy, cosine similarity, and adjusted cosine similarity [7, 23, 47]. Nevertheless, the Pearson coefficient remains the most popular method of determining the similarity between users and has empirically been shown to outperform cosine similarity [7, 23]. Thus, this project utilizes the Pearson correlation coefficient to determine the similarity between users.

In addition, while research has suggested that negative correlations are not valuable in increasing prediction accuracy [23, 24], this project uses the most highly correlated users, even if the correlation is negative. Because some items may not have many ratings, it was decided that it was not worth the risk to eliminate highly correlated users based on the correlation being negative.

Because it is possible for users to appear similar based on only a few items in common, the algorithm implements a Significance Weighting factor, which devalues correlations based on the number of co-rated items [23]. Preliminary testing showed that the optimal value for the number of co-rated items is 240. Thus, if two users have less than 240 items in common, the similarity is divided by 240 and multiplied by the number of co-rated items. If the two users have 240 or more co-rated items, the similarity is not modified.

Issues. The domain of anime may not be optimal for a collaborative filtering algorithm. A domain should have certain qualities to make collaborative filtering a good fit [44]. First, there may be issues with the data distribution. A suitable domain for collaborative filtering should have many ratings per item, often requiring more users than items [44]. However, this domain has a lot of niche items that are not rated by many users. While an average of 38.5 ratings per item is already low, compared to typical data sets [24], the data set used for this algorithm contains 3400 items with fewer than 5 ratings. This is in stark contrast to the 70 items in the database with over 500 ratings. Furthermore, the data set contains more items than it does users, and while this may not be indicative of the domain as a whole, it has the potential to be problematic.

Likewise, in a suitable domain for collaborative filtering, items should be homogenous; every item in the database should be identical by all objective criteria, differing only in subjective criteria [44]. The rationale for this is that collaborative filtering is the most valuable when evaluation of items is done in a subjective measure [44]. While this largely holds true in the domain of anime, the most notable exception is in the case of the length of the item. The database includes everything from movies that last roughly an hour to long-running TV shows that have hundreds of episodes.

One of the biggest issues concerning user-based collaborative filtering is scalability, the ability to continue functioning well when working with large data sets [28, 33]. To find the most similar users in the data set, collaborative filtering must check every user that has rated the active anime as well as every item both users rated. While the worst case scenario, in which every user has rated every item, nets this algorithm a complexity of O(NM), where M is the number of users and N the number of items, the algorithm's performance is closer to O(M + N) due to the fact that although the algorithm has to cycle through all of the available users, a large majority of those users will have rated only a few items, regardless of the number of items in the catalog [28]. Techniques for sampling are able to reduce the data set by a large factor in order to reduce the performance issues [21], but these techniques can reduce recommendation quality [28]. While the data set this project worked with is not as large as the sets used by major companies, there was still some lag in generating recommendations with this algorithm. However, no dimensionality reduction techniques were implemented, as the value of implementing one would not be worth the cost.

Item-based collaborative filtering

In 2003, item-to-item collaborative filtering was proposed as a solution to the scalability issues faced by user-based collaborative filtering when working with the tens of millions of users Amazon has [28]. The algorithm uses similarity between individual items, rather than the similarity between users, to make recommendations [28, 43]. As a model-based collaborative filtering algorithm, the ratings are used to compute and store the similarity offline, and the stored similarity calculations form the model with which predictions are made [43]. The similarity is computed by looking at the ratings given to the compared items by every user that rated both items [28, 33, 43]. While the worst case complexity is $O(N^2M)$, in practice, not all item pairs are actually seen, so the complexity is actually closer to O(NM) [28]. The complexity of the online component of the algorithm is dependent only on how many items the active user has rated [28]. In theory, off-line similarity computation is possible with user-based filtering, but because the number of overlapping items between any two users is generally small, a few ratings can make a large difference on the computed similarity; in contrast, the similarity between items remains fairly stable, keeping accuracy from being lost [43]. The predicted rating for an item can be calculated with a weighted average. This approach has been shown to lead to faster online systems [28] and improved recommendations [43], even when faced with limited user data [28], compared to user-based collaborative filtering.

Data. The item-based collaborative filtering implementation computed the item similarities from the same data used by the user-based collaborative filtering algorithm. 11,352,376 co-rated pairs were found, out of 30,314,790 possible pairs. The similarities were computed offline and each item got a separate text file of similarities so that the algorithm must only load one file at a time, instead of all computed similarities at the beginning of the program. However, as a result, every similarity was listed on both applicable files, doubling the amount of necessary reading in the case that every item is recommended.

Similarity – adjusted cosine similarity. Cosine-based similarity is a common measure of similarity in which two items are thought of as vectors with the number of users having rated both items as the number of dimensions [28, 43]. “In item-based recommendation approaches, cosine similarity is established as the standard metric, as it has been shown that it produces the most accurate results” [24]. The cosine between the vectors can then be computed and used as similarity. Adjusted cosine similarity takes into account the fact that not all users rate on the same scale by subtracting the corresponding user average from each co-rated pair [24, 43]. Thus, the similarity between items I and j is given by:

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

where U is the set of items rated by both users, $\overline{R_u}$ is the average rating given by user u, and $R_{u,i}$ is the rating given to item i by user u [24, 43]. As with the Pearson coefficient, similarity values produced by adjusted cosine similarity range from -1 to 1.

As with user-based collaborative filtering, this implementation includes a Significance Weighting factor to devalue correlations with a low number of co-rated items [23]. Preliminary testing suggested that this value be set to 80; after that returns greatly diminish.

Content-based recommendation

Content-based recommendation looks at the metadata associated with the media for a single user and finds similarities [33, 50]. To do this, the algorithm requires both extensive knowledge of what is being rated and a profile that describes the active user’s preferences [8, 24]. “The objects to be recommended need to be described so that meaningful learning of user preferences can occur” [10]. Unlike collaborative filtering, however, the algorithm does not rely on knowledge of what others have done in the past. This is significant because it allows content-based approaches to avoid both the new-item problem, in which a new, or obscure, item has no or few ratings and is unable to be recommended by a collaborative filter algorithm, and the issue of false information, in which fake profiles can (intentionally or unintentionally) influence recommendations [33].

In the simplest cases, the algorithm creates a search query to find other popular items based on the content of items the active user has previously enjoyed [28, 33]. There are two main problems with this simple form, the first of which is that the algorithm does not provide numerical predictions, but binary predictions. Furthermore, the algorithm does not scale well, providing either too many recommended items or reducing the data and providing poorer recommendations [28].

More complex algorithms involve assigning weight to different pieces of metadata to compare the shows the active user has rated to the active item to predict a rating for it. However, in systems like that, it becomes necessary to know which factors influence a user’s ratings the most. Often this information is gathered into the profile through something explicit, like a questionnaire that is completed before recommendations can be made [35]. The problem has also been solved through user profiling systems that pull information about the user from social media [26]. Without probing the user for more information, the recommender is left with only two options: the recommender can determine user influences through inferring from behavioral data, or the recommender can assume every user is the same and use weights that are predetermined by the programmer and hard coded into the software. This implementation relies on the assumption that all users are similar enough in their decision making and uses hard coded weights for the item features.

This project implements a nearest neighbor algorithm, also known as a lazy learner, in which the rating for the active item is determined from the ratings given to the set of most similar items [2, 20, 30, 37].

Data. Content-based filtering requires only the ratings of one user and the content information of the anime that user has seen. As with UBCF, the ratings for the active user were pulled from the Hummingbird API. A synopsis was also pulled from the Hummingbird API. All the content information, including a synopsis, was pulled from the Anime News Network (ANN) Encyclopedia API.

As needed, XML data for each anime was pulled from the ANN Encyclopedia API and saved offline. The file was then referenced whenever the anime appeared again. The XML data is parsed using TinyXML-2 by Lee Thomason.

At this point, it became clear that not every anime in each database could be used. Of course, if Hummingbird did not have an anime, it would not be referenced in ANN's Encyclopedia. However, if ANN did not have an anime, it could not be used in regards to Content-Based Filtering. To maintain consistency, these items were also ignored by the other algorithms. These "extra" items can be divided into three groups, based on why they had issues. The first group was not included by ANN because they were made in America, and so are not technically anime. The second group was removed because they were not full shows, but bonus content that ANN decided was not significant enough to be considered a separate show/season. The final group consisted of movie series that ANN groups, but Hummingbird separates.

In the case of the movie series, the first movie was mapped to the entire series on the assumption that if the movies are similar enough for ANN to group them, then they are similar enough that just getting the rating for the first one should be enough. The assumption was also made that everyone who has seen one or more of the movies in the series has seen the first movie.

Similarity – hybrid similarity approach. There are two forms that data can take: structured and unstructured [37, 49]. Structured data is represented by a well-defined set of features and corresponding values [30, 37, 46, 49]. Unstructured data is represented by unrestricted free-text, such as the contents of a document [5, 37]. Unsurprisingly, the items in the domain of anime contain both structured and unstructured data, "as they are represented by features as well as full-text descriptions" [49]. To take both the unstructured and the structured data into account, a hybrid similarity approach [49] was implemented.

This implementation computes the similarity between items by looking at 5 unique categories of metadata. The similarity between these aspects of the shows was computed and then combined with a weighted average. The aspects and corresponding weights, as determined through preliminary testing, are as follows:

- Cast – 16%
- Staff – 16%
- Companies – 16%
- Genres – 26%
- Synopsis – 26%

Similarities between structured categories (staff, cast, companies, and genres) are determined using the Dice Coefficient, a measure of the overlap between sets [19, 24]. If a category of structured data is defined as a set of terms, *S*, then the similarity between two items, *I* and *j*, is given by:

$$\frac{2 \times |S_i \cap S_j|}{|S_i| + |S_j|}$$

To compare synopses, term frequency vectors were generated. Stop words, common words such as articles and prepositions, were omitted from the term vector [24, 34]. The list of stopwords used is the default list from Ranks NL. Furthermore, terms were stemmed and reduced to their root word, via Sean Massung's implementation of the Porter2 English stemming algorithm [24, 34, 39]. These vectors were encoded with normalized term frequency, a technique used to minimize the impact that document length has when looking at term frequency [24, 30]. The similarity between synopses was computed via the cosine similarity (the most commonly used metric) of their document vectors [24, 30].

Issues. Content-based algorithms will have trouble when not enough metadata is associated with the items, or when the content is hard for a computer to analyze [33]. This likely holds true in regards to anime, as the metadata affiliated with the show isn't the same as the show's content. In practical settings, technical descriptions of an item, such as genres or creators, are usually readily accessible in electronic form, but subjective, qualitative features still remain a challenge for electronic assessment [24]. This is readily demonstrated by the Music Genome Project used by Pandora.com to provide content-based music recommendations for their listeners [24, 25]. In order to make these recommendations, trained analysts must attempt to identify which of 450 distinct characteristics are exhibited in each song, a task that takes them 20 to 30 minutes per song [1, 24]. “These attributes capture not only the musical identity of a song, but also the many significant qualities that are relevant to understanding the musical preferences of listeners” [1]. This is an example of the difficulty of applying content analysis to art. If it takes 20 to 30 minutes to analyze 3 to 5 minutes of audio information, then it would be unreasonable to suggest applying a similarly thorough analysis of anime, where the standard item is about 4 hours of both audio and visual information.

Finally, content-based filters lack the ability to give recommendations for items that the user may like that do not contain content from a user’s profile [33]. “It may be that listeners who enjoy free jazz also enjoy avant-garde classical music, but a content-based recommender trained on the preferences of a free jazz aficionado would not be able to suggest items in the classical realm since none of the features (performers, instruments, repertoire) associated with items in the different categories would be shared” [8]. Consequently, content-based filtering struggles to provide good recommendations for items that are unlike any the user has ever seen. When facing these issues, collaborative filtering algorithms are superior.

Hybrid algorithms

“From a knowledge source perspective, a hybrid recommendation is really a matter of combining knowledge sources that have not traditionally been put together” [11]. In order to get the benefits from both systems, many researchers attempt to combine Collaborative Filtering with Content Analysis [33].

A simple approach, sometimes known as a naïve hybrid, merely allows the individual methods to work separately and merges the results in a hybridization step [12, 13]. These algorithms fall into a category described as parallelized, consisting of three subgroups: weighted, mixed, and switching [8, 24]. The separate results can be combined using a weighted average that takes into account factors in which one algorithm is likely to perform better than the other [8, 12].

Another category of algorithms that require individual algorithms to work separately is described as pipelined and consists of the passing of output of one recommender to the next recommender as input [8, 24].

The final category of algorithms is described as monolithic; these hybrids “consist of a single recommender component that integrates multiple approaches by preprocessing and combining several knowledge sources” [24]. This group is further divided into two subgroups: feature augmentation and feature combination [8, 24]. These algorithms do not have to worry about the resource concerns that many, but not all, of the other algorithms do, as they are not reliant on running two separate algorithms [9]. In feature augmentation, pre-processing steps are applied by one recommendation technique to transform one form knowledge source into the knowledge source used by the primary recommender technique [8]. In contrast, feature combination does not convert one type of information to the other but uses both types of information to make a recommendation [8].

Content-boosted collaborative filtering is a good example of feature augmentation [8, 9, 24]. The algorithm combats the problem of sparsity common to collaborative filtering algorithms by using content analysis to determine the “missing” ratings for all of the users in a database, before using that database for collaborative filtering [32]. This is similar to the idea of default voting, the practice of assigning default values to items that have not been rated, in order to increase the overlap between users [7]; but rather than inserting the same value for every user, the algorithm personalizes the inserted value. Consequently, content-boosted collaborative filtering was shown to work better than either the naïve hybrid of the two methods or using either one alone [32, 33]. While it was suspected that this algorithm would work well in the domain of anime, the preprocessing step required to fill in the missing ratings is incredibly time-consuming.

This project tested a feature combination algorithm that combines item-based collaborative filtering with content analysis to solve the same problem addressed by content-boosted collaborative filtering. Traditional collaborative filtering relies on the ratings of users to determine the similarity between items [28], which causes problems for obscure items [33]. Even for popular items, the way similarity is determined plays a large role in the quality of the ratings [29]. Thus, this algorithm employed an item-based collaborative filtering design that relies on content-based similarity when collaborative filtering techniques are believed to be untrustworthy. This algorithm should improve the measured similarity between items, giving better results in general, especially for more obscure titles.

Similarity. This algorithm uses a combination of content-based similarity and item-based collaborative similarity based on the number of users that rated both items. Preliminary testing suggests that similarities computed with an overlap of 80 or above are reliable, while similarities computed with an overlap below 30 are especially unreliable. Using this information, a method was devised to utilize the similarity computed through content analysis to make up for possible unreliability in the collaborative-based similarity. This is formalized in terms of content-based similarity (s_{cb}) and collaborative similarity (s_{cf}) as:

$$S(n) = \begin{cases} s_{cb}, & n \leq 30 \\ \dfrac{(n-30)s_{cf} + (80-n)s_{cb}}{50}, & 30 < n < 80 \\ s_{cf}, & n \geq 80 \end{cases}$$

where n is the amount of overlap. The content-based and item-based collaborative similarities are computed with the same metrics described previously.

Rationale. Collaborative filtering algorithms are better suited for handling domains where subjective preferences come into play. Likewise, they work well when it is hard for the items to be analyzed by a computer or for metadata to capture the essence of an item. In contrast, content-based approaches often lack the ability to make a prediction for items that are unlike anything the user has previously seen. While item-based collaborative filtering could potentially fall victim to the same weakness, it bases similarity on whether users tend to rate the items the same way; thus, its similarity is not bound by conventional wisdom or intuition. Because of this, collaborative filtering should be superior in the domain of anime. Even so, collaborative filtering fails when handling items that are obscure or unpopular. For items like this, it is difficult for collaborative filtering to accurately gauge similarity. This is worse than determining with certainty that there are no similar items, as these items may be included in or omitted from the neighborhood inappropriately.

While item-based similarity seems to be the best option for the domain of anime, it lacks reliability in many cases. These cases can be adjusted by using content-based similarity. However, by not being solely reliant on content-based filtering, the algorithm avoids inheriting many of its weaknesses, including the inability to make predictions for items that are wholly dissimilar to the user's previously seen items.

Results

Algorithm	**MAE**	**NDPM**
User-based Collaborative Filtering	0.5689	0.2826
Item-based Collaborative Filtering	0.6101	0.3712
Content-based Filtering	0.6633	0.4702
Hybrid	0.6109	0.3853

To collect data, a list of users was compiled that exhibited certain qualifications. They had over 20 rated items with heterogeneity in ratings (i.e., ratings that weren't all the same), and these ratings were not included in the training data used for the collaborative algorithms or the set of users used for the preliminary testing. This set included 1377 users. Each user had 10 anime removed at random. A prediction was computed for each of those anime by each of the 4 algorithms. The mean average error (MAE) was computed for each user as seen in the table above. Similarly, the normalized distance performance measure (NDPM) was used to determine how closely the order of items given by the recommender matches the order given by the user [45, 48]. These values range from 0 to 1, with 0 being a perfect match between the user-produced ranking and the algorithm-produced ranking.

Case	**Users for MAE**	**Users for NDPM**
CBF outperforms IBCF	950	1084.5
UBCF outperforms IBCF	938	956
HYBRID outperforms IBCF	927.5	932.5
UBCF outperforms CBF	858	907.5
UBCF outperforms HYBRID	836.5	888
CBF outperforms HYBRID	N/A	738
$p < .001$		

Furthermore, to compare the overall performances of the algorithms, the number of users in which one algorithm outperformed another was counted. Using the sign test [14, 45], these performances were found to be significant with p=0.001.

Conclusions

The data does not support the hypothesis that using content information to improve item-based filtering would produce the most accurate predictions in the domain of anime. Instead, it suggests that user-based collaborative filtering is the best algorithm for the domain, while item-based collaborative filtering is the worst for this domain. Because the hybrid algorithm is based on an item-based collaborative filtering framework, the hybrid algorithm does not perform well. This outperforming of item-based collaborative filtering by user-based filtering suggests an error in the assumptions regarding the potential pitfalls the domain of anime lends to user-based collaborative filtering. It is likely that this error comes from the high item to user ratio. Because each anime has fewer ratings than each user has, it is easier to find similar users than similar anime. This leads to better performance by user-based collaborative filtering.

It is significant to note that the hybrid algorithm did perform better than the item-based collaborative filter. This supports the idea that combining content analysis with collaborative filtering will improve the predictions made by the collaborative filter. Along the same line, while the content-based algorithm outperformed both the item-based collaborative filter and the hybrid algorithm, overall it had larger errors and worse ranking ability than those two algorithms. This suggests that, while it can successfully outperform other algorithms in many cases, in cases where it fails, it fails worse than other algorithms. Thus, one should be skeptical of its utility.

Future work should attempt to exploit the large average number of anime rated by users through improved user-based collaborative filtering. This algorithm could potentially be improved by content analysis. Furthermore, the domain itself has more areas to be explored. Not only do users assign ratings to anime they have seen, but many users have information on whether they are currently watching the show, have completed the show, have stopped watching the show part way through, and what shows they plan to watch. This additional domain information is worth considering in future research.

Works Cited

[1] About The Music Genome Project®: *http://www.pandora.com/about/mgp*. Accessed: 2016-06-17.

[2] Adomavicius, G., Sankaranarayanan, R., Sen, S. and Tuzhilin, A. 2005. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems*. 23, 1 (Jan. 2005), 103–145. DOI:https://doi.org/10.1145/1055709.1055714.

[3] Adomavicius, G. and Tuzhilin, A. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. on Knowl. and Data Eng.* 17, 6 (Jun. 2005), 734–749. DOI:https://doi.org/10.1109/TKDE.2005.99.

[4] Amatriain, X., Jaimes, A., Oliver, N. and Pujol, J.M. 2011. Data Mining Methods for Recommender Systems. *Recommender Systems Handbook*. F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor, eds. Springer US. 39–71.

[5] Balabanović, M. and Shoham, Y. 1997. Fab: Content-based, Collaborative Recommendation. *Commun. ACM*. 40, 3 (Mar. 1997), 66–72. DOI:https://doi.org/10.1145/245108.245124.

[6] Boulton, J. 2014. Collaborative Filtering. *100 ideas that changed the web*. Laurence King Publishing.

[7] Breese, J.S., Heckerman, D. and Kadie, C. 1998. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (San Francisco, CA, USA, 1998), 43–52.

[8] Burke, R. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*. 12, 4 (Nov. 2002), 331–370. DOI:https://doi.org/10.1023/A:1021240730564.

[9] Burke, R. 2007. Hybrid Web Recommender Systems. *The Adaptive Web*. P. Brusilovsky, A. Kobsa, and W. Nejdl, eds. Springer Berlin Heidelberg. 377–408.

[10] Burke, R., Felfernig, A. and Göker, M.H. 2011. Recommender Systems: An Overview. *AI Magazine*. 32, 3 (Jun. 2011), 13–18. DOI:https://doi.org/10.1609/aimag.v32i3.2361.

[11] Burke, R. and Ramezani, M. 2011. Matching Recommendation Technologies and Domains. *Recommender Systems Handbook*. F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor, eds. Springer US. 367–386.

[12] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M. 1999. *Combining Content-Based and Collaborative Filters in an Online Newspaper*.

[13] Cotter, P. and Smyth, B. 2000. PTV: Intelligent Personalised TV Guides. *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence* (2000), 957–964.

[14] Demšar, J. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* 7, (Dec. 2006), 1–30.

[15] Desrosiers, C. and Karypis, G. 2011. A Comprehensive Survey of Neighborhood-based Recommendation Methods. *Recommender Systems Handbook*. F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor, eds. Springer US. 107–144.

[16] Encyclopedia - Anime News Network: *http://www.animenewsnetwork.com/encyclopedia/*. Accessed: 2016-05-11.

[17] Felfernig, A. and Burke, R. 2008. Constraint-based Recommender Systems: Technologies and Research Issues. *Proceedings of the 10th International Conference on Electronic Commerce* (New York, NY, USA, 2008), 3:1–3:10.

[18] Fields, B. 2011. *Contextualize your listening : the playlist as recommendation engine*. Goldsmiths College (University of London).

[19] Frakes, W.B. and Baeza-Yates, R. eds. 1992. *Information Retrieval: Data Structures and Algorithms*. Prentice-Hall, Inc.

[20] Gemmis, M. de, Iaquinta, L., Lops, P., Musto, C., Narducci, F. and Semeraro, G. 2010. Learning Preference Models in Recommender Systems. *Preference Learning*. J. Fürnkranz and E. Hüllermeier, eds. Springer Berlin Heidelberg. 387–407.

[21] Goldberg, K., Roeder, T., Gupta, D. and Perkins, C. 2001. Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Inf. Retr.* 4, 2 (Jul. 2001), 133–151. DOI:https://doi.org/10.1023/A:1011419012209.

[22] Gomez-Uribe, C.A. and Hunt, N. 2015. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Manage. Inf. Syst.* 6, 4 (Dec. 2015), 13:1–13:19. DOI:https://doi.org/10.1145/2843948.

[23] Herlocker, J.L., Konstan, J.A., Borchers, A. and Riedl, J. 1999. An Algorithmic Framework for Performing Collaborative Filtering. *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 1999), 230–237.

[24] Jannach, D. ed. 2011. *Recommender systems: an introduction*. Cambridge University Press.

[25] John, J. 2006. Pandora and the music genome project. *Scientific Computing*. 23, 10 (Aug. 2006), 40–41.

[26] Kanoje, S., Mukhopadhyay, D. and Girase, S. 2016. User Profiling for University Recommender System Using Automatic Information Retrieval. *Procedia Computer Science*. 78, (2016), 5–12. DOI:https://doi.org/10.1016/j.procs.2016.02.002.

[27] Koren, Y. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery*

and Data Mining (New York, NY, USA, 2008), 426–434.

[28] Linden, G., Smith, B. and York, J. 2003. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*. 7, 1 (Jan. 2003), 76–80. DOI:https://doi.org/10.1109/MIC.2003.1167344.

[29] Liu, H., Hu, Z., Mian, A., Tian, H. and Zhu, X. 2014. A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*. 56, (Jan. 2014), 156–166. DOI:https://doi.org/10.1016/j.knosys.2013.11.006.

[30] Lops, P., Gemmis, M. de and Semeraro, G. 2011. Content-based Recommender Systems: State of the Art and Trends. *Recommender Systems Handbook*. F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor, eds. Springer US. 73–105.

[31] MacWilliams, M.W. 2008. *Japanese Visual Culture*. M.E. Sharpe.

[32] Melville, P., Mooney, R.J. and Nagarajan, R. 2002. Content-boosted Collaborative Filtering for Improved Recommendations. *Eighteenth National Conference on Artificial Intelligence* (Menlo Park, CA, USA, 2002), 187–192.

[33] Melville, P. and Sindhwani, V. 2011. Recommender Systems. *Encyclopedia of Machine Learning*. C. Sammut and G.I. Webb, eds. Springer Publishing Company, Incorporated.

[34] Micarelli, A., Sciarrone, F. and Marinilli, M. 2007. Web Document Modeling. *The Adaptive Web*. P. Brusilovsky, A. Kobsa, and W. Nejdl, eds. Springer Berlin Heidelberg. 155–192.

[35] Mooney, R.J. and Roy, L. 2000. Content-based Book Recommending Using Learning for Text Categorization. *Proceedings of the Fifth ACM Conference on Digital Libraries* (New York, NY, USA, 2000), 195–204.

[36] Otmazgin, N. 2014. Anime in the US: The Entrepreneurial Dimensions of Globalized Culture. *Pacific Affairs*. 87, 1 (Mar. 2014), 53–69. DOI:https://doi.org/10.5509/201487153.

[37] Pazzani, M.J. and Billsus, D. 2007. Content-Based Recommendation Systems. *The Adaptive Web*. P. Brusilovsky, A. Kobsa, and W. Nejdl, eds. Springer Berlin Heidelberg. 325–341.

[38] Poitras, G. 2001. *Anime Essentials: Every Thing a Fan Needs to Know*. Stone Bridge Press.

[39] Porter, M.F. 1980. An algorithm for suffix stripping. *Program*. 14, 3 (1980), 130–137.

[40] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (New York, NY, USA, 1994), 175–186.

[41] Resnick, P. and Varian, H.R. 1997. Recommender Systems. *Commun. ACM*. 40, 3 (Mar. 1997), 56–58. DOI:https://doi.org/10.1145/245108.245121.

[42] Ricci, F., Rokach, L. and Shapira, B. 2011. Introduction to Recommender Systems Handbook. *Recommender Systems Handbook*. F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor, eds. Springer US. 1–35.

[43] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. 2001. Item-based Collaborative Filtering Recommendation Algorithms. *Proceedings of the 10th International Conference on World Wide Web* (New York, NY, USA, 2001), 285–295.

[44] Schafer, J.B., Frankowski, D., Herlocker, J. and Sen, S. 2007. Collaborative Filtering Recommender Systems. *The Adaptive Web*. P. Brusilovsky, A. Kobsa, and W. Nejdl, eds. Springer Berlin Heidelberg. 291–324.

[45] Shani, G. and Gunawardana, A. 2011. Evaluating recommendation systems. *Recommender systems handbook*. Springer. 257–297.

[46] Smyth, B. 2007. Case-Based Recommendation. *The Adaptive Web*. P. Brusilovsky, A. Kobsa, and W. Nejdl, eds. Springer Berlin Heidelberg. 342–376.

[47] Su, X. and Khoshgoftaar, T.M. 2009. A Survey of Collaborative Filtering Techniques. *Adv. in Artif. Intell.* 2009, (Jan. 2009), 4:2–4:2. DOI:https://doi.org/10.1155/2009/421425.

[48] Yao, Y.Y. 1995. Measuring Retrieval Effectiveness Based on User Preference of Documents. *J. Am. Soc. Inf. Sci.* 46, 2 (Mar. 1995), 133–145. DOI:https://doi.org/10.1002/(SICI)1097-4571(199503)46:2<133::AID-ASI6>3.0.CO;2-Z.

[49] Zanker, M., Gordea, S., Jessenitschnig, M. and Schnabl, M. 2006. A Hybrid Similarity Concept for Browsing Semi-structured Product Items. *E-Commerce and Web Technologies* (Berlin, Heidelberg, Sep. 2006), 21–30.

[50] 2014. *An Integrated Approach to TV and VOD Recommendations*. Red Bee Media Ltd.