Federal University of Rio Grande do Norte
Center for Exact and Earth Sciences

Bachelor in Computer Science

An empirical study of trope-based recommendation systems for animes

Ana Caroline Medeiros Brito

Natal-RN

November 2018

Ana Caroline Medeiros Brito

An empirical study of trope-based recommendation systems for animes

Undergraduate report submitted to the Federal University of Rio Grande do Norte as a partial requirement for obtaining the degree of Bachelor in Computer Science.

Advisor

Márjory Da Costa-Abreu, PhD

Federal University of Rio Grande do Norte – UFRN

Natal-RN

November 2018

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Setorial Prof. Ronaldo Xavier de Arruda - CCET

Brito, Ana Caroline Medeiros.
An empirical study of trope-based recommendation systems for animes / Ana Caroline Medeiros Brito. - 2018.
55f.: il.

Monografia (Bacharelado em Ciência da Computação) - Universidade Federal do Rio Grande do Norte, Centro de Ciências Exatas e da Terra, Departamento de Informática e Matemática Aplicada, Curso de Ciência da Computação. Natal, 2018.
Orientadora: Márjory Da Costa-Abreu.

1. Computação - Monografia. 2. Recommender systems - Monografia. 3. Hybrid systems - Monografia. 4. Tropes - Monografia. 5. Japanese animation - Monografia. I. Costa-Abreu, Márjory Da. II. Título.

RN/UF/CCET CDU 004

Elaborado por Joseneide Ferreira Dantas - CRB-15/324

Undergraduate thesis under the title *An empirical study of trope-based recommendation systems for animes* presented by Ana Caroline Medeiros Brito and accepted by the Department of Informatics and Applied Mathematics of the Center for Exact and Earth Sciences of the Federal University of Rio Grande do Norte, being approved by all members of the examining board specified below:

Natal-RN, November 2018.

For my parents.

Acknowledgments

I would like to express my thankfulness to my parents, my friends, and my professors. First of all, my parents, who were – and are – my first support in life. I'm really grateful for everything you did for me: every conversation, every advice, and every care.

All the moments shared with my friends during graduation helped me in this journey, I feel very lucky to have met them. In special, I would like to mention Esther Bárbara, Fernanda Isabel, João Pedro Holanda, Karyd'ja Souza, Lucas Torres, Luis Tertulino, Luísa Rocha – you suggested me one of the main ideas used in this work –, Luís Cláudio Rocha, Raquel Oliveira, and Ronaldo Silveira. I will always admire you and I hope we still have a lot of moments to live together.

Some friends who are with me since before I started my journey in Computer Science: Luan Lêdo, Mariana Leocádio, Morena Gil – your words were really important these days –, and Sofia Aravanis. Life sometimes distances us but I feel like we will always meet again.

Finally, I believe education is the way to open opportunities for everybody, the teachers have a very important role in this process. They helped me to develop reasoning and analytical skills. Special thanks to my advisor, Márjory, that always believed in me when I was doubting myself and helped me to see things in a simple way.

"Happiness only real when shared."

Christopher McCandless

An empirical study of trope-based recommendation systems for animes

Author: Ana Caroline Medeiros Brito

Advisor: Márjory Costa-Abreu, PhD

Abstract

Recommender Systems is a fundamental field of study considering the big volume of information and data available everywhere – when we need to choose something to buy, to use, to experience. These systems have the purpose of help users in the task of choosing or even just suggesting some services. This work is an overview of the main approaches used to make a recommendation and an investigation about how features present in stories – called tropes – can be helpful to increase prediction quality. There were tested seven algorithms, two of them using tropes. We have tested said algorithms with a new database composed of Japanese animations and tropes that is also a contribution of this work. Some interesting conclusions were obtained analyzing the results in the new database proposed and the other one already known – MovieLens enriched with tropes –, the testes with tropes disclose no substantial positive impact in predictions.

Keywords: Recommender Systems, Hybrid Systems, Tropes, Japanese animation.

List of figures

List of tables

List of Symbols

u_i	user i
i_j	item j
$P_{u,i}$	rating prediction to user u and item i
$s_{i,j}$	similarity between entities i and j (items or users)
$r_{i,j}$	rate of item j given by user i
$diff_{i,j}$	average difference in rating values between items i and j
λ	overall average rating

Contents

1 Introduction

Recommender systems are diffused in the daily lives of almost all people using digital technology – suggesting travel destinations, hotels, products on sale, restaurants, movies, and series.

In literature, we find "Recommender systems" defined as follow:

> Recommender systems are information filtering systems that deal with the problem of information overload by filtering vital information fragment out of large amount of dynamically generated information according to user's preferences, interest, or observed behavior about item. Recommender system has the ability to predict whether a particular user would prefer an item or not based on the user's profile (ISINKAYE; FOLAJIMI; OJOKOH, 2015).

It is remarkable the volume of information constantly growing in web-based systems, to provide this kind of advice is essential to keep users interest in the system and consequently keep the system running and generating profits (POLATIDIS; GEORGIADIS, 2013; LEE; HOSANAGAR, 2014).

This competition for attention reflects nowadays reality, we are surrounded by all kind of advertising on websites and mobile apps. There are lots of specialized services in offer tagging advertising inferred from user behavior – sites accessed, location, and so on (CUI et al., 2005; CHEN et al., 2014; RIBEIRO-NETO et al., 2005). For example, when we search for airline tickets once, we pass some days receiving advertising about it in our social networks (PURCELL; RAINIE; BRENNER, 2012).

Computer scientists interested in this problem have the challenge of finding out a precise way to extract user information or item features to make useful recommendations. Some contests are known in this field, it is the case of The Netflix Prize[1] in 2009, a good example of companies and scientists looking for the same goal. This initiative was really

[1]Accessed: Jun/2018. Available in: `https://www.netflixprize.com/`.

relevant to Netflix maintain their users active in the system once when users do not watch any movie or series it means not satisfied users. Other examples of well-known services using Recommender Systems are Amazon[2] with e-commerce, Facebook[3] suggesting "friends" and Spotify[4] in music domain.

1.1 Motivation

Everything we read, watch or listen is available in variety and quality. Recommender Systems studies are incited – in a general way – by the aspiration of to make always better and more precise recommendations. This dynamic can be very interesting once these systems help the users in the process of choose services to consume. On the other side, those who are offering the services need to ensure users attention or they will lose to the competing services – it explains the economic motivation behind this field of study.

A common feature present in every story is the use of narrative mechanisms defining its main elements (BATISTA et al., 2016). These "mechanisms" can be called tropes, somehow they are similar to tags in the way the users can categorize (GEMMIS et al., 2008). The use of tropes can be interesting even if limited only in the scope of items with narrative stories, once this set is relatively large: books, movies, series, comics, and also songs.

One first motivation to use the tropes is to explore how they can be useful in recommendation once there are just a few works using this kind of data (BATISTA et al., 2016). More than this, the tropes can be an alternative to represent items without depending on users feedback in the system and they have potential to represent the narratives in stories in a rich way, so they probably can be used to explain recommendations (RICCI; ROKACH; SHAPIRA, 2011).

Considering the increasing popularity of Japanese animations in Occidental World (MIYAKE, 2002), we proposed a new database composed of these animations motivated by the scarce use and analysis of this kind of data in the Recommender Systems field.

[2]Accessed: Jun/2018. Available in: `https://www.amazon.com.br/`.
[3]Accessed: Jun/2018. Available in: `https://www.facebook.com/`.
[4]Accessed: Jun/2018. Available in: `https://www.spotify.com/br/`.

1.2 Objectives

Our main goal is to investigate the wide range of techniques for recommendation applied to Japanese animations and using tropes. Thus, first of all, we will review the state of the art algorithms in Recommender systems to make possible select some algorithms to implement. Second, a new database will be proposed to be compared with the standard MovieLens. Third, we will make an analysis to understand how tropes influence in predictions. In the end, we will be able to compare the already known algorithms in literature and the one that uses tropes.

A way to define the main aims of this work:

- Review literature
- Propose a new database enriched with tropes
- Investigate in literature ways to use tropes in recommendations – with one or more variations
- Evaluate the trope influence in predictions

1.3 Structure

The next chapter provides the background information necessary to understand the work proposed and the database description. Chapter 3 presents an overview of the main approaches used in Recommender System involving more general purposes or the use of tropes. The databases are analyzed in Chapter 4. The results are discussed in Chapter 5. Finally, the conclusions and suggestions of future works.

2 Previous Concepts

Before proceeding with our related works study, it is necessary to explain about two concepts present in this work: Tropes and Animes. Our goal is an investigation into the state of the art in Recommender Systems but it is also an investigation about how Tropes (this concept will be explained below) can contribute to predictions. To this end, we have used the same movies database presented in (BATISTA et al., 2016). Furthermore, we also proposed a new database composed by Animes enriched with Tropes.

2.1 Tropes

Tropes are overused plot devices, the list of possibilities is big: can be a plot trick, a setup, a narrative structure, a character type, etc (BATISTA et al., 2016). They are always found in narratives such as animations, TV shows, movies, comics, books, games, so the fans started to recognize these patterns and classify them[1].

One of the platforms used to list and organize tropes is TVTropes[2]. All content is maintained by fans in collaborative work. There, we can start a search from a trope – we find a list of narratives where it is used. Alternatively, starting from a story, we can find a list with the correspondent plot devices used on the story.

As an illustration, given the classic movie Spirited Away (MACWILLIAMS, 2014). The plot is about a girl, Chihiro, lost in a world of spirits and other magical creatures, some of the tropes found on its page[3]:

- Coming-of-Age Story: the protagonist starts the history as a hard-headed little girl and as she lives new experiences during the narrative, in the end, she has gained fundamental learning to live;

[1]Accessed: Nov/2018. Available in: `https://tvtropes.org/pmwiki/pmwiki.php/Main/Trope`

[2]Accessed: Nov/2018. Available in: `https://tvtropes.org/`

[3]Accessed: Nov/2018. Available in: `https://tvtropes.org/pmwiki/pmwiki.php/Anime/SpiritedAway`

- Fish out of Water: Chihiro is detested in the spirit world because humans are known for disregard spirits;
- Parents in Distress: Chihiro's parents put themselves in a dangerous situation and she needs to protect herself and find a way to save them.

The tropes used in our databases were obtained from DBTropes (KIESEL; GRIMNES, 2010) – a Linked Data wrapper for TV Tropes –, it uses an ontology created in Skipforward to represent the data – which is easier to extract the tropes than to use web scrapping directly in TV Tropes web page. The version used is the last available in the official web page `skipforward.opendki.de/wiki/DBTropes`, dated Jun/2016 – with 61915 items, 27254 feature types, and 3580311 feature instances.

The items in the DBTropes are not all Japanese animations or movies, so it was necessary to filter the items and cross with the other databases.

2.2 Animes

Since we will work with a new database that will be presented later in this text, we need to understand what Japanese Animes are.

In Japan, Anime is a term used to refer to all kind of animations – short episodes aired in television or direct published in DVD format, and movies, for example –, but outside Japan, it is known as a term to refer to Japanese animations. When these productions started all content was focused in local public – Japanese themselves. But in recent decades, the public in the West Side has increased, turning into a cultural phenomenon and now it is part of pop culture in general (MIYAKE, 2002).

These animations are classified in a way similar to occidental movies and series – action, adventure, comedy, and so on – with the exception of some few genres such as *shojo* and *shonen* present specifically in anime context. The first refers to animations with female characters characterized by ultra feminine and passivity, while the second refers to animations defined as "boys' comics" (CLEMENTS; MCCARTHY, 2006).

Other specific genres present in animes universe are *mecha*, *ecchi*, and *hentai*. The first is stories about robots, usually involving fights and showing high technology (MIYAKE, 2002). The second and the third are adult content, usually *ecchis* have less explicit scenes than *hentais* (CLEMENTS; MCCARTHY, 2006).

Our main data source is MyAnimeList web site[4], it is a social network to anime fans. The user can save it as a list format what he/she have watched – with additional information like date and rating –, wants to watch or gave up watching. It is also an anime database with the cast, genres, studio, director, number of episodes, and other relevant data.

There is a MyAnimeList database available at `https://www.kaggle.com/CooperUnion/anime-recommendations-database` (see Table 1), but the number of items and users is too large (6337241 tuples with user, item and rating). To use this database, initially, we made a pre-filtering: selecting as valid genres only the top 18 most frequent genres in the database (see Table 2) and another important step was to select just the users with 20 or more animes rated. The dataset resulted has 6102494 entries. The criteria used were inspired by MovieLens Database (HARPER; KONSTAN, 2015) features such as the minimum number of ratings by user equal or more than 20 and the total number of different genres in the MovieLens 100K version.

Field name	**Description**
anime_id	MyAnimeList index
name	Anime title
genre	List of genres of the anime
type	Movie, TV, OVA, etc
episodes	Number of episodes
rating	Overall average
members	Members that added the anime to their list

Table 1: MyAnimeList Database field descriptions

In Table 2 it is observable that the base is unbalanced which means that there are some genres really popular when compared to others. The Table 3 describes the progressive reductions and database versions.

[4]Accessed: May/2018. Available in: `http://myanimelist.net/`

Genre	Number	Genre	Number
Comedy	4575	Slice of Life	1204
Action	2768	School	1176
Adventure	2316	Hentai	1133
Fantasy	2242	Supernatural	1001
Sci-Fi	2036	Mecha	929
Drama	1977	Music	842
Shounen	1684	Historical	798
Kids	1598	Magic	747
Romance	1437	Ecchi	628

Table 2: Number of animes by genre (the top 18 most frequent).

Database	Size
Original	7813737
Removed tuple with invalid rate	6337241
Removed invalid genres	6288425
Removed user with $<$ 20 rates	6102494

Table 3: Progressive reduction of database

The rating values in MyAnimeList are in [1,10] range, so they were normalized to [0.5,5] and removed the entries with no valid rate, once this database has also the entries when the user just sets the anime as watched and does not give a rating.

2.3 Chapter Summary

In this chapter, it was presented the concepts necessary to understand the data used in this work. Tropes are plot mechanisms used to get the public attention, this kind of categorization can be found in dedicated websites, such as TVTropes, where the users can list the tropes found in stories. Animes are Japanese animations that recently became more popular in the West Side. Similar to movies, they have a lot of genres – some of them are specific in animes, influenced by Japanese culture. Once there are lots of anime fans, MyAnimeList is a social network that can be used by users to list all animes watched, for example. We will present a database that was obtained in MyAnimeList and enriched with TVTropes data.

3 Theoretical Foundation

This chapter will present the main algorithms already known in Recommender Systems field. Here, we are focused on explicit feedback works using the users' rates – in other words, it is available how much a user liked an item – in some way to make predictions.

First, it is defined the main ways to measure how "good" an algorithm is solving the item prediction task, then we will be able to compare the different related works.

3.1 Validation Metrics

There are different kinds of measure to evaluate an algorithm, which will be used depends on the objective, for example, some works are focused in predict the rating the user eventually will give to an item, in this case, mean absolute error (MAE) (see Equation 3.1) and root-mean-square error (RMSE) (see Equation 3.2) are popular used metrics. The lower values mean better accurate predictions.

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \tag{3.1}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (p_i - q_i)^2}{N}} \tag{3.2}$$

Where:

p_i	is the rating value
q_i	is the predicted value
N	is the total number of predictions

3.2 Recommender Systems Algorithms

According to (AGGARWAL, 2016), the Recommender Systems are separated into three main pillars: Collaborative Filtering, Content Filtering, and Knowledge-based methods.

The first set of algorithms uses users rates in a cooperative way to make suggestions. The second set works using data features, each kind of item can have a different model to represent the most important features, alternatively, users information can be also used such as genre and occupation. The last one called Knowledge-based is the systems that work based on explicitly soliciting user requirements – this we will not enter in details, once it is not our focus. An extra pillar can be defined as the Hybrid Methods which combine the previously mentioned ones.

In Recommender Systems, it is common to mention the term *user-item matrix*, a way to represent data (see Table 4). The cell $r_{x,y}$ corresponds to the rate given by user u_x to the item i_y and, in most cases, each user rates just a small fraction of all items.

	i_1	i_2	...	i_n
u_1	3	4.5	...	5
u_2	2	2	...	4.2
...	...	...	...	...
u_m	5	4.8	...	4.7

Table 4: User-item matrix example

Our task is defined as predict the empty cells in the user-item matrix. In real world systems this predictions can be used in a rank where the top N greater values indicates the items the user is more likely to like, inclusive that is why there is a field specific focused only in how sort the item predictions (AGGARWAL, 2016).

When it was not mentioned, the database used is the MovieLens (HARPER; KONSTAN, 2015).

3.2.1 Collaborative Filtering

An algorithm is considered in Collaborative Filtering class if it uses ratings from multiples users in a collaborative way to predict missing ratings in user-item matrix (AGGARWAL, 2016). Some examples of algorithms with this principle are: User-based (HERLOCKER et al., 1999), Item-based (SARWAR et al., 2001), Slope One (LEMIRE; MACLACHLAN, 2005), and based on models – like Baseline (KOREN; BELL, 2015) and Matrix Factorization (KOREN; BELL; VOLINSKY, 2009).

3.2.1.1 Neighborhood Methods

User-based and Item-based are also called neighborhood methods because they work defining relations between elements, usually represented as vectors, by similarity functions – Jaccard and Cosine similarities are usual metrics. The User-based method estimates users similarity and make predictions based on closer neighbors likes (see Equation 3.3).

$$P_{u,i} = \frac{\sum_j (s_{i,j} * r_{j,i})}{\sum_j |s_{i,j}|} \tag{3.3}$$

Where:

$P_{u,i}$	is the predicted rate for user u on item i (User-based approach)
j	is in the set of the users most similar to u
$s_{i,j}$	is the similarity between users i and j
$r_{j,i}$	is the rate of item i given by user j

Analogously, the Item-based uses ratings to estimate items similarity and make predictions considering items more similar to user already rated items in a positive way – using weighted average (see Equation 3.4), for example.

$$P_{u,i} = \frac{\sum_j (s_{i,j} * r_{u,j})}{\sum_j |s_{i,j}|} \tag{3.4}$$

Where:

$P_{u,i}$	is the predicted rate for user u on item i (Item-based approach)
j	is in the set of the items most similar to i
$s_{i,j}$	is the similarity between items i and j
$r_{u,j}$	is the rate of item j given by user u

Observing Equations 3.4 and 3.3 – calculating prediction based on neighbors similarity – it is notable that they look almost equal, once they use the k most similar neighbors (users or items) principle to calculate prediction, approaches like these are also considered as KNN algorithms.

Their reported MAE values are nearly, User-based (HERLOCKER et al., 1999) presents almost all values in [0.73,0.735] range while Item-based (SARWAR et al., 2001) presents almost all values in [0.725,0.73] range – when varying the number of neighbors.

These techniques sound interesting, but they have some problems such as *sparsity* and *scalability* (SARWAR et al., 2001). Users usually rate a low number of items, it is called user-item *matrix sparsity* – both Item-based and User-based approaches suffer from it. Other complication is *scalability*, present in User-based method, because the number of users in Recommenders Systems is always increasing and the algorithm principle is calculate similarity between users, so every time a new user enjoy the system, all similarities should be recalculated (SARWAR et al., 2001).

Item-based Collaborative Filtering was revolutionary when proposed, because it works avoiding the *scalability* problem since the number of items in a system does not increase so fast as the number of users. But inevitably, new users/items are also a problem, and for this we have *new user* and, analogously, *new item* (RICCI; ROKACH; SHAPIRA, 2011) problems definitions. In both cases, we are dealing with lack of data, without information we do not know how user/item profile can fit in recommendation process.

Slope One was first proposed by (LEMIRE; MACLACHLAN, 2005), this method is an alternative to Item-based and User-based when considering computational efforts. It works using the "popularity differential" principle, better explained by a simple example as follows. Given users u_1 and u_2 and the following information in Table 5. To predict user u_2 rate to item i_b, we consider that the difference between rates given by u_1 – in this case, 5 - 4 = 1 – will be preserved by u_2, so the unknown "?" entry could be estimated by 3 + (5 - 4) = 4. This idea works well when considering the trade-off: easy implementation, efficient query time and accurate predictions. The MAE value obtained by this method is 0.752, which means reasonably closer to User-based and Item-based methods, but with the gains mentioned before.

	i_a	i_b
u_1	4	5
u_2	3	?

Table 5: Slope One example

Given the example, the difference in rates values can be formally defined as $diff_{i,j}$ (see Equation 3.5) and the prediction can be calculated $P_{u,i}$ (see Equation 3.6).

$$diff_{i,j} = \frac{\sum_{u \in S_{i,j}} (r_{u,i} - r_{u,j})}{|S_{i,j}|} \tag{3.5}$$

Where:

$S_{i,j}$	is the set of users rated both item i and j
$r_{u,i}$	is the rate given by u on item i
$r_{u,j}$	is the rate given by u on item j

$$P_{u,i} = \frac{\sum_{j \in C_i}(diff_{i,j} + r_{u,j})}{|C_i|} \tag{3.6}$$

$$P_{u,i} = \frac{\sum_{j \in R_u}(diff_{i,j} + r_{u,j}) * |S_{i,j}|}{\sum_{j \in R_u}|S_{i,j}|} \tag{3.7}$$

Where:

C_i	is the set of items rated by user u and other users that also rated item i
$diff_{i,j}$	is defined in Equation 3.5
$r_{u,j}$	the rate given by user u on item j
$S_{i,j}$	is the set of users rated both item i and j

There is also a modified version of Slop One, the Weighted Slop One (see Equation 3.7). The justification is to emphasize the item pairs with more rates.

3.2.1.2 Model-based Methods

Some approaches use the rating to find patterns in users behaviors but without using the most similar neighbors, the predictions can be generated without the dataset, only based on the generated model in the training process. These methods can be classified as model-based (HU; KOREN; VOLINSKY, 2008; KOREN; BELL; VOLINSKY, 2009; KOREN; BELL, 2015).

MovieLens and other datasets compound by ratings are a good criterion to compare different procedures, but some applications contexts make impossible have user explicit feedbacks. So Implicit Feedback emerges as an alternative. Compared to Explicit Feedback, some different challenges appears: no negative feedback, data noises, and alternative evaluation metrics as well pointed by (HU; KOREN; VOLINSKY, 2008).

As an example of model-based collaborative method applied to implicit feedback (HU; KOREN; VOLINSKY, 2008) works defining a vector to user-factors and other to item-factors by applying Singular Vector Decomposition algorithm. A low-cost function – considering the inner product between the factors vectors as prediction with some adjusts to avoid over fitting – is defined and a loop to optimize this value is executed, starting from recomputing all user-factors, then all item-factors and repeat until a predefined ideal number of iterations. The database used was a digital television service and their metric to estimate accuracy is totally unlikely MAE or other already presented metrics. They called it *rank* and defined to measure how much their estimation was close to the real-time the user spent watching the TV shows. The values presented were down to nearly 0.08, in contrast to approximately 0.11 or greater values obtained by other algorithms such as a simple recommendation by popularity.

Matrix Factorization (KOREN; BELL; VOLINSKY, 2009) such as (HU; KOREN; VOLINSKY, 2008) – can be viewed as an alternative to neighborhood methods, it also uses users ratings to model user behavior, but it works extracting features – not always explicit or easy ones to understand –, combining what the user gives importance and relevant item features in general. The great point of this approach is the support to add new features – even Implicit Feedback or temporal dynamics can be used to improve accuracy. The challenge is to map each user to a vector with user-factors and each item to a vector with item-factors – the same idea presented in (HU; KOREN; VOLINSKY, 2008) – with low dimension. For this purpose, the error between the real values and the estimated (Equation Equation 3.8) is minimized using the known rates in Stochastic Gradient Descent – see Equations 3.9 and 3.10 – or Alternating Least Squares methods. The results presented in (KOREN; BELL; VOLINSKY, 2009), in RMSE values, were better to matrix factorization with temporal dynamics, approximately 0.88. The dataset used was from Netflix.

$$e_{u,i} = r_{u,i} - (\lambda + b_i + b_u + q_i^T p_u) \tag{3.8}$$

$$q_i = q_i + \gamma(e_{u,i}p_u - \phi q_i) \tag{3.9}$$

$$p_u = p_u + \gamma(e_{u,i}q_i - \phi p_u) \tag{3.10}$$

Where:

$e_{i,u}$	is the error
$r_{u,i}$	the rating given by user u on item i
λ	is the overall average rating
b_i	the observed deviations of item i
b_u	the observed deviations of user u
q_i	vector representing the i factors
p_u	vector representing the u factors
γ	learning rate constant
ϕ	regularization constant

As a progressive development of Matrix Factorization, some works proposes the generalization of items and users in factors using Neural Networks. (XUE et al., 2017) uses two neural networks combined by dot product, as input they use the user-item matrix – but considering explicit and implicit information –, one neural network receives the users representations (the lines of the matrix) and the other neural network receives the items representations (the columns of the matrix) and the loss function proposed by them measures the error considering the different types of feedback. So, while Matrix Factorization has a matrix to represent users factors and a matrix to represent items factors, in this new approach the representation is the weights in output layer of the networks. The results reported are in different metrics, called Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG), used to measure ranking quality. The first measures the items predicted that actually are in the Top N, the second measures the quality if the high predictions are at top positions. The best results are [0.38,0.411] for HR and [0.68,0.69] for the other, when considering 64 factors and the MovieLens versions 100k and 1m.

Baseline or biases algorithm (KOREN; BELL, 2015) is another example of model-based interesting, by being simple and efficient (see Equation 3.11). There is no result reported in the reference used.

$$P_{u,i} = \lambda + b_u + b_i \tag{3.11}$$

$$b_i = \frac{1}{\lambda_1 + |R_i|} \sum_{u \in R_i} (r_{u,i} - \lambda) \tag{3.12}$$

$$b_u = \frac{1}{\lambda_2 + |R_u|} \sum_{i \in R_u} (r_{u,i} - \lambda - b_i) \tag{3.13}$$

Where:

λ	is the overall average rating
λ_1	is a positive constant
λ_2	is a positive constant
$r_{u,i}$	is the rate given by user u on item i

3.2.2 Content Filtering

Once there are available ratings there are no problems to make recommendations using Collaborative Filtering methods, but if this condition is not supplied alternatives should be searched. In this case, Content Filtering methods are useful.

A method is classified as Content Filtering when it uses items or users features to make predictions. For example, when considering movie domain, the features can be genres or directors (OZGEN, 2011; AGGARWAL, 2016). Alternatively, users features could be age or genre.

The lack of ratings is usually associated with the *new item problem*. In Collaborative Filtering, when a new item is inserted in the system, it is not possible to make a suggestion using this item because if there are neighbors but any of them rated the new item when it will not be recommended or there are no ratings to calculate similarity between new item and any other item (RICCI; ROKACH; SHAPIRA, 2011). The advantage of Content Filtering is not to be depended on ratings of other users to make predictions, once it uses features to calculate similarity.

Another benefit is transparency in recommendations because the process of matching up features with items is really clear and easy to show as information to user (RICCI; ROKACH; SHAPIRA, 2011).

Although the solution of *new item problem*, there are several disadvantages. The first one is the nature of Content-based recommendations tend to be very similar to the features identified as compatible to the user profile. Given this, just items with the same features will be recommended to the target used, so the suggestions tend to be very obvious and very similar to each other (RICCI; ROKACH; SHAPIRA, 2011).

The second disadvantage pointed is the persistence of *new user problem*. It is necessary the use of user entries to match the items pointed as interesting and the other items. Fortunately, there is the possibility of allowing the user directly point the features instead of inferring it from the history (AGGARWAL, 2016).

Analogous to neighborhood methods in Collaborative Filtering, the similar idea can be applied in Content Filtering, for example, instead of using the item similarity matrix based in rates, some vector item representation can be used to calculate the matrix similarity and then use Equation 3.4. An analogous adaptation can be done with users.

More recently, some works are appearing using tags (NGUYEN; RIEDL, 2013) – usually defined by users – to improve suggestions accuracy. Here, the proposed model uses Information Theory to define the mutual information of two random variables that in this context are user rating history and the relevance of tag t considering the history. Given this definition, they use Linear Regression (LR) and Support Vector Machine (SVM) as options to make predictions. Their MAE values were 0.54 for LR and 0.64 for Optimized

SVM, it is interesting compared to Collaborative Filtering results.

The use of the tag in recommendations is a promising idea, but some difficulties come along such as not all data has tags, how to extract what tags are relevant to the user (NGUYEN; RIEDL, 2013). Somehow it is possible to consider tropes as a specific type of tag, but instead of depending on users collaborative behavior to tag items, we use an alternative database dedicated to it – TVTropes.

3.3 Hybrid methods

With a view to achieving the benefits of different research lines, there are Hybrid methods – combining more than one approach to make better recommendations.

When considering hybrids methods using Collaborative and Content Filtering – our focus –, the different combinations can be defined in four ways, according to (BOBADILLA et al., 2013):

1: Collaborative and Content Filtering separated and then the predictions are combined

2: Content Filtering is inserted in Collaborative Filtering

3: Collaborative Filtering is inserted in Content Filtering

4: One model using both approaches

Latent factor models can be hybrid because they are able to incorporate different pieces of information from user rates but also user demographic data in the same model and so on. This specific method will depend on data used to be classified as category 4 or not even to fit this classification proposed because it is flexible to accept more than Filtering and Content approaches.

Some research lines use user features (GUPTA; PATIL, 2015), while others use item features (OZGEN, 2011; BATISTA et al., 2016). (GUPTA; PATIL, 2015) uses KNN to cluster users based on genre, occupation, and age then uses Hierarchical Clustering algorithm Chameleon – this method is classified in category 2. To make predictions, the frequency of rates in the user cluster is used to calculate the most frequent value. The MAE values for this method were in the range [0.60-0.70].

(OZGEN, 2011) defines five ontologies and similarity metrics to compare any two items. Then apply this Content Filtering method in the user-item matrix to predict unknown rating – a way to solve the *matrix sparsity*. After, User-based Collaborative Filtering is applied to make predictions for a user – the items with the highest predicted rates are selected. It is also classified as category 2.

3.4 Tropes

Finally, we found few works using tropes, (BATISTA et al., 2016) is one of them. It uses a linear combination of Weighted Slope One (see Equation 3.14), a variation of Slope One considering as weight the number of times the pair of items was rated by different users, and a similarity – Cosine and Jaccard – distance calculated using vector representation TF-IDF with variations, each value in vector corresponds to trope or trope with genre. The dataset used was MovieLens ∩ TV Tropes (to select the tropes for each movie). The best results were trope with genre using TF-IDF representation and cosine similarity, RSME value was 0.9015.

$$P_{u,i} = \frac{\sum_{j \in R_u} (diff_{i,j} + r_{u,j}) * \frac{|S_{i,j}|}{SD_{i,j}}}{\sum_{j \in R_u} \frac{|S_{i,j}|}{DS_{i,j}}} \tag{3.14}$$

Where:

$diff_{i,j}$	is defined in Equation 3.5
$r_{u,j}$	the rate given u on item j
$S_{i,j}$	is the set of users rated both item i and j
$SD_{i,j}$	distance similarity between items i and j

(BATISTA et al., 2016) proposed a modification in (YANG; HU; QU, 2013) method that uses semantic distances applied in a graph representation with different kinds of data – inclusive tropes – as metric in $SD_{i,j}$. Instead of direct and indirect distances (PASSANT, 2010), (BATISTA et al., 2016) represents the items as a vector where each position represent a different trope and the value in a position represents the number of times the trope occurs.

Given this vector representation some popular metrics of similarity can be used as Cosine and Jaccard. Once $SD_{i,j}$ is a *distance* metric, the Cosine and Jaccard similarity can be adapted to distance as follow (see Equations 3.16 and 3.18):

$$Sim_{Jacc} = \frac{|x \cap y|}{|x \cup y|} \tag{3.15}$$

$$D_{Jacc} = 1 - Sim_{Jacc} \tag{3.16}$$

$$Sim_{Cos} = \frac{x \cdot y}{|x| * |y|} \tag{3.17}$$

$$D_{Cos} = 1 - Sim_{Cos} \tag{3.18}$$

Where:

x, y	vector representation of items x and y

Finally, to verify the effectiveness of the change in $SD_{i,j}$, a linear combination was used as follows:

$$P_{u,i} = (1-\alpha) * \frac{\sum_{j \in R_u} (diff_{i,j} + r_{u,j}) * |S_{i,j}|}{\sum_{j \in R_u} |S_{i,j}|} + \alpha * \frac{\sum_{j \in R_u} (diff_{i,j} + r_{u,j}) * \frac{|S_{i,j}|}{SD_{i,j}}}{\sum_{j \in R_u} \frac{|S_{i,j}|}{SD_{i,j}}} \tag{3.19}$$

Where:

$diff_{i,j}$	is defined in Equation 3.5
$r_{u,j}$	the rate given u on item j
$S_{i,j}$	is the set of users rated both item i and j
$SD_{i,j}$	distance similarity between items i and j
α	is a parameter to adjust the influence of modified version with $SD_{i,j}$

Our work uses the method proposed by (BATISTA et al., 2016), but focused on just how the additional information offered by tropes can influence in recommendation, this is why in the original analysis is made a lot of variations in the use of tropes with different filters considering combination with genre and so on, while in this work we are restricted in use just one simple way to represent the items with tropes.

3.5 Chapter Summary

Recommender Systems algorithms can be defined with the goal of predicting the rating by a user on an item, the way the algorithm works to make this prediction can be categorized in Collaborative Filtering, Content Filtering, Knowledge-based – this one is not our focus –, and Hybrid methods that combine both previously mentioned. The Collaborative is known by the principle of use users ratings to find patterns and make predictions – examples are Item-based and User-based Collaborative Filtering, Matrix Factorization, and Baseline. The Content-based principle works using items or users information to make recommendations, this alternative can be useful when there is not enough information, for example, a new item never rated before or a new user with a clean rate history. The hybrid methods are supposed to be the best of both. There are few works using Tropes, one of them is Slop One Weighted Hybrid (BATISTA et al., 2016), the Slop One is a Collaborative Filtering algorithm and the tropes are used to try improving the predictions precision.

4 Data Analysis

For our experiments on how to analysis techniques for system recommendation applied to Japanese Animes, we have selected a list of algorithms to implement and to evaluation we use two databases: MovieLens Database and MyAnimeList, both enriched with TVTropes data. Next, we have done analysis to know better the data. We hope this chapter can be helpful to understand better the results of the experiments.

4.1 MovieLens Database

The MovieLens Database (HARPER; KONSTAN, 2015) is a standard reference to evaluate recommender systems algorithms – there are some different versions of the database, we used one derivative from 100k version –, thus we have decided to firstly use it. Its data contains movie ratings – varying 1 up to 5 stars – collected at MovieLens Recommender System (available at `https://movielens.org/`). The database has 1682 movies categorized in 18 different genres (see Table 6).

In Table 7, it is showed the number of items (column N) and the number of genres presented in the items. Despite the number of different genres, each item on average has at most 3 genres.

Genre	Number	Genre	Number
Comedy	292	Crime	62
Drama	288	War	54
Action	189	Mystery	44
Thriller	162	Musical	41
Romance	135	Animation	22
Adventure	105	Western	17
Sci-Fi	84	Fantasy	17
Children	77	Film-Noir	16
Horror	67	Documentary	4

Table 6: Number of movies by genre

Number of genres	**N**	**Percent (%)**
1	307	36.33
2	316	37.39
3	165	19.52
4	45	5.32
5	10	1.18
6	2	0.23
Total	845	100

Table 7: Number of genres by item (MovieLens database)

Characteristic	**MovieLens**	**MovieLens ∩ TVTropes**
Number of entries	100000	80339
Number of items	1682	845
Number of users	943	943

Table 8: MovieLens database and MovieLens ∩ TVTropes summary

The experiments in (BATISTA et al., 2016) used an intersection between MovieLens 100k and DBTropes – this base will be referred as MovieLens ∩ TVTropes. The main information of the database is in Table 8. The number of items lost in the intersection database is notable, in the following sections some analysis about the final version database (MovieLens ∩ TVTropes column) are made.

There are 10366 different tropes distributed by items as described in Table 9 – it is interesting that more than half of all movies have less than 50 tropes described. Additionally, when considering the number of occurrences of each trope in the database, the vast majority have few occurrences (see Table 10). It occurs because tropes tend to be very particularized and depend on voluntary work for categorization, so only a small set of items – probably the famous ones – has a more detailed number of tropes listed and it would be necessary an organized work to give equal attention to all items.

Analyzing statistics related to rates, there are some points to be highlighted (see Figure 1 and Table 11). The predominant rating value is 4, maybe it is a first signal that the predictions values tend to be close to this value. So if there is no information about your data, a good way to make a prediction is just try the overall average value. The number of ratings received can reflect in the prediction quality, at least, when considering items without rates, it is a problem – already explained at Chapter 3.

Finally, observing the users behavior (see Table 12). The majority of the users is located in [0,100) interval, in other words, they rated a low number of items. The original MovieLens database has at least 20 rates by each user but the intersection analyzed

Number of different tropes by item	**N**	**Percent (%)**
[0,50)	458	54.2011
[50,100)	188	22.2485
[100,150)	71	8.4023
[150,200)	48	5.6804
[200,250)	35	4.1420
[250,300)	13	1.5384
[300,350)	9	1.0650
[350,400)	11	1.3017
[400,450)	4	0.4733
[450,∞)	8	0.9467
Total	845	100

Table 9: Statistics about number of different tropes by item (MovieLens ∩ TVTropes)

Number of times the trope occurs in database	**N**	**Percent (%)**
[0,20)	9892	95.4273
[20,40)	347	3.3474
[40,60)	74	0.7138
[60,80)	25	0.2411
[80,100)	17	0.1639
[100,∞)	11	0.1061
Total	10366	100

Table 10: Statistics about number of times the trope occurs in database (MovieLens ∩ TVTropes)

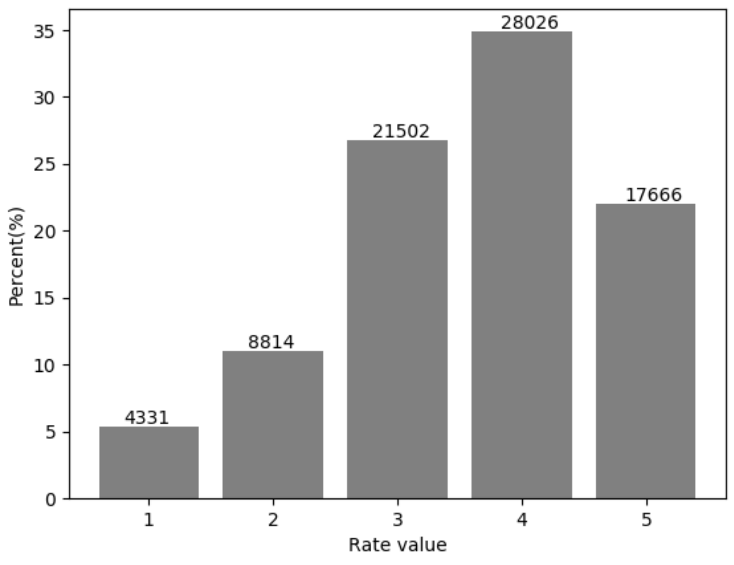

Figure 1: Rate values distribution (MovieLens ∩ TVTropes).

Number of rates by item	**N**	**Percent (%)**
[0,100)	563	65.69
[100,200)	185	21.58
[200,300)	77	8.98
[300,400)	20	2.33
[400,500)	8	0.93
[500,600)	4	0.46
Total	845	100

Table 11: Number of rates by items (MovieLens ∩ TVTropes)

Number of items rated by users	**N**	**Percent (%)**
[0,100)	647	68.61
[100,200)	201	21.31
[200,300)	73	7.74
[300,400)	17	1.80
[400,500)	5	0.53
Total	943	100

Table 12: Number of rates users usually have in their history (MovieLens ∩ TVTropes)

does not obey this restriction because of the movies removed when they were not find in TVTropes, the minimum value of ratings by user is 9. In these cases, the prediction precision eventually is prejudiced.

4.2 MyAnimeList Database

The MyAnimeList Database – in short: MAL – is extracted from `myanimelist.net`, the version used in this work is a subset of the database available at Kaggle[1] – according to what was described previously in Section 2.2. Here, we describe the criteria used to reduce the database and analyze the statistics just as done with MovieLens ∩ TVTropes.

There is no database to MyAnimeList with tropes, so it was necessary to extract the tropes using DBTropes and merge the items present in both databases, the resulted database is called MyAnimeList ∩ TVTropes.

The tropes in the DBTropes are represented in a ontology, following a markup language, patterns were found to extract just the list with trope names. Once these lists were generated for each item, there were represented as lists in a new column in the database. When running the algorithms the tropes were transformed in vectors where each position represents a specific trope that can occurs or not (1 or 0).

[1]Accessed: Nov/2018. Available in: `https://www.kaggle.com/CooperUnion/anime-recommendations-database`

Number of genres	N	Percent(%)
1	174	9.92
2	349	19.89
3	490	27.93
4	380	21.66
5	237	13.51
6	84	4.78
7	24	1.36
8	13	0.74
9	3	0.17
Total	1754	100

Table 13: Number of genres by item (MyAnimeList database)

Characteristic	Original MAL	MAL	MAL ∩ TVTropes
Number of entries	7813737	101242	57061
Number of items	11200	5074	1754
Number of users	73515	1000	1000

Table 14: MyAnimeList and MyAnimeList ∩ TVTropes summary

The summary of the resultant database can be observed in Table 14, the first column referred as "Original MAL" is the original version available at Kaggle, we made a first filtering (more details in Section 2.2), a second filtering (it will be described below), and then, it was possible to select a part of the database – the one named "MAL".

Some criteria were used to select part of the database and generate a base similar to MovieLens 100K. It was removed from the database all users that rated more than the upper limit (whisker) in the box plot (see Figure 2), just to avoid users that do not represent the majority usual users behavior. Once this filtering was applied, the "MAL" database present in Table 14 version was obtained selecting 1000 users randomly.

The MyAnimeList ∩ TVTropes is the intersection between the MyAnimeList subset generated as described above and the animes found in TVTropes. Observing in Table 14, it is notable the difference between the number of animes in the final version and the intermediary version, only 34.57% was preserved. One of the reasons for this problem was the difficulty to merge the common data in the different bases once MyAnimeList usually uses the original Japanese title of the anime while TVTropes uses the English title.

Observing Table 15 and Table 9 (the equivalent version to MovieLens), there is no big difference in distribution despite the bigger number of items in MAL database. The total number of tropes is 19467 and the way they are distributed can be viewed in Table 16. In general, the occurrences of tropes are more frequent when compared with Table 10 where

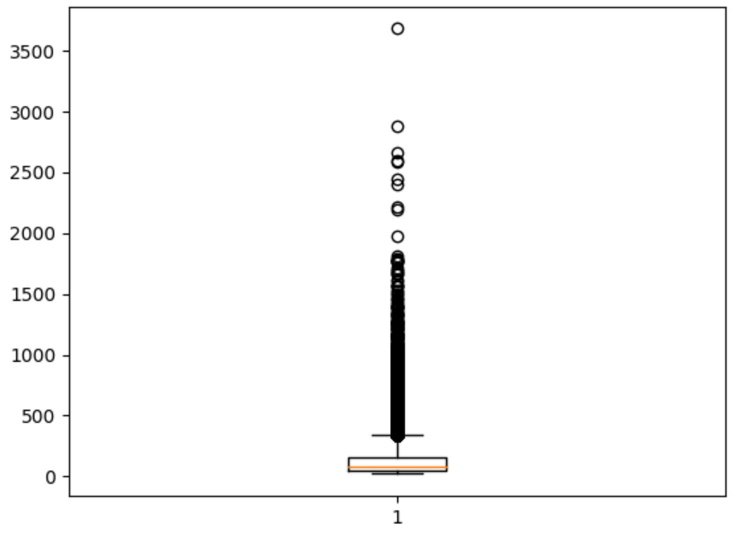

Figure 2: Box plot – axis y is the number of rates given by the users.

Number of different tropes by item	**N**	**Percent (%)**
[0,50)	831	47.3774
[100,150)	211	12.0296
[150,200)	120	6.8415
[200,250)	33	1.8814
[250,300)	57	3.2497
[300,350)	23	1.3112
[350,400)	10	0.5701
[400,450)	26	1.4823
[450,500)	21	1.1972
[500,∞)	316	18.0159
Total	1754	100

Table 15: Statistics about number of different tropes by item (MAL $\cap$ TVTropes)

Number of times the trope occurs in database	**N**	**Percent (%)**
[0,20)	9543	49.0214
[20,40)	3620	18.5955
[40,60)	2359	12.1179
[60,80)	1455	7.4741
[80,100)	901	4.6283
[100,120)	584	2.9999
[120,140)	369	1.8955
[140,160)	250	1.2842
[160,180)	157	0.8064
[180,200)	108	0.5547
[200,∞)	121	0.6216
Total	19467	100

Table 16: Statistics about number of times the trope occurs in database (MAL ∩ TVTropes)

there are fewer rows in the table because the number of trope occurrences in the database does not exceed much more than 100. Maybe it can be explained by the more restricted nature of animes database, that is a more specific kind of data, so the animes tend to be more similar each other – as consequence they can have more tropes in common – than movies each other where the scope is huger.

The values in Figure 3 was discretized in the intervals [0,1], (1,2], (2,3], (3,4], and (4,5] – called 1, 2, 3, 4, and 5, respectively. It is notable the predominance of values in (4,5] range, what means that the users majority likes very much what they watch. Maybe it can be explained by the predominant users demography, in general, is young, male, and very enthusiasts about animes (REYSEN et al., 2016). Other hypothesis to justify this distribution is the MyAnimeList web site allows the user just set the anime as watched without no rate, so some users when do not like the item just do not give a rating. The MovieLens and MyAnimeList rate distribution can be better compared in Figure 4.

Table 17 and Table 18 show some statistics about rating distribution. It is curious to observe in Figure 17 that there is a notable number of items with few rates – maybe it can be explained by the fact that animes are more longer than movies, so naturally it takes more time to a user watch a higher number of items. Table 18 presents a distribution similar to Table 12. If in general, the number of items rated by a user is lower than 100, it means that each user does not usually know more than 6% of all items available in the system – illustrating the sparse matrix problem.

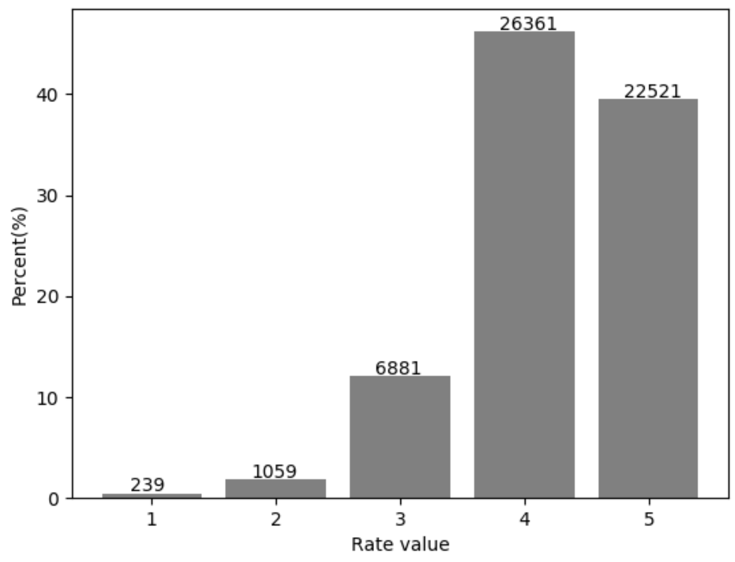

Figure 3: Rate values distribution (MAL ∩ TVTropes).

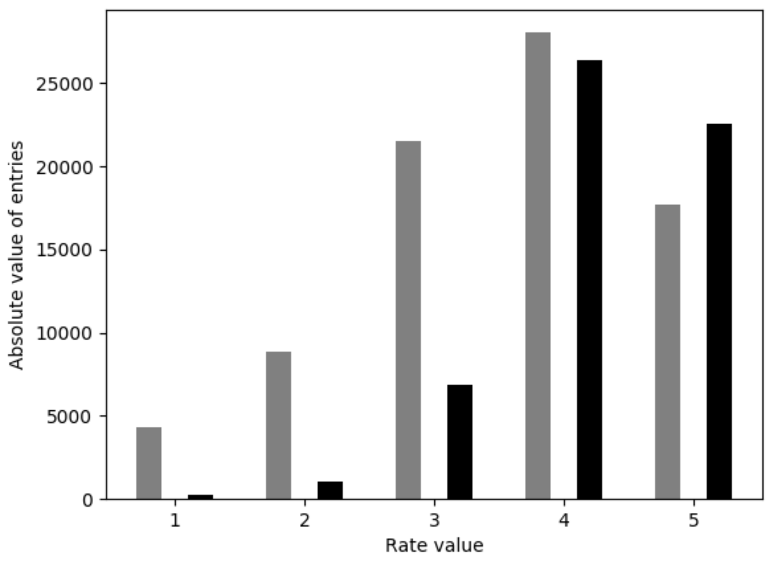

Figure 4: Rate values distribution comparison, black is MAL ∩ TVTropes and gray is ML ∩ TVTropes.

Number of rates by item	**N**	**Percent (%)**
[0,100)	1607	91.61
[100,200)	102	5.8
[200,300)	28	1.59
[300,400)	9	0.51
[400,500)	7	0.39
[500,600)	0	0
[600,700)	1	0.05
Total	1754	100

Table 17: Number of rates by items (MAL ∩ TVTropes)

Number of items rated by users	**N**	**Percent (%)**
[0,50)	546	54.6
[50,100)	289	28.9
[100,150)	133	13.3
[150,200)	31	3.1
[200,250)	1	0.1
Total	1000	100

Table 18: Number of rates users usually have in their history (MyAnimeList ∩ TVTropes)

4.3 Chapter Summary

MovieLens and MyAnimeList database were briefly analyzed in this chapter. The first consists of rates on movies – 5-star system – and the second rates on animes – 0 up to 10 rate values normalized to [0.5,5] range. Some interesting and important patterns found are about the number of tropes by each item that usually is low and the number of times a trope occurs in the database is usually low too. So there are lots of tropes and only a small part of them are considerably rated by a big number of users. One difference between the databases is the rating values distribution, MovieLens it is more equally distributed when compared with MyAnimeList where more than 50% of all ratings is in [4,5] range.

5 Experiments and results

In order to validate our experiments on analysing several algorithms for Japanese Anime recommender system, we have decided to use 5-cross-validation and the results are reported using RMSE metric, as specified in Equation 3.2. When Jaccard and Cosine similarities are mentioned it is a reference to Equations 3.15 and Equation 3.17.

The cross-validation was chosen because when an algorithm is trained with only one data set there are some possible problems that can be not identified such as overfitting, but when this method is used, we can measure better how the algorithm is accurate varying different train and test sets.

The tests were validated using MovieLens ∩ TVTropes – the 5-cross-validation database used is the same used in (BATISTA et al., 2016), it follows the MovieLens pattern – and MyAnimeList ∩ TVTropes databases. MyAnimeList ∩ TVTropes database was sliced in 5 parts (each user history was divided into 5 parts randomly selected) to generate 5 different combinations of train and test of the database. In each version, one part is separated as the test and the rest is used as the train.

The database versions used are the same used to data analysis in the last chapter. For convenience, instead of MyAnimeList and MovieLens, we also use the short version MAL and ML, respectively.

When there is not enough information to apply the recommender method, the value used as default was the overall average of the ratings in the database.

The projected was implemented using Python 3, NumPy[1], and SciPy[2]. All algorithms were based on the references already mentioned before. The databases was also mounted and processed using Python and the other packages mentioned.

[1]Accessed: Dez/2018. Available in: `http://www.numpy.org/`.
[2]Accessed: Dez/2018. Available in: `https://www.scipy.org/`.

5.1 Algorithm Results

The following algorithms: User-based and Item-based Collaborative Filtering, Content-based on Genres, and – alternatively – Tropes, Matrix Factorization, Baseline and Slop One Weighted Hybrid were selected based on their relevance in literature considering the scope of Recommender Systems using explicit feedback and focused on rate prediction.

5.1.1 Baseline

Baseline (KOREN; BELL, 2015) is a simple algorithm but with good results. The parameters λ_1 and λ_2 used were 0 and 0 – see Equations 3.13 and 3.12. It has a notable good performance when compared to other more complex algorithms (see Table 19).

	ML	**MAL**
RMSE	0.9379	0.6249

Table 19: Baseline results for MAL and ML databases

5.1.2 User-based Collaborative Filtering

The User-based Collaborative Filtering has two main parameters that we vary: the function similarity and the k number of neighbors considered in prediction. See results in Figure 5.

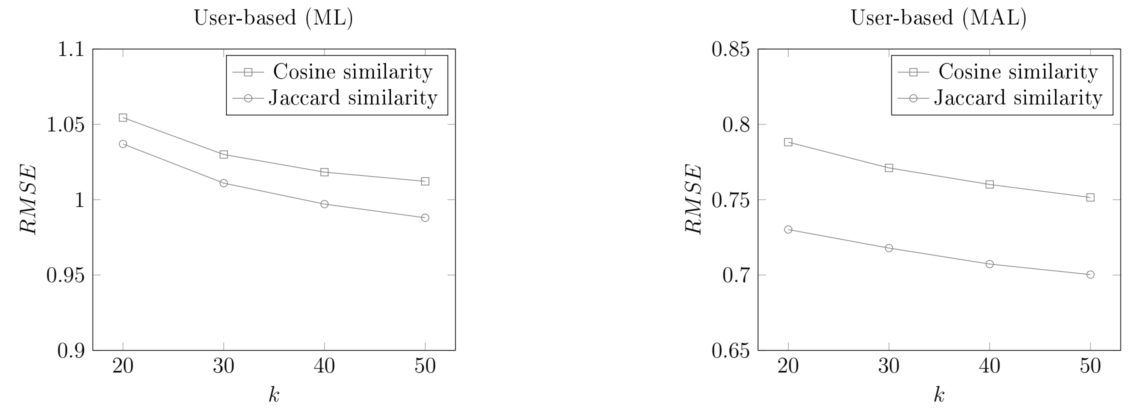

Figure 5: User based results

5.1.3 Item-based Collaborative Filtering

The Item-based Collaborative Filtering (SARWAR et al., 2001) has two main parameters that we vary: the function similarity and the k number of neighbors considered in prediction. See the results in Figure 6. In general, the k variation reflects, as expected, the curve behavior – the more neighbors you have the more information you have to more precision predictions.

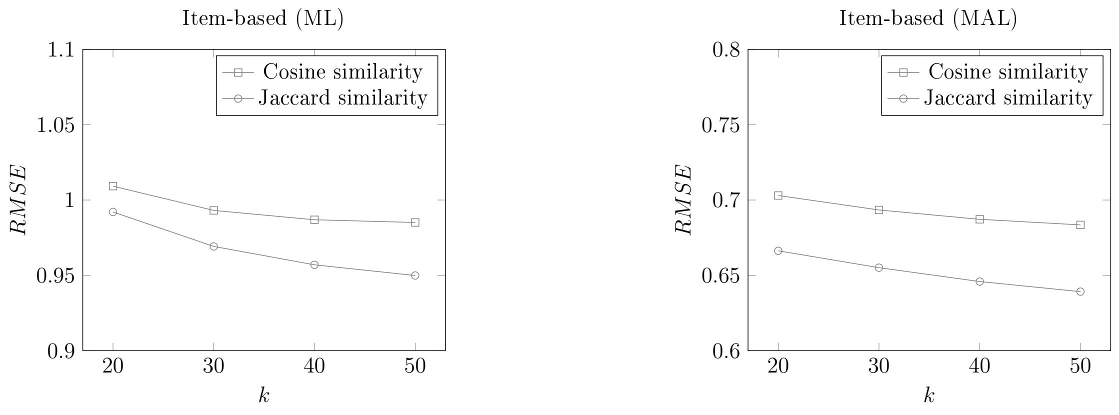

Figure 6: Item based results

5.1.4 Matrix Factorization

Matrix Factorization (KOREN; BELL; VOLINSKY, 2009) has a lot of different parameters, the implemented version uses just explicit feedback – like the other algorithms – and some of the parameters were fixed such as learning rate and regularization, the version implemented uses Gradient Descent to minimize the error and biases. The first fixed as 0.05 and the second as 0.02. The parameters changed were the number of factors (to represent items and users) and the number of iterations. See the results plotted in Figure 7.

It is interesting to observe that the number of k factors increase starts impacting in a negative way and then go back to improve the results in the ML case. While in MAL case the variation just keeps the results or help to decrease the error.

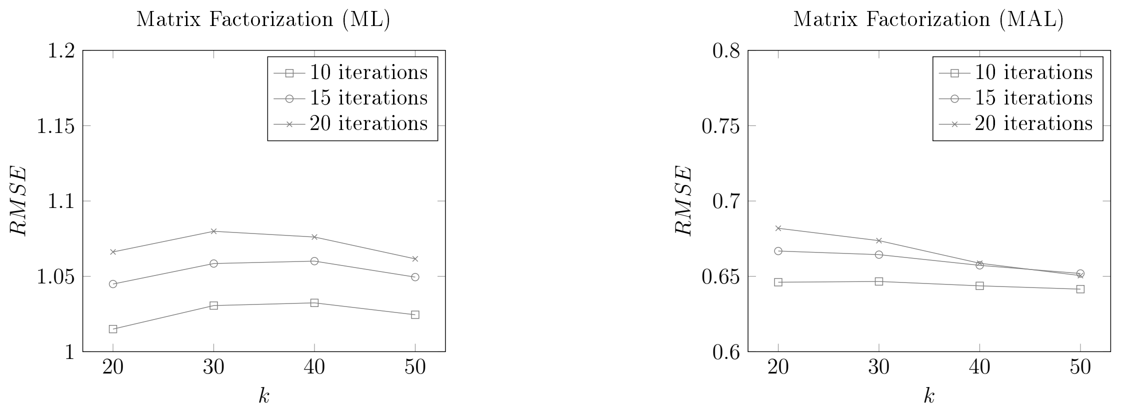

Figure 7: Matrix Factorization results

5.1.5 Content-based: genres and tropes

The content-based algorithms (AGGARWAL, 2016) were implemented using genres and tropes – there is no reference pointing to the use of tropes as data representation for this algorithm. The varied parameters were the function similarity and the number of neighbors used to calculate prediction. See Figure 8 and Figure 9. It is curious that to vary the function similarity did not impact in predictions, for example, in Item-based or User-based, there is a notable difference when changing the function, but considering the content description used to represent the items, there is no difference.

The vector representation used has N positions representing the N different tropes and filled with ones and zeros, i.e. the trope frequency in the item it was not used, it is only used if it occurs or not (1 or 0).

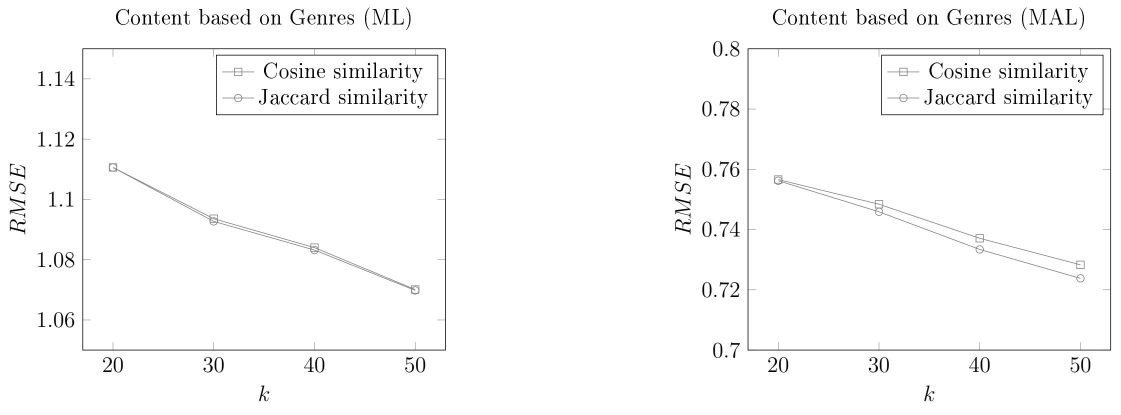

Figure 8: Content based on genres results

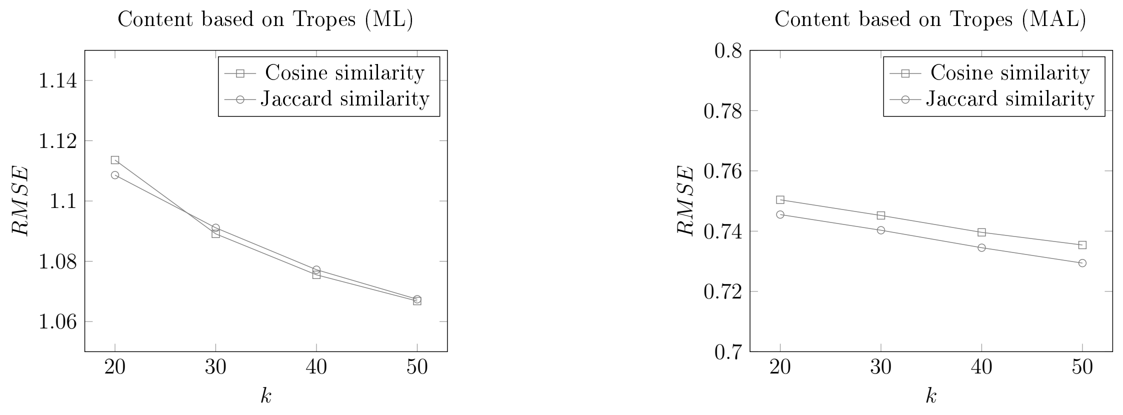

Figure 9: Content based on tropes results

5.1.6 Slop One Weighted Hybrid

The Slop One Weighted Hybrid algorithm (BATISTA et al., 2016) has two main parameters to vary. The first one is α that defines the influence of the hybrid part in prediction calculus (see Equation 3.7). The other parameter is the distance function used to compare two items. See Figure 10.

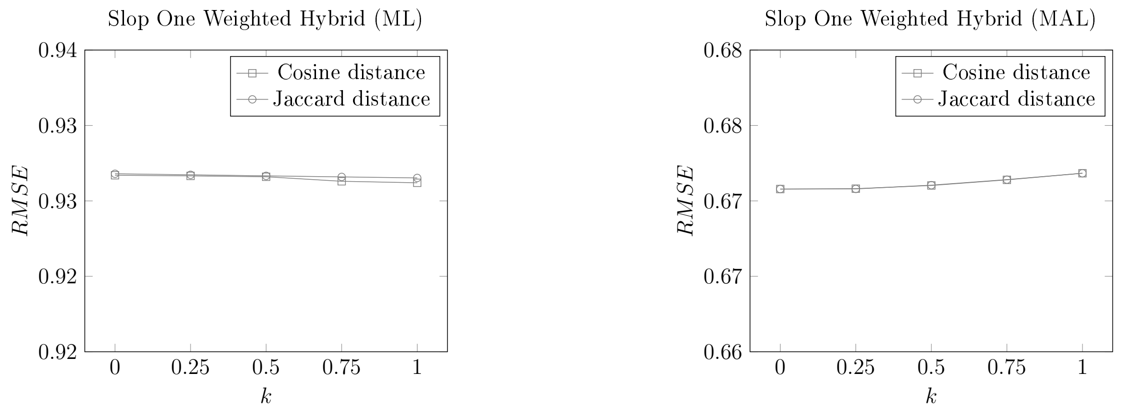

Figure 10: Slop One Weighted Hybrid results

The RMSE values obtained for the ML database are a little higher than the related in (BATISTA et al., 2016), there are two possible explanations. The first is we did not use any kind of the previous process in the tropes, so it is possible some subtle difference between the implementation in the calculus of the distance. (BATISTA et al., 2016) did not explain the way to solve the problem when there is no information enough available.

5.2 General view and discussion

An overview is summarized in Table 20. The "Default" algorithm is a simple approach that returns always the overall average considering the ratings available. Our analysis is based on three parts: a comparison between values in each database and a comparison between algorithms. The last part is a brief comment about the impact of the use of tropes in the methods.

Algorithm (with configuration)	**RMSE**
Default	1.1071
Baseline	0.9379
Item based (k = 50; Jaccard similarity)	0.9499
User based (k = 50; Jaccard similarity)	0.9880
Matrix Factorization (factors = 20; iters = 10)	1.0150
Content based on Genres (k = 50; Jaccard similarity)	1.0698
Content based on Tropes (k = 50; Cosine Similarity)	1.0668
Slop One Weighted Hybrid ($\alpha = 1$; Cosine Distance)	0.9312

Table 20: The best results for each algorithm (ML database)

Algorithm (with configuration)	**RMSE**
Default	0.7589
Baseline	0.6249
Item based (k = 50; Jaccard similarity)	0.6392
User based (k = 50; Jaccard similarity)	0.7003
Matrix Factorization (factors = 50; iters = 10)	0.6415
Content based on Genres (k = 50; Jaccard similarity)	0.7238
Content based on Tropes (k = 50; Jaccard Similarity)	0.7294
Slop One Weighted Hybrid ($\alpha = 0.5$; Cosine Distance)	0.6710

Table 21: The best results for each algorithm (MAL database)

The results selected to summarize the tests were based on the best configuration to obtain the best results for each algorithm. Almost all configurations in ML were also selected to MAL, the cases where it did not happen are explained by the proximity between the values when varying the parameters. For example, the Content-based on Tropes, see Figure 9, the values in axis Y vary from 1.05 to 1.15 – in ML results, nevertheless, the two lines still very close and there is a cross.

5.2.1 Database Comparison Results

The RMSE values observed to ML database were all higher than the values observed to MAL database. There are some hypothesis to justify this difference. The first one is visible when the rate distribution is observed – see Figure 1 and Figure 3. The way the users rate the items in MovieLens is different when compared to MyAnimeList. In the first, the values are more distributed, the value 4 is the most usual but the other values represent more than half of all entries. While in MyAnimeList, the value 4 is the most usual but the value 5 is in second place, and when compared to the other values frequency, basically it is just these two values predominating.

As result of the rate distribution, it is probable that all value predictions tended to values around 4 and 5 in the MAL case, so the errors measured by RMSE consequently were smaller than the errors in ML case.

Another explanation for the difference is the rating values range. Originally, MyAnimeList allows the users to rate values in [0,10] range. Once MovieLens has a 5-star system, the values in MyAnimeList were normalized to [0.5,5] range. Until this point, there is no visible problem, but observing the rate values in MAL base, there are a lot of float numbers, so the error measure eventually could be influenced by those less speared values.

5.2.2 Algorithms Comparison Results

Observing Table 8 and Table 14 the first unexpected result is the top best algorithm performances were obtained by Slop One Weighted Hybrid, Baseline, followed by Item-based and finally by Matrix Factorization in MovieLens valuation. Baseline, Item-based, Matrix Factorization, and Slop One modified version were on top 4 in MyAnimeList valuation.

Considering the top 4 best algorithms, some considerations can be pointed. First, trying to understand why Matrix Factorization did not have better results. The parameters configuration is a good point to be analyzed, it could be better if some tests varying learning rate and regularization were done or even just test less or more factors, considering the curve behavior, it could be verified if the values would keep decreasing or not.

Thinking about the Matrix Factorization idea, the database size certainly impacts the learning process. Both ML and MAL are small databases when compared, for example, to other larger versions available of the MovieLens. As a result, other algorithms can have a better performance with fewer data.

One more aspect to analyze is the nature of the data, in literature, Matrix Factorization is one of the best approaches known, and it is also known by the flexibility to insert more data from different types – implicit, temporal, content-based, etc – so, the version implemented could be enriched to have better results.

The value obtained by the Default algorithm can be used as a warning, if some algorithm had been worst results than this, it would be interesting to review the reason for so bad performance. It was not the case, the worst values obtained were from the Content-based algorithms. In literature, the content based approaches are known as an alternative to solve the lack information problem of the new item or new user, but they rarely are used alone to prediction, so these results are not a bad indicative.

Finally, one last point to emphasize is the difference, why Slope One modified version had better RMSE results in MovieLens, but not in MyAnimeList. Considering the Slop One first step is to calculate the Equation 3.5, based on the difference values when a user rates a pair i and j of items. So, Table 17 shows the number of rated by items, in MyAnimeList the predominant range is [0,100) – more than 90% – , in other words, each item is less rated in general when compared to MovieLens (see Table 11). In contrast, Baseline was more accurate in MyAnimeList database because it was benefited by the unbalanced rating distribution once it uses the overall average ratings in the prediction calculus.

5.2.3 Tropes

Finally, two analysis can be done, the first comparing the Content-based approaches. And the second observing the Slop One Weighted Hybrid results.

When comparing the genres and the tropes representation there is no relevant difference upon the results obtained. It was unexpected when considering that the number of tropes by item is considerable higher and potentially could be richer to describe the items than only genres. For example, see Table 7 and Table 13, the number of genres by item is lower than the number of tropes by item. One possible explanation is that the tropes occurrences by themselves are not enough to influence in predictions, once there are a lot of different tropes and maybe each one not occurs a number of times enough in the database to be a good differentiating criterion. Alternatively, tropes is a kind of information that changes the impact in story influenced by other factors such as genres, one of the discussions in (BATISTA et al., 2016) points to this conclusion.

When analyzing the tests with the modified version of Slop One (see Figure 10), there is no notable difference between the the use of tropes or not. The progressive increase of the α impacts in a positive way in the MovieLens $\cap$ TVTropes database, but the RMSE value differences are really small. While in MyAnimeList $\cap$ TVTropes database, the error value increased, but like in the other database, it was also a small difference. So, such as the last comparison, we are not able to affirm that tropes impacts in a notable positive way. The conclusion about the tropes use in MovieLens $\cap$ TVTropes database is in line with the related results in (BATISTA et al., 2016).

5.3 Chapter Summary

It was implemented 7 different algorithms – Baseline, Item-based, User-based, Matrix Factorization, Content-based genres and alternatively in tropes, finally, Slop One Weighted Hybrid –, and one more called “Default” that it is just a naive approach used as a reference of the interval where the RMSE value should be. Comparing the results obtained in MovieLens and MyAnimeList databases, there is a notable difference in the RMSE values, the hypothesis to justify this difference is the rating values distribution and the denser – not only integer values – present in MyAnimeList ratings.

When comparing the algorithms, it was not expected, but the best result was obtained with the Slope One modified version implementation, followed by, Baseline, Item-based, and Matrix Factorization – for the MovieLens database. While in MyAnimeList the best results were obtained by Baseline, Item-based, Matrix Factorization, and Slop One modified version.

6 Conclusions

The presented work is an overview of Recommender Systems field and an investigation of the tropes impact in predictions evaluated in a new Japanese database and in the standard MovieLens database, both with tropes. To this end, it was implemented and compared the results of some algorithms in the state of the art. In addition, some investigation about tropes was made based on results obtained comparing methods using trope information or not.

Based on the literature review it was selected some of the main methods used to recommendation such as Item-based or Matrix Factorization and a hybrid method that uses tropes called Slop One Weighted Hybrid (BATISTA et al., 2016).

The tests were performed in two different databases, the first was MovieLens enriched with TVTropes information – the same used in (BATISTA et al., 2016) – and the second was MyAnimeList – selected from a previous database available – and enriched with TVTropes.

Before relate and discuss the results, both databases were analyzed. Some interesting patterns were observed, for example, there is a difference in the way the rates values are distributed in each database. Despite the predominance of one single value, the MyAnimeList is more dense in higher rating values while MovieLens has a little more distributed values.

The results pointed to best performance were Slop One Weighted Hybrid, Baseline, Item-based Collaborative Filtering, and Matrix Factorization – on this order when considering MovieLens database. For MyAnimeList the best results were obtained by Baseline, Item-based, Matrix Factorization, and Slop One modified version, respectively. There is a notable difference between the results of RMSE obtained in MovieLens and MyAnimeList databases for the same algorithms and parameters. One first explanation for this difference is the rate values distribution, as mentioned before.

The trope use analysis pointed to no expressive difference between methods using or not these extra content information. Some explanations for this results is that the tropes as much as they seem a rich source of information, there are a lot of different kinds of tropes and they not occur a number of times enough in the databases.

6.1 Future Work

Some future works that should be considered:

- Add tropes data in Matrix Factorization method to verify the impact of the additional data;
- Analyze the relations between tropes and genres if their occurrences have some correlation to understand better how this data can be used;
- Test different kinds of pre-filtering the tropes, once there is a lot of different tropes, maybe not all this data is really useful;
- Explore ways to uniformity the number of tropes by item because currently, the tropes distribution by item is nonuniform;
- Explore how tropes can be used to explain recommendations to users, it is one good topic inside Recommender Systems field where tropes probably can be well used.

References

AGGARWAL, C. C. *Recommender Systems: The Textbook*. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2016. ISBN 3319296574, 9783319296579.

BATISTA, A. F. et al. *Utilizando Tropes em modelos de recomendação híbridos*. Dissertação (Mestrado) — Programa de Pós-graduação em Informática - Universidade Federal do Amazonas, 2016. Instituto de Computação. Disponível em: <http://tede.ufam.edu.br/handle/tede/5617>.

BOBADILLA, J. et al. Recommender systems survey. *Knowledge-based systems*, Elsevier, v. 46, p. 109–132, 2013.

CHANG, S.; HARPER, F. M.; TERVEEN, L. Using groups of items for preference elicitation in recommender systems. In: *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work &#38; Social Computing*. New York, NY, USA: ACM, 2015. (CSCW '15), p. 1258–1269. ISBN 978-1-4503-2922-4. Disponível em: <http://doi.acm.org/10.1145/2675133.2675210>.

CHEN, P.-T. et al. Mobile advertising setting analysis and its strategic implications. *Technology in Society*, Elsevier, v. 39, p. 129–141, 2014.

CLEMENTS, J.; MCCARTHY, H. The anime encyclopedia: Revised & expanded edition. *A guide to Japanese animation since 1917*, 2006.

CUI, Y. et al. *Content-targeted advertising using collected user behavior data*. [S.l.]: Google Patents, jan. 27 2005. US Patent App. 10/649,585.

GEMMIS, M. de et al. Integrating tags in a semantic content-based recommender. In: *Proceedings of the 2008 ACM Conference on Recommender Systems*. New York, NY, USA: ACM, 2008. (RecSys '08), p. 163–170. ISBN 978-1-60558-093-7. Disponível em: <http://doi.acm.org/10.1145/1454008.1454036>.

GUPTA, U.; PATIL, N. Recommender system based on hierarchical clustering algorithm chameleon. In: IEEE. *Advance Computing Conference (IACC), 2015 IEEE International*. [S.l.], 2015. p. 1006–1010.

HARPER, F. M.; KONSTAN, J. A. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, ACM, New York, NY, USA, v. 5, n. 4, p. 19:1–19:19, dez. 2015. ISSN 2160-6455. Disponível em: <http://doi.acm.org/10.1145/2827872>.

HERLOCKER, J. L. et al. An algorithmic framework for performing collaborative filtering. In: ACM. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.], 1999. p. 230–237.

HU, Y.; KOREN, Y.; VOLINSKY, C. Collaborative filtering for implicit feedback datasets. In: IEEE. *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on.* [S.l.], 2008. p. 263–272.

ISINKAYE, F.; FOLAJIMI, Y.; OJOKOH, B. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, v. 16, n. 3, p. 261 – 273, 2015. ISSN 1110-8665. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1110866515000341>.

KIESEL, M.; GRIMNES, G. A. Dbtropes—a linked data wrapper approach incorporating community feedback. In: CITESEER. *Proceedings of EKAW*. 2010. Disponível em: <http://skipforward.opendfki.de/wiki/DBTropes>.

KOREN, Y.; BELL, R. Advances in collaborative filtering. In: *Recommender systems handbook*. [S.l.]: Springer, 2015. p. 77–118.

KOREN, Y.; BELL, R.; VOLINSKY, C. Matrix factorization techniques for recommender systems. *Computer*, IEEE, v. 42, n. 8, 2009.

LEE, D.; HOSANAGAR, K. Impact of recommender systems on sales volume and diversity. 2014.

LEMIRE, D.; MACLACHLAN, A. Slope one predictors for online rating-based collaborative filtering. In: SIAM. *Proceedings of the 2005 SIAM International Conference on Data Mining*. [S.l.], 2005. p. 471–475.

MACWILLIAMS, M. W. *Japanese visual culture: Explorations in the world of manga and anime*. [S.l.]: Routledge, 2014.

MIYAKE, L. *Anime from Akira to Princess Mononoke: Experiencing Contemporary Japanese Animation*. [S.l.]: JSTOR, 2002.

NGUYEN, T. T.; RIEDL, J. Predicting users' preference from tag relevance. In: CARBERRY, S. et al. (Ed.). *User Modeling, Adaptation, and Personalization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 274–280.

OZGEN, C. *An Ontology-based Hybrid Recommendation System using Semantic Similarity measure and feature weighting*. Tese (Doutorado) — Middle East Technical University, 2011.

PASSANT, A. Measuring semantic distance on linking data and using it for resources recommendations. In: *AAAI spring symposium: linked data meets artificial intelligence*. [S.l.: s.n.], 2010. v. 77, p. 123.

POLATIDIS, N.; GEORGIADIS, C. K. Recommender systems: The importance of personalization in e-business environments. *International Journal of E-Entrepreneurship and Innovation (IJEEI)*, IGI Global, v. 4, n. 4, p. 32–46, 2013.

PURCELL, K.; RAINIE, L.; BRENNER, J. Search engine use 2012. Pew Internet & American Life Project Washington, 2012.

REYSEN, S. et al. An examination of anime fan stereotypes. *The Phoenix Papers*, v. 2, n. 2, p. 90–117, 2016.

RIBEIRO-NETO, B. et al. Impedance coupling in content-targeted advertising. In: ACM. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval.* [S.l.], 2005. p. 496–503.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: *Recommender systems handbook.* [S.l.]: Springer, 2011. p. 1–35.

SARWAR, B. et al. Item-based collaborative filtering recommendation algorithms. In: ACM. *Proceedings of the 10th international conference on World Wide Web.* [S.l.], 2001. p. 285–295.

XUE, H.-J. et al. Deep matrix factorization models for recommender systems. In: *IJCAI.* [S.l.: s.n.], 2017. p. 3203–3209.

YANG, R.; HU, W.; QU, Y. Using semantic technology to improve recommender systems based on slope one. In: *Semantic Web and Web Science.* New York, NY: Springer New York, 2013. p. 11–23. ISBN 978-1-4614-6880-6.