Collaborative Filtering and Content Based Models (NLP) for Amazon Video Games Rating Predictions

Huayue Li **Ziyi Zhong**
hul036@ucsd.edu **z7zhong@ucsd.edu**

1. Introduction of Dataset

These days, Amazon has collected the comments of customers to products into a dataset. This dataset is interesting and worth to study. Therefore, in our project, we use the dataset of Amazon Product Reviews. Because this dataset is so large and it has already divided into many groups based on product categories, we decide to choose one category of product to analyze in our project. The category we decide to choose is "Video Games".

There are 2565349 product reviews for "Video Games" in total. Therefore, we think this dataset is large enough to study. All the information is stored in a dictionary and for every review, it contains the following information: (1) $overall$: the rating of the product (a float which range from 1.0 to 5.0 (rating of product from low to high)). (2) $verified$: whether this review is verified or not (a bool which is whether true or false). (3) $reviewTime$: the time when the review was made (a string to represent time). (4) $reviewerID$: the ID of the reviewer who made this review to distinguish from others (a string to represent the reviewer). (5) $asin$: the ID of the product (a string to represent the product). (6) $reviewerName$: the name of the reviewer who made this review (a string to represent the reviewer). (7) $reviewText$: the comments made by the reviewers to the product in detail (a string to represent comments). (8) $summary$: the summary of the comments made by the reviewers to the product in short (a string to represent summary of comments). (9) $unixReviewTime$: time of the review (unix time) (a integer to represent time). One example of the review is shown as follows:


```
{'overall': 1.0,
 'verified': True,
 'reviewTime': '06 9, 2014',
 'reviewerID': 'A21ROB4YDOZA5P',
 'asin': '0439381673',
 'reviewerName': 'Mary M. Clark',
 'reviewText': 'I used to play this game years ago and loved it. I found this did not work on my computer even though it said it would work
with Windows 7.',
 'summary': 'Did not like this',
 'unixReviewTime': 1402272000}
```


Fig 1 Datum Structure

In our exploratory analysis, there are some interesting findings:

(1) The reviews with "true" verified attribute is relatively trustable than the reviews with "false" verified attribute. For example, for one product, if a lot of reviews give low rating to it, another review with "true" verified attribute will always give low rating to it while another review with "false" verified attribute may give high rating to it. Therefore, we think to make our prediction task more trustable, in our training process, we will pick up the "true" verified reviews to train our model instead of the whole dataset

to make our model more available.

(2) We find that the rating for one product will change with time goes, which I mean the later reviews to one product may be a little different from the earlier ones. Therefore, we think in our predictive task, we prefer to use the latest reviews which is relatively more suitable.

(3) We find that there is a positive relationship between the comments and the rating of the reviews. In the comment of a review, if the feedback in the comment is positive, the rating is relatively high. When the feedback is negative, the rating is relatively low. Therefore, we can make our prediction task to be predict rating based on the sentiment analysis of the $reviewText$.

(4) Since our data contains Time Stamp, our intuitive thought is to consider the influence of time. However, we select some popular items and analysis its average rating based on time, the influence is not linear as we thought (shown in fig 2). Since there are abundant context information and latent determining factors, we mainly focus on these fields and decide to skip temporal influence.

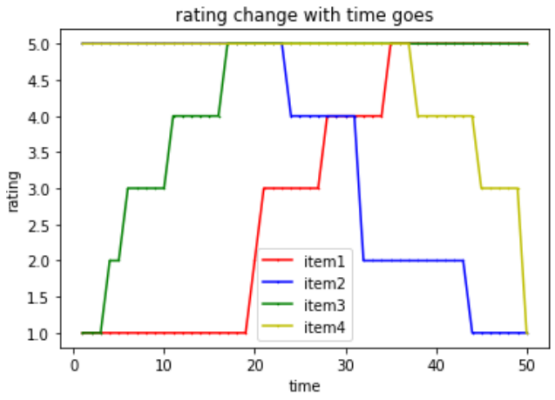

Fig 2 Ratings of Some Items based on Time

2. Predictive Task

In our project, we will establish two sets of recommender models for our task.

As we mentioned and have shown that the dataset contains users and items pairs, we would like to discover latent factors and build suitable models. Additionally, the datum is rich of context information and we want to dig in to search more interesting findings.

2.1 Collaborative Filtering

In this part, our team is attempting to discover users' preferences.

We will predict the rating of new reviews based on the past reviews. Here we will use three features: $overall$, $reviewerID$ and $asin$. These features represent rating, users, and items respectively. As mentioned in section 1, we shuffle the whole dataset, and divide them into training set, valid set and test set.

2.2 Content Based (NLP)

In this part, our team is attempting to discover content information.

We will make prediction of rating based on the comments of reviews and summary of comments of reviews. Here we will use three features: $overall$, $reviewerID$, $asin$, $reviewText$ and $summary$. One important thing is that some reviews do not have $reviewText$ or $summary$, so we will clean them out.

Besides, the data has a sorting problem that is the data are sorted in items ID order. This will probably lead to huge overfitting or inaccuracy issue. Hence, we will shuffle the dataset first.

For the two predictive tasks, we will divide the first 2300000 reviews to train, 200000 reviews to valid and the last 65349 to test. Finally, we will use the MSE to judge the accuracy of rating prediction.

Our baseline model is to make predictions according to the mean of ratings for all users and each user alone. This model makes prediction firstly based on user's own

mean rating with his history collected. Then if the user is new to model, the predicting rate would be mean of whole users

Since we have our baseline model and two sets of models, we will compare their performances both with baseline one and each other.

3. PREDICTIVE MODELS

Since our dataset is Amazon Product Review, our research aims to make analysis based on users' purchase history and their comment. Therefore, our goal is to establish a recommender model to predict the rating with user and item pair.

For baseline model, as we mentioned in section 2, is to make predictions according to the mean of ratings for all users and each user alone. This model makes prediction firstly based on user's own mean rating with his history collected. Then if the user is new to model, the predicting rate would be mean of whole users.

Besides that, our first intuitive thought is simple latent factor model. The simple latent factor model considers the influence of both users and items.

$$f(u,i) = \alpha + \beta_u + \beta_i$$

Based on that, our next step is using complete latent factor model and SVD, since both models have much more complexity than original simple latent factor models.

$$f(u,i) = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i$$

The formula is shown above. Comparing to simple latent factor model, this one assumes that there might be latent factors determining users' decisions.

Latent factor models can perform well since they can fully learn biases and preferences and find those potential determining factors.

On the second hand, since the data contains not only user's IDs, item's IDs and ratings, but also user's reviews and their summaries. Since the data contains abundant context information, our team also applied text mining models such as Word2Vec and Item2Vec.

The Item2Vec focuses on the similarity of the items. We assume the closer item ID shares the higher similarity because the ID represents the category and some related information. We calculate the similarity by using similarities of each item.

The Word2Vec however embedded the word to a vector and apply a regressor to train and predict a model.

While we apply and train our models, we encountered bunch of issues.

Firstly, we found that data are listed in a sorting order, which significantly increases the probability of the pairs of users and items being new to model. This will increase the case we predict by using rating mean. Plus, specifically, the text mining model will gain no similarities that makes the model collapsed. That is to say, we need to shuffle the dataset first to train models sufficiently

Secondly, since the dataset is huge, training usually takes the whole day, we shrink the training data, and then we encountered the issue of overfitting. The balancing seems a good means.

Another problem is our Word2Vec model performs really terrible at first with MSE of 10.1. Then we solved our problem by normalizing our vector.

Our team modified our model by setting different hyperparameters like the learning rate, the regression lambda, the dimensions of factors, the window size for text mining and the vector size. On some attempts, our NLP model even performs worse than just predicting as mean rating, since the model apparently learnt nothing. However, we still managed to train a good model that performs

well.

4. RELATED WORK

In recent times, the recommender systems have considerable importance in academia, commercial activities, and industry. They are widely used in various domains such as shopping (Amazon), music (Pandora), movies (Netflix), travel (TripAdvisor), restaurant (Yelp), people (Facebook), and articles (TED).

Our dataset is established by Amazon and has been widely studied. In some papers, the vision has been broadly used in recommender system [1], [2]. Visual data has been trained by using deep convolutional neural network to calculate similarities to make predictions and research on fashion

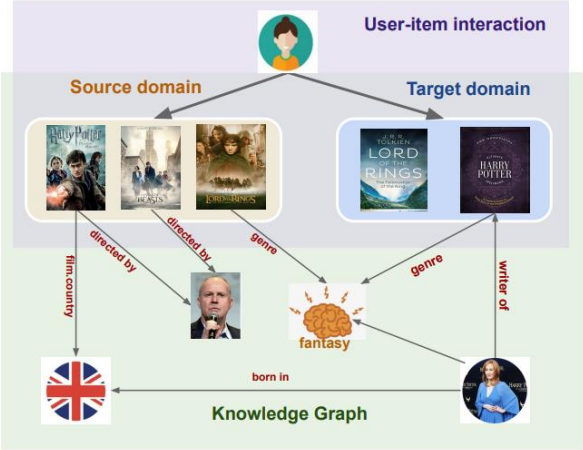

Fig 3 The Knowledge Graph

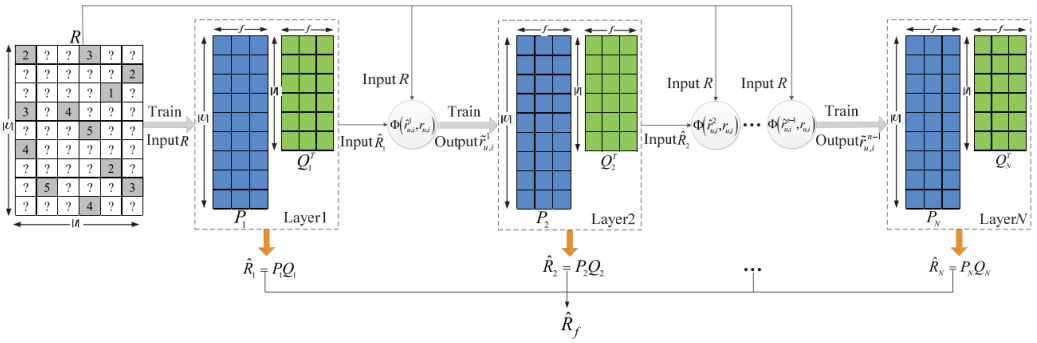

Fig 4 Deep Learning Method

trend. In specific domain like fashion, visual models can obtain better more useful information on users' preferences.

On the other hand, another team focus on cross-domain recommendation (CDR) [3]. In this paper, they proposed using a knowledge graph to let different domains share knowledge (shown in Fig 3).

However, our research mostly focuses on the rating predicting problem, therefore, we didn't pay much attention on the graph data or different domains. Instead, the key to our model is content information and biases for users and items.

For latent factor model's respective, there are a lot of great teams' work on SVD.

Since the data are huge and such a matrix is usually high-dimensional and sparse with users and items exploding, Wu proposed a deep a deep-structured model by sequentially connecting multiple latent factor (LF) models instead of multilayered neural networks through a nonlinear activation function (shown in Fig 4) [4].

The deep learning model is also studied by another team by introducing deep-structure in collaborative filtering model. Their work introduces the deep

latent factor model (deepLFM) [5]. They found that the dimensions of factors cannot be too low, which might cause the model misses some important features, and on the contrary, it cannot be too high as well, which might introduce too much complexity.

This thought is similar to ours, since we find that in a large data pool we need to appropriately set more factors to ensure our model have a better understanding of users' preferences.

On the other field, some teams specifically studied content information by using text mining.

Ghuribi [6] found that collaborative filtering (CF) like SVD is a popular group of methods employed to build effective traditional techniques, while a content-based (CB) approach mines the appropriate recommendations for a user based on his recent behaviors according to what the user liked, bought or watched. Thus, they manage to combine both methods to obtain more useful information as more as possible.

Another team focus on text mining techniques as well by using named entities and topic hierarchies to exploit meaningful information in users' comments [7].

5. CONCULUSIONS

In this paper, we propose two sets of models to predict potential ratings. Our method builds the models step by step. Firstly, the simple latent factor model has been applied and then complete latent factor model and SVD. To make influence analysis applicable in real applications, we continue attempting different hyperparameters. We further compare SVD with different n-factors.

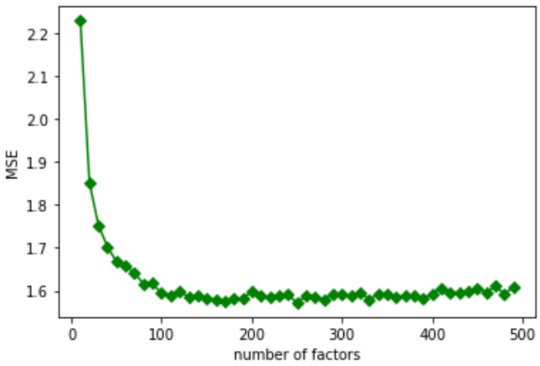

Fig 5 MSE of SVD on Valid Set

As shown in Fig 5, our conclusion is similar to the one proposed by Mongia. The more factors introduce more complexity, hence it probably extract more information.

Table 1 Models' Performance

Model	MSE	Feature
Baseline	2.428836	simple
Simple Latent Factor	1.946863	simple
Complete Latent Factor (K=2)	1.927704	More complexty
SVD (K=250)	1.572233	Much more complex, but costly

However, with dimensions increasing, when the factors approach 100, the improvement slows down. And even worsen slightly with over 300.

According to Ockham's Razor, it probably will become overfitting as it becomes more.

Another set is text mining models. Because of the abundant context information, we applied Item2Vec model and Word2Vec. Item2Vec compares the item and calculate the similarities. We found that the similarities between items improves the performance compared to the baseline and simple latent ones.

The Word2Vec focus on different aspect of content information. It embedded the word to a vector and we use a regressor to train and predict.

Table 2 Performance of Text Mining

Model	Baseline	Item2Vec	Word2Vec
MSE	2.428836	1.939668	1.410602

Apparently, as shown in Table 2, the Word2Vec seems much better because of more text information it exploits and learns. By applying word vector, we use a mean vector of all vectors for separate words to represent the whole review sentences can exploit users' sentiment and potential preferences.

All in all, with two sets of models, our team found that SVD performs better due to its high dimensions and ability to discover latent relations. The SVD preforms better with dimension of 220-270, since more factors means more complexity and more accuracy and means risk of overfitting.

As for NLP model, Item2Vec seems appropriate as a result of its ability to discover similarities, but Word2Vec seems much better because it extracts and exploit the knowledge behind the users' reviews.

REFERENCES

[1] Mcauley J, Targett C, Shi Q, et al. Image-based Recommendations on Styles and Substitutes:ACM,10.1145/2766462.2767755[P]. 2015.

[2] He R, Mcauley J. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering[J]. International World Wide Web Conferences Steering Committee, 2016..

[3] Li Zhang, Yan Ge, Jun Ma, et al. Knowledge-aware Neural Collective Matrix Factorization for Cross-domain Recommendation [J]. https://arxiv.org/pdf/2206.13255.pdf

[4] Wu, Di, et al. "A deep latent factor model for high-dimensional and sparse matrices in recommender systems." IEEE Transactions on Systems, Man, and Cybernetics: Systems 51.7 (2019): 4285-4296.

[5] Mongia, Aanchal, et al. "Deep latent factor model for collaborative filtering." *Signal Processing* 169 (2020): 107366.

[6] Al-Ghuribi, Sumaia Mohammed, and Shahrul Azman Mohd Noah. "Multi-criteria review-based recommender system–the state of the art." *IEEE Access* 7 (2019): 169446-169468.

[7] Domingues, Marcos Aurélio, et al. "Exploiting text mining techniques for contextual recommendations." *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*. Vol. 2. IEEE, 2014.