Grand Challenge:
Recommendation System of Hahow

Group 24

R11921063 吳少寶、**R11922056** 劉文心、**R11922096** 張家誠、**R11944014** 戴靖婷

Abstract

In this project, we have tried various algorithms on the four different tasks. We will introduce our best possible methods for each tasks, our experiments, and some discussions in the following sections.

Introduction

The goal of this project is to predict the courses or the subgroups that the users will purchase in the future based on the information of users and courses in Hahow.
There are four tasks which in combination of: seen users and unseen users, course prediction and subgroup prediction.

Related work

BERT is proposed in [1] as a powerful architecture to build various applications in Natural Language Processing (NLP). The basic idea of the BERT model is to learn the relationship between word tokens by randomly masking some tokens in a sentence. Based on the idea of BERT, Fei Sun *et al.* proposed a recommendation model [2] which can learn the relationship between interacted items of each customer. Since the dataset used in [2] seems large enough to get a good performance, we wonder whether a relatively small dataset like the one provided by Hahow can make the model well-trained in this project.

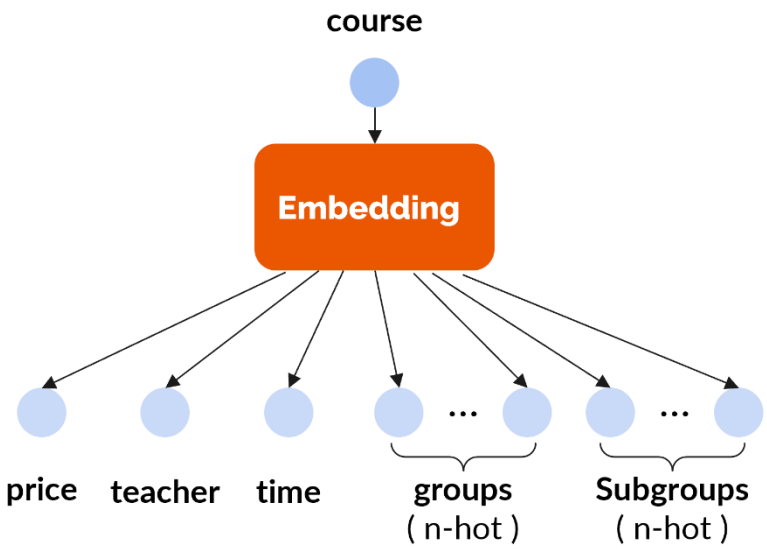
Approach

1. Seen user - Course prediction

Embedding

Since there may be lots of information related to each course, we can hardly find out the relationship between courses by their ID only. Therefore, embedding is used here to transform the courses into a high-dimensional space.

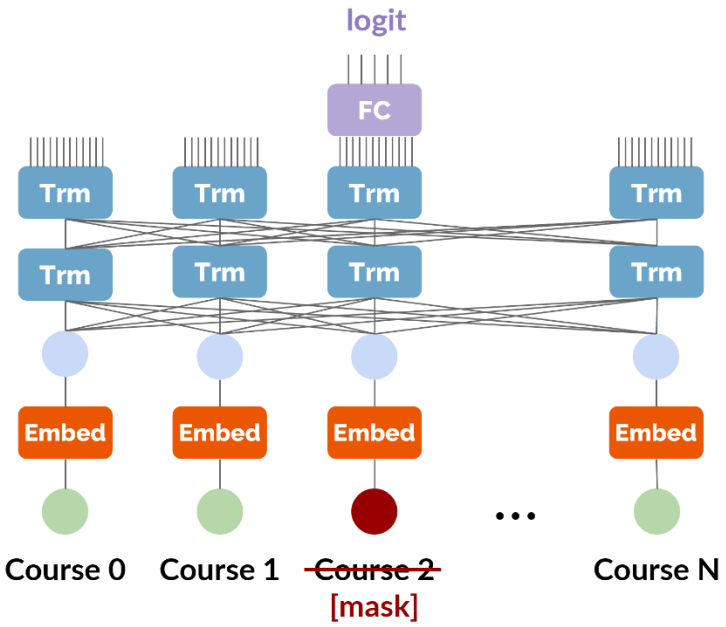
In the training dataset, we can obtain some attributes about each course, e.g. price, teacher, time and etc. Therefore, the courses can be mapped into points scattering in the space spanned by such attributes.



Model

To find out the relationship between courses, a Bert-liked model is used here. In the original use of Bert model, words in a sentence are partially masked and the transformer encoder layers will try to predict the masked token by learning the relationship between input tokens.

With such an idea, we take courses purchased by a user in the training dataset as the input tokens. After courses are embedded by the method we just mentioned above, through the following transformer layers and fully connected layers we can obtain the logit of each course, which can be used to classify which course is the masked token.



2. Seen user - Subgroup prediction

The methods we use for this task are **multilabel classification** and **alternating least squares**.

Data Preprocessing - Multilabel Classification

In multilabel classification, we first need to construct input and output for the model. For input, we make use of the user's interest, and only the subgroup part of interest will be reserved. After removing the group information, subgroup part of interest is concatenated with the user's recreation with comma. The output of the model is the prediction of course subgroup a given user will purchase, and the number of the subgroup can be 0 to 91.

E.g. Input can be "創業,平面設計,電腦繪圖,繪畫與插畫,刺繡,影像創作,手作小物,手作,插畫,插花,水彩".

Data Preprocessing - Alternating Least Squares

For Alternating Least Squares, we need to construct rating matrix. Hence, we first determine the rating of each subgroup for each user, and the format of each training data is "subgroup_id, rating, user_id" (e.g. 27, 4.0, '5bdecbfffec014002166796a').The most important part is how to evaluate the rating of each subgroup, and it will have a great influence on prediction performance. Hence, we consider the following two criteria:

- First, we will increase the rating of subgroup s by x if the course bought by the user belongs to the subgroup s. This process will be conducted for every course bought by the given user. Hence, if some subgroup of the course is bought many times, the score of that subgroup will be higher. By this way, we are able to know the importance of each subgroup for the user.
- In order to consider the subgroups that the user has not purchased but will likely purchase, we will increase the rating of the subgroup s by y if the interest of the user belongs to the subgroup s.

Method - Multilabel Classification

The following is the configuration of our model for multilabel classification. It is important that the problem type must be set to multi_label_classification. And the loss function we implement is binary cross entropy.

- Configuration
    - Model: bert-base-chinese
    - Architecture: BertForSequenceClassification
    - Problem type: Multi_label_classification
    - Hidden size: 768
    - Number of attention heads: 12
    - Number of hidden layers: 12

Method - Alternating Least Squares

The goal of alternating least squares if to find two lower dimensional matrix X and Y to approach the rating matrix R by matrix factorization. X is the matrix for user representaion, and Y is the matrix of subgroup representation. The ALS model will learn to factorize rating

matrix into user and subgroup representation, which allows model to predict better personalized subgroup ratings for users. With matrix factorization, the less-known subgroups can have rich latent representations as much as popular subgroups.

- Note: Although rating matrix R is a sparse matrix, matrix factorization can perform well on sparse matrix.

Prediction

For multilabel classification, the model will generate the probability for each subgroup, and we set a threshold to filter the subgroups that user is less likely to purchase. However, we cannot predict the order of the subgroups.
For alternating least squares, we will calculate the score of each subgroup for a given user. Then we can sort the subgroups by scores, and obtain the top k subgroups that user will be more likely to purchase.

3. Unseen user - Course prediction

The strategy we use for this task is the *best-seller recommendation*. In other words, we simply recommend popular courses to unseen users. Specifically, we generate recommended courses for an unseen target user based on the purchase history of other users who share one or more common interests with the target.

Building a Best-Seller Recommendation System

We use the training data and the “interests” column in users.csv to build a python dictionary whose keys are interests and values are python counters that count the number of purchases of different courses. For example, if an user has two interests (I1 and I2) and he/she has bought two courses (C1 and C2), the corresponding dictionary is as follows:

Key	Value
I1	Counter({C1: 1, C2: 1})
I2	Counter({C1: 1, C2: 1})

If another user has two interests (I2 and I3) and he/she has bought two courses (C2 and C3), the updated dictionary is as follows:

Key	Value
I1	Counter({C1: 1, C2: 1})
I2	Counter({C1: 1, C2: 2, C3: 1})
I3	Counter({C2: 1, C3: 1})

Predicting

There are two steps when predicting. First, if the target user has several interests, add all the corresponding counters together. Second, sort the final counter and pick the top 50 courses as our predictions.

4. Unseen user - Subgroup prediction

Data Preprocessing

Before tokenizing, we do character cleansing on every used data:

- Keep Chinese characters.
- Delete all the punctuations, English characters, and html format.

For Chinese tokenization, we use **CkipWordSegmenter** for word segmentation and **CkipPosTagger** for part of speech tagging. Besides, we set model to **albert-base** instead of bert-base.

After tokenizing, we do another tokenized data cleansing:

- Delete one-word segmentation since one-word often has no meaning in Chinese.
- Keep only nouns and verbs because they contain most of the meaning.

Method

We use BM25, an algorithm for querying a set of documents and returning the most relevant ones to the query, to predict courses first. Each document consists of course name, groups, groups, subgroups, topics, and description from courses.csv. Each query consists of the target user's interests and recreations from users.csv.

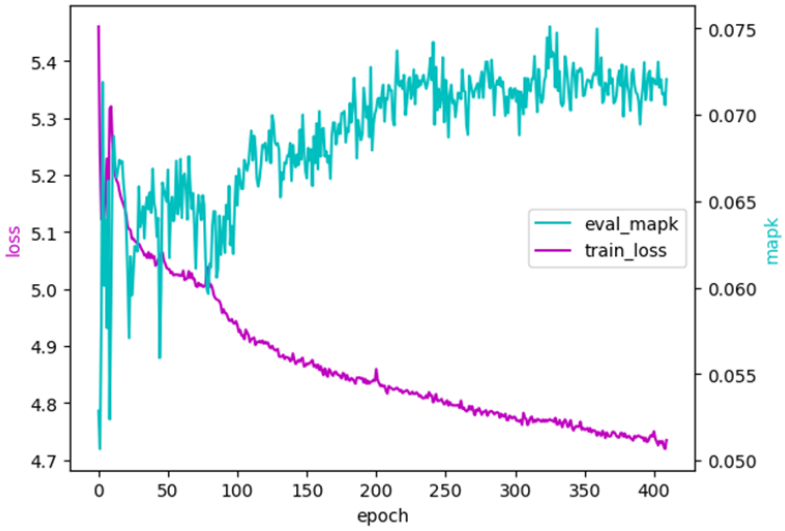
After BM25 predicts several courses, we use a voting system to predict the final topic by the subgroups of courses. The details of the voting system is as the following:

- Set the decay weight (W) and the decay step (S) to adjust the weight of the votes, which means every S courses, the voting weight is multiplied and reduced by W.
- e.g. W = 0.9, S = 5. The vote weights of the first five courses are 1, and the weights of the six to ten courses are 0.9.

Experiments

1. Seen user - Course prediction

As you can see in the plot, the evaluation score keeps increasing as the training loss decreases. Moreover, it seems like the training loss is still decreasing, which implies that the performance can further improve if we have more time to train our model.



Configuration:

Batch = 1024
Max Length = 32
dim_embed = 256
Transformer encoder
Num_layers = 2
Num_head = 6
Dropout = 0.1
Dim_feedforward = 2048
Loss function: Cross Entropy

2. Seen user - Subgroup prediction

In multilabel classification, our model doesn't perform well on test set. Although we replace the bert-base-chinese model with hfl/chinese-roberta-wwm-ext model, we still cannot get the higher score than 0.14. The following chart is our experiment on multilabel classification.

mAP@50	Test mAP@50
bert-base-chinese	0.13396
hfl/chinese-roberta-wwm-ext	0.13697

In alternating least squares, we tried different way to determine the rating and the different dimension of representation. First of all, we find that the model will have poor performance if the dimension of representation is too large. Hence, we set the dimension to be 10 and conduct the following experiments.

- The first one excludes the user's interest, and only analyza the subgroups of course purchased by a given user. Here the rating of subgroup will increase by 2 if the course purchased by the user belongs to the subgroup. The score we obtain on test set is 0.2372, and it has a great improvement compared with multilabel classification. (At the beginning, we tried to give the same score to each subgroup in train_group.csv without considering the possibility that a subgroup could be purchased by a user multiple times. However, we were unable to get higher score than 0.2 in our experiments, which shows that further analysis of the courses can effectively improve the importance of the subgroups.)
- The second one includes the user's interest in order to consider the subgroups that users may be interested in but have not yet purchased. And the rating of subgroup will increase by 1 if the user's interest belongs to the subgroup. It can be known from the experiments that there is good improvement compared to the first method. This suggests that there is a strong possibility that the interests contain subgroups that the user has not yet purchased.
- The third one is to increase the rating of subgroup by 2 if the user's interest belongs to the subgroup. This makes the weigth of interests same as the courses. And the score of this method is 0.27659, which is our best result.

Rating method	**RMSE**	**Val mAP@50**	**Test mAP@50**
X = 2 (with no interest)	0.127380	0.221538	0.2372
X = 2, Y = 1	0.332722	0.246324	0.2619
X = 2, Y = 2	0.461578	0.253977	**0.27659**

3. Unseen user - Course prediction

We tried to use different information about the users as the key of the python dictionary and generate different numbers of predictions. The experiment results are in the following table. As you can see, using the "interests" column alone will get the better score. In addition, the validation score improves when we generate more predictions.

mAP@50	Top 10	Top 20	Top 30	Top 40	Top 50
interests	0.06462	0.07122	0.07361	0.07504	**0.07593**
recreation_names	0.05682	0.06234	0.06463	0.06596	0.06668
interests + recreation_names	0.06362	0.07016	0.07252	0.07395	0.07484

4. Unseen user - Subgroup prediction

First of all, we have set some fixed parameters:

- n_course = 150, decay_weight = 0.9, decay_step = 5.

Besides, we have tried different data cleansing after tokenization:

- clean: deletes one-word segmentation.
- posclen: pos stands for part of speech tagging, and only keeps nouns and verbs.

Then, we have tried different models of BM25, i.e. BM25L, BM25Plus, and BM25Okapi. It is observed that even though BM25L works better on course prediction, BM25Okapi generates the best results on subgroup prediction.

Below is the mAP@50 on the validation data:

mAP@50	corpus, query	corpus_clean, query_clean	corpus_posclean, query_clean
BM25L	0.284447	0.284834	0.281998
BM25Plus	0.297724	0.298504	0.299091
BM25Okapi	0.302589	0.303644	**0.304404**

In another experiments, we have set some the algorithm and cleansing documents:

- algorithm=BM25Okapi, corpus=corpus_posclean, query=query_clean.

In order to test the voting system, we have tried several combinations. We may observe that the different number of courses performs better with different decay weights and decay steps. For better performance, the smaller the predicted number of courses, the quicker the decay frequency should be and the lower the decay weight.

- Set the number of courses to 100, 150, and 200.
- Do a lot of experiments with distinct combinations of decay weights and decay steps.

Below is the mAP@50 on the validation data:

mAP@50	weight = 0.8 step = 10	weight = 0.9 step = 5	weight = 0.95 step = 2
n_course = 100	0.303738	0.303403	0.307124
n_course = 150	0.304534	0.304404	**0.308726**
n_course = 200	0.304878	0.304601	0.305770

Discussion

1. Embedding methods

In the previous section, we take the given course attributes to embed the courses into high-dimensional space. However, the information may not be rich enough to describe all the features of the courses well. In order to find out whether there is a better embedding method that can help the model perform better in telling the difference or similarity between the courses, we then try several additional methods. Here we further discuss the effect of different embedding methods.

- **Attribute embedding**
    - As mentioned above, such attributes are given in the dataset, which is probably labeled manually by the dataset provider. However, the courses can have more features than the number of attributes, and the attributes are not guaranteed to be rich enough to describe the features.
- **One-hot embedding**
    - Since the number of courses is not incredibly large, one-hot encoding can be applied directly without encountering the curse of dimensionality.
- **Linear embedding**
    - We can use a linear layer to map the courses into a space with a specific number of dimensions. The benefit of using this method is that the weights of the embedding layer can be updated while gradient descent, which makes it possible to fit the data better by learning together with the transformer layers.
- **Hybrid embedding**
    - By concatnating multiple types of embbeding result, the model can get the important information naturally. However, the performance may not be better if too many unnessary data are included in the embedding result.

2. Comparison between unseen tasks

The subgroup prediction of the unseen users are also generated from course prediction. Therefore, we have also compared the performances of the best model for each of them.
Surprisingly, the subgroup prediction does not always perform better when the course prediction performs better.

- For example, BM25Okapi performs better on subgroup prediction while BM25L performs better on course prediction. It might be because some courses are very popular that everyone wants to purchase, but the subgroups are not suitable to the user's interests or recreation.
- Therefore, we have also tried predicting courses by probability. It works better on course prediction, but it only gets about 0.18 in mAP@50, which is much lower than the result of BM25.

Conclusion

We propose four different methods to solve the four different tasks in this project. For seen user - course prediction, we train a deep learning-based recommendation model. For seen user - subgroup prediction, we use the Alternating Least Square algorithm. For unseen user - course prediction, we build a best-seller recommendation system. Finally, for unseen user - subgroup prediction, we adopt Okapi BM25 algorithm. Although our performance is not very outstanding, we have learned a lot during the process of exploration.

Work Distribution

- R11921063 吳少寶：Seen user - Course prediction
- R11922096 張家誠：Seen user - Subgroup prediction
- R11922056 劉文心：Unseen user - Course prediction
- R11944014 戴靖婷：Unseen user - Subgroup prediction

Reference

1. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova: “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, 2019
2. Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, Peng Jiang: “BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer”, 2019
3. ALS: https://github.com/mattzheng/pyALS