ADL Final Project Report - Hahow

許育騰
Department of CSIE
National Taiwan University
r11922003@ntu.edu.tw
洪郡辰
Department of CSIE
National Taiwan University
r11944050@ntu.edu.tw
黃禹喆
Department of CSIE
National Taiwan University
r11922110@ntu.edu.tw

January 5, 2023

Abstract

我們的 final project 是探討如何在 Hahow 線上學習平台上為用戶推薦他們可能感興趣的課程及類別，在我們的方法中，將 course prediction 及 topic prediction 都視為 multi-label classification 問題以方便實作。首先我們嘗試了簡單的 DNN model 並只考慮用戶的基本資料，在 topic prediction 任務上取得不錯的成績。之後我們使用自然語言處理中常用的 BERT fine-tuning 方法，將用戶基本資料或課程描述經過 BERT encode 生成 embedding，最後輸進 classifier 進行預測，此外我們也嘗試將 BERT 產生的最後一層 hidden state 作為 feature，計算各課程間的 consine similarity，推薦與購買過課程相似度高的課程給用戶，實驗結果為上述兩種方法的 aggregation 有比較好的效果。最後我們嘗試在 BERT fine-tuning 方法基礎上套用 Progressive Layered Extraction (PLE) 的技術，使不同任務的 model 之間可共享一些參數資訊，實驗結果為 PLE 在 unseen course 有比 BERT fine-tuning 有更好的結果，但其他 task 則沒有進步。

1 Introduction

我們的主題是 Hahow，針對 Hahow 所提供的使用者資料預測使用者的購買情況。預測共分為四種模式：

- Unseen topic: test dataset 中的使用者並未出現在 train dataset 中，預測前五十名可能購買的 subgroup。

- Seen topic: test dataset 中的使用者出現在 train dataset 中，預測前五十名可能購買的 subgroup。
- Unseen course: test dataset 中的使用者並未出現在 train dataset 中，預測前五十名可能購買的 course。
- Seen course: test dataset 中的使用者出現在 train dataset 中，預測前五十名可能購買的 course。

使用者資料集中，每筆使用者資料有五種 category，分別為:

- user id
- gender
- recreation names
- interests
- occupation titles

我們會將 gender、recreation names、interests、occupation titles 的資料作為 features，根據不同模型需求對這四種類型的 data 做前處理後，透過模型進行訓練。

在 Related Work 及 Approach 中，我們會使用不同的方法根據使用者資料及購買紀錄預測購買狀況，並且比較這些方法效果的差異並從中找出最佳的預測結果。

2 Related Work

2.1 Traditional Recommendation System

在深度學習興起之前，傳統的推薦系統主要可分成三類：

1. Contented-based filtering: 由商品的描述內容計算出兩種商品間的相似度，常見的作法是使用 TF-IDF (Term Frequency–Inverse Document Frequency) 得到可代表每個商品的向量，再計算兩兩商品間的 cosine similarity。系統會將與用戶曾經購買過的商品相似度最高的幾個商品，作為推薦選項。
2. Collaborative filtering: 主要考慮用戶與商品間的購買或評分記錄，從相似的用戶或商品來做推薦，常見的有兩種做法:
   (a) User-based collaborative filtering: 計算兩兩用戶的商品購買或評分記錄之間的相似度，系統會推薦與用戶相似度高的其他用戶所購買的商品。
   (b) Item-based collaborative filtering: 計算兩兩商品的用戶購買或評分記錄之間的相似度，系統會推薦與用戶買過的商品相似度高的其他商品。

3. Hybrid method: 混合以上兩種方法的推薦系統。

我們有嘗試在本次主題上使用 Contented-based filtering 預測課程，但結果不是很好 (kaggle public score: 0.0081)，而且只能用於 seen user，不過 cosine similarity 的概念也給我們之後的方法一些靈感。

2.2 Missing Value

在觀察使用者資料後，我們發現許多使用者資料都有缺值的情形，並不是所有資料都有完整的資料。在四種 category 中，我們發現 interests 與 subgroup 有著最強的關聯性，因此我們訓練了一個模型對 interests 欄位有缺漏的資料進行補缺值。interests 在使用者資料中共有 95 種。

以下是補缺值模型的相關設定：

- Model: bert-base-chinese
- Optimizer: AdamW
- Scheduler: linear
- Learning rate: 3e-5
- Loss: binary cross entropy

模型的輸入為欄位名稱加上欄位資料 (性別：...。職業：...。喜好：...。)，將每筆使用者資料串成一串文字。輸出為各個 interest 的機率，取 interest 分為兩種方式:

- Prob. softmax: 取 interest 機率最高的欄位進行補缺值。
- Prob. threshold: 取 interest 機率通過 threshold 的欄位進行補缺值，threshold 設定為 0.2。

我們之後使用了在 Approach 介紹的方法對補缺值後的資料集進行訓練，發現兩種補缺值方式在各個模型都沒有讓預測結果提升，反而下降了一些。我們認為可能是因為有進行補缺值的使用者資料中，有太多一樣的資料，造成有許多完全一樣的資料，使得模型效果下降。

3 Approach

3.1 DNN Training

首先我們先用簡單的 deep neural network (DNN) 作為 model，輸入用戶基本資料來預測 seen topic 和 unseen topic 這兩個任務。我們可以將 topic prediction 看成是一種 multi-label classification 問題，也就是對於每個 user，預測他最有可能購買哪些 subgroup 的課程。

我們會先預處理用戶基本資料，我們統計了 user.csv，得到 gender 有 3 種選項、interests 有 95 種選項以及 recreation names 有 31 種選項，總共有 129

個選項，我們便可用一個 129 維的向量去表示一個用戶，每一維對應到不同的選項，若用戶有某個選項，則在對應的那一維填上 1，否則為 0。

接著我們介紹所使用的 DNN model，此 model 是一個 fully-connected neural network，包含 input layer、hidden layer 以及 output layer，在 hidden layer 與 output layer 之間的 activation function 為 ReLU，output layer 後會再接一個 sigmoid function (為了與 loss function 搭配) 才是 model output。

在 training 時我們會使用 multi-label 的方式標示 label，loss function 則使用 binary cross entropy。而 model output 是一個 91 維的向量，代表用戶可能購買各個 subgroup 課程的分數。在 test 時，我們取前 50 高分數的 subgroup 由分數高排到低作為最後的預測結果。

3.2 BERT Fine-Tuning

對於 seen course, unseen course, seen topic, unseen topic 這四個任務，我們都將其視為分類問題，並嘗試了 multi-label 和 multi-class 兩種方法：

- Multi-label: 購買過的類別標為 1，其餘則為 0，loss function 為 binary cross entropy loss。
- Multi-class: 購買過的類別標為總購買數量分之一，其餘則為 0，loss function 為 cross entropy loss with probabilities for each class。

其他共通設定包含輸入只使用 users.csv 中所有的資訊，並會在各欄位資料前加上欄位名稱後串成一串文字 (性別：...。職業：...。興趣：...。喜好：...)；模型使用 bert-base-chinese [1]；訓練時採用 AdamW optimizer 和 linear decay learning rate scheduler；輸出為各個類別的機率，並把這些機率排序後取前 50 名即為最後提交的答案，而因購買過的課程不能被重複購買，所以會把買過的課程刪除。

此外，針對 seen course 和 seen topic，我們嘗試加入購買紀錄的資訊，分為測試時加入和訓練時加入：

- During test: 包含以下三種方法。
  - Input chain: 直接把購買紀錄按照前面的格式，串在輸入的使用者資訊後面再做預測。
  - Similarity matrix: 靈感來自 TF-IDF，只不過這邊是把課程名稱/課程子類別名稱輸入到前面訓練好的模型，並拿最後一層 hidden state 的平均來當作 embedding，再用這些 embedding 計算出 cosine similarity matrix，之後根據使用者的購買紀錄把對應的 row 取出來，並對每個 column 都找這些 row 之間最大的相似度，排序後一樣能得到一組結果。
  - Hybrid: 把前兩種方法得到的排序做 weighted rank aggregation。
- During training: 將購買紀錄切一半，一半加到輸入的使用者資訊後面，並用另一半或是原本全部的購買紀錄來當答案。

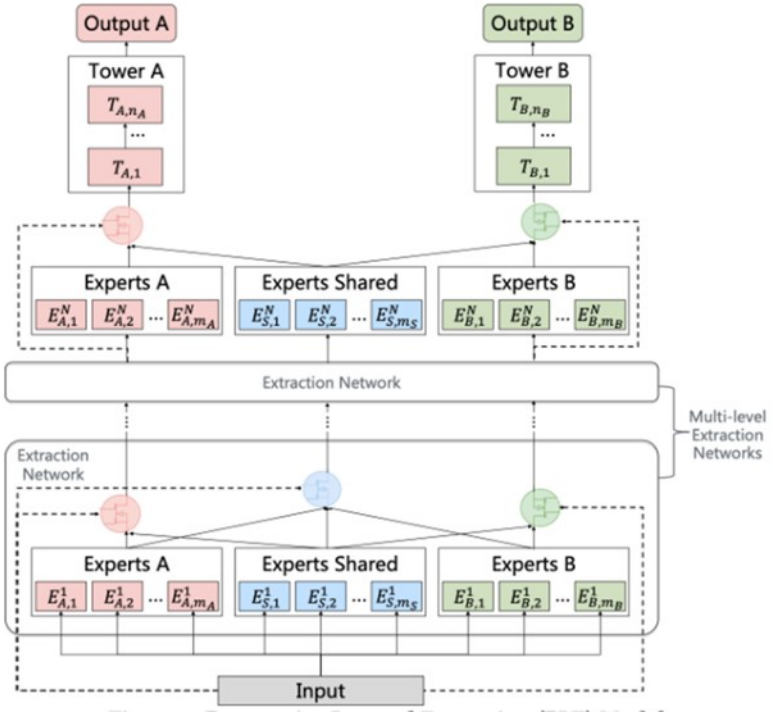

Figure 1: PLE model

3.3 PLE

Progressive Layered Extraction(PLE)[2] 是一個用於 multitask 的 model，它透過多層 extraction network 萃取 information。在 extraction network 中它將資訊分為 task-specific 及 task-shared 兩種模式進行學習及萃取。

在 Figure 1中，expert A 及 expert B 代表 task-specific imformation 的部分，expert shared 代表 task-shared imformation 的部分。在之後透過 expert block 上方圓形的部分進行資訊融合，圓形的部分被稱為 gate。Gate 也有分為 task-specific 及 task-shared，紅色及綠色圓形是 task-specific gate，藍色圓形是 task-shared gate。這兩種 gate 使用了不同的輸入，task-specific gate 使用了 expert A/B 的 information 及 expert shared 的 information 進行融合，task-shared gate 使用了所有 expert 的 information 進行融合。

在通過多層 extraction network 後，將萃取後的資訊作為 task-specific tower 的輸入，每個 tower 代表一種 task，最後透過 task-specific tower 從萃取後的資訊進行學習。這樣的架構能讓模型不會被過多 information 互相干擾，進而降低學習效果。

對於 topic 及 course 兩種任務，我將 model 設計成對每個 subgroup 及 course 進行二元分類的 multitask model：

- Subgroup: 91 tasks

- Course: 728 tasks

在 Feature 部分，我將 user.csv 中的 gender、recreation names、interests、occupation titles 視為 features，總共有 149 個 features，個別 feature 數量如下：

- gender: 3
- recreation names: 31
- interests: 95
- occupation titles: 20

在 label 部分，我使用了在 BERT 提到的 multi-label 和 multi-class 兩種方式進行訓練。

在訓練上設定上，我使用 Adam optimizer，loss 使用 binary cross entropy，透過改變 batch size 及 epoch 測試訓練效果。

在輸出部分，每個 task 的輸出為有購買這個 subgroup/course 的機率，我將這些機率排序後取前 50 名作為提交的答案。在課程部分，會將購買過的課程刪除。

4 Experiments and Discussion

4.1 DNN Training

訓練時所使用的 configuration 如下:

- Model: Fully-connected neural network with single hidden layer (256 nodes in the hidden layer)
- Batch size: 64
- Number of epochs: 200
- Optimizer: Adam (lr=3e-5)
- Loss function: Binary cross entropy loss

Learning curve 如 Figure 2所示。

Table 1: Public scores of 2 tasks using DNN

Task	Public score
Seen Topic	0.28074
Unseen Topic	0.32046

Table 1 是 DNN 在 Kaggle 上的表現，我們可以看到雖然 model 的架構很簡單，但仍有不錯的表現。這顯示就算沒有課程的詳細資訊，以用戶基本資料來預測未來會購買哪些 subgroup 的課程是可行的。

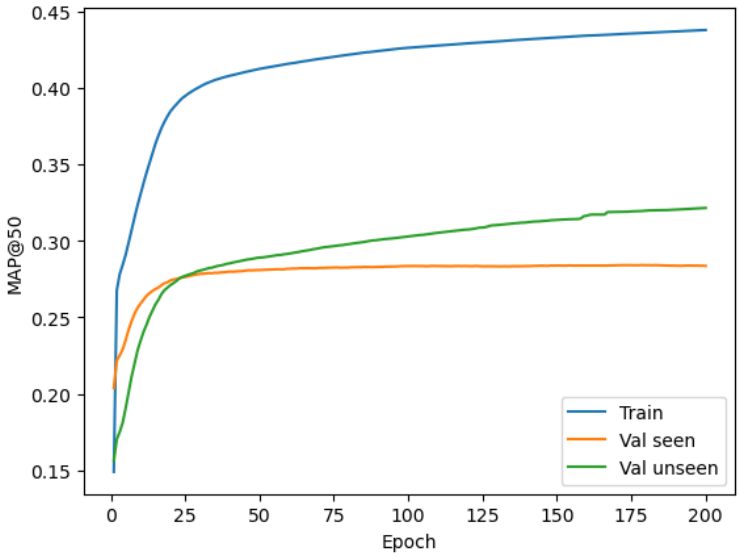

Figure 2: Learning curve of DNN training

4.2 BERT Fine-Tuning

Table 2: Public score of 4 tasks using multi-label and multi-class classification

Task	Multi-label	Multi-class
Seen Course	0.04436	**0.05115**
Unseen Course	0.05026	**0.07879**
Seen Topic	0.20368	**0.28562**
Unseen Topic	0.22670	**0.31197**

Table 2 呈現了 multi-label 和 multi-class 的結果，可以發現在這四個任務上，multi-class 的表現明顯都比 multi-label 還要好，我們認為可能原因為，multi-label 計算 loss 時是各個類別都會算 binary cross entropy loss，而 multi-class 只有 label 非 0 的類別才會產生 loss，因為現在這個任務並不需要太嚴格的限制，畢竟之前未購買過的之後還是有可能被購買，所以 multi-class 這種比較間接的方式就有機會得到更好的效果。

測試時和訓練時加入購買紀錄的結果在 Table 3，可以發現測試時加入的三個方法中 Similarity matrix 的表現最差，畢竟它只有使用到購買紀錄的資訊，而 Hybrid 的表現最好代表結合使用者和購買紀錄的資訊確實是有幫助的；此外，我們認為訓練時加入購買紀錄的資訊應該也是有幫助，只不過會讓輸入有出現過的類別對應機率明顯提升，seen course 的部分因為會把購買過的課程刪除，所以不會有太多負面影響，但 seen topic 並不會這麼做，這就導致輸出的多樣性降低，因此結果才會比測試時加入的方法差。

Table 3: Public score of 2 tasks when adding purchase record during test and training

Task	During Test			During Training	
	Input Chain	Similarity Matrix	Hybrid	Half + Half	Half + All
Seen Course	0.05200	0.03246	0.05394	0.05718	**0.05939**
Seen Topic	0.29175	0.23510	**0.29451**	0.11377	0.24988

Table 4: Public score of 4 tasks after increasing training epochs

Seen Course	Unseen Course	Seen Topic	Unseen Topic
0.06440	0.08823	0.31111	0.33523

我們根據前面的實驗結果，為四個任務挑選出效果最好的方法，並增加訓練的時間，最終的結果在 Table 4，可以發現分數又有再提升一點。

4.3 PLE

Table 5: Public score of 4 tasks using multi-label and multi-class classification

Task	Multi-label	Multi-class
Seen Course	**0.05762**	0.04233
Unseen Course	**0.0942**	0.06342
Seen Topic	**0.29009**	0.27541
Unseen Topic	**0.32464**	0.30403

Table 5 中呈現了 multi-label 和 multi-class 的結果，模型設定如下：

- model: PLE
- lr: 3e-5
- batch size: 512
- epoch: 10

可以發現在這四個任務上，multi-label 的表現都比 multi-class 還要好，我們認為可能是因為 multi-label 在 PLE 中比較能夠讓 expert 學到各個 task 之間的差異，因此 multi-label 效果比 multi-class 好一些。

Table 6: Public score of 4 tasks using 256 batch size and different epochs

Task	5 epoch	10 epoch	15 epoch
Seen Course	0.05341	**0.05349**	0.05281
Unseen Course	0.08032	**0.09098**	0.07351
Seen Topic	**0.27874**	0.25932	0.24751
Unseen Topic	**0.30129**	0.28912	0.26743

Table 7: Public score of 4 tasks using 512 batch size and different epochs

Task	5 epoch	10 epoch	15 epoch
Seen Course	0.04599	**0.05762**	0.05311
Unseen Course	0.08071	**0.0942**	0.092
Seen Topic	0.26637	**0.29009**	0.26782
Unseen Topic	0.2809	**0.32464**	0.26101

Table 8: Public score of 4 tasks using 1024 batch size and different epochs

Task	10 epoch	20 epoch	25 epoch
Seen Course	0.04654	**0.05419**	0.05408
Unseen Course	0.06403	0.09107	**0.09436**
Seen Topic	**0.27839**	0.26179	0.2485
Unseen Topic	**0.29956**	0.29243	0.27011

在比較完 multi-label 及 multi-class 的結果後，我們選用 multi-label 繼續我們的測試。我們開始測試不同 batch size 及 epoch 對結果的影響。在 Table 6、Table 7、Table 8中呈現了不同 batch size 及 epoch 訓練的結果。

可以發現在 batch size=256 時，因為收斂速度過快，很快地就進入過擬合，導致 epoch 上升反而使得準確率下降。在 batch size=512 時，四個任務在 epoch=10 時的分數都比 batch size=256 時的分數高，我們認為這是因為 batch size 擴大使得模型能夠更好地收斂。在 batch size =1024 時，因為收斂速度相較前面兩個 batch size 較慢，因此我們把測試的 epoch 調整為 10、20、25。可以看到分數比 batch size=512 時的分數低，我們認為是因為 batch size 過高，導致模型泛化能力不足，無法有效地學習。

Table 9: best Public score of 4 tasks using PLE and BERT

Task	PLE	BERT
Seen Course	0.05802	**0.06440**
Unseen Course	**0.09459**	0.08823
Seen Topic	0.29009	**0.31111**
Unseen Topic	0.32464	**0.33523**

Table 9中是 PLE 及 BERT 兩種方法的最佳結果。在我們的測試中，我們使用了 PLE 方法在 unseen course 得到了較好的成績。其他部分則都比 BERT 差。在 topic 部份，我們覺得可能是因為 PLE 參數不如 BERT 那麼多，因此效果比 BERT 差了一點。在 course 部份，我們覺得在 unseen course 部分表現較好可能是因為 course 之間的 relation 能夠被 expert shared 比較好地進行學習，而在 seen course 部分則是因為無法像 bert 一樣加入購買資訊，提升學習效果，因此表現稍微差了一些。

5 Conclusion

在本次專案中，我們針對 Hahow 的四種子問題做了許多實驗，結果顯示相較於傳統的推薦系統或是簡單的 DNN，使用預訓練好的 BERT 或是更專門的 PLE，是有機會表現得更好，尤其是針對 unseen course 這個最困難的任務 (沒有購買紀錄且類別數最多)，PLE 這種 multitask 的作法確實有助於共享不同任務間的參數資訊，進一步提升模型的泛化能力。

由於我們的實驗方法基本上都是基於深度學習的模型和技巧，並沒有太多資料分析和統計的部分，因此我們認為這是未來可以繼續嘗試的方向，或許透過統計的方式來找出所有使用者之間大致的購買趨勢，並直接加到或是使用 rank aggregation 來與模型預測的排序結合，就有機會得到更好的效果。

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[2] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Proceedings of the 14th ACM Conference on Recommender Systems*, RecSys '20, page 269–278, New York, NY, USA, 2020. Association for Computing Machinery.

A Work Distribution

- 許育騰：BERT Fine-Tuning

- 洪郡辰：Missing Value、PLE
- 黃禹喆：Traditional Recommendation System、DNN Training