EE655000 Machine Learning Final Project

馬拉松運動博覽會參訪動線類別預測

Group: Team18　Team member: 林蔭澤, 吳書磊, 羅宇辰, 涂皓鈞

Introduction

Analyzing the traffic flow of the people helps us understand the preferences and stay of visitors, thus making good use of the limited space and providing more appropriate information for those people with specific demands. The dataset for the issue was collected from 2018 Marathon Expo, and the visitors had been categorized into five types. In such considerable amount of data, how to train and establish a good decision model will be the challenge of this topic.

In this project, we mainly implement three kinds of models to perform the classification tasks, including 1) time-dependent models: Transformer, Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU). 2) time-independent model: Convolutional Neural Network (CNN), and Fully-Connected Neural Network (FC-NN). 3) conventional machine learning algorithm: eXtreme Gradient Boosting (XGBoost). The model architectures and design consideration of the above will be introduced in the following sections.

Framework

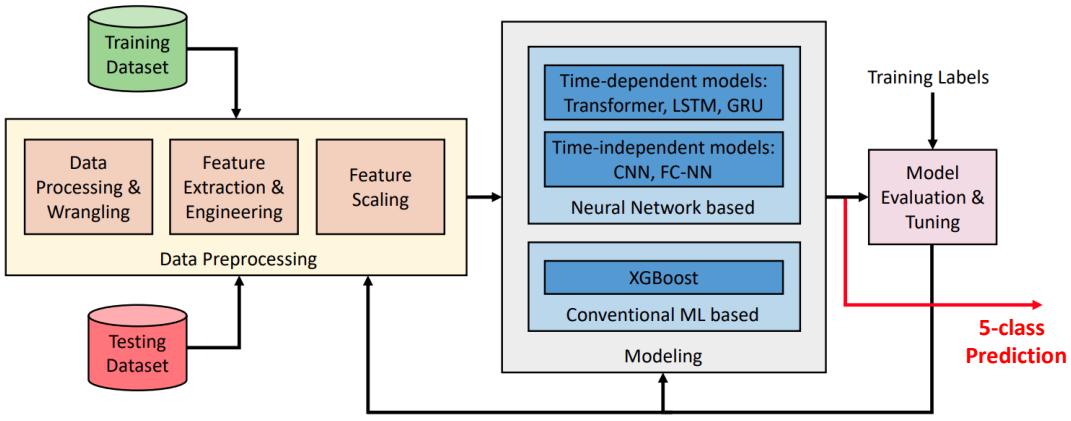


Fig. 1: Flow chart of our whole training framework.

Fig. 1 shows the flow chart of our training framework in this project, which can be divided into three parts: data preprocessing, modeling and evaluation. Data

preprocessing part basically organizes the raw data into the specific pattern, removes or alleviates the outlier in dataset, extracts and standardizes the feature, which will be introduced attentively in the next section. Modeling includes which models we choose, how we train the model. In this part, we utilize two different bases of models: neural network based and conventional machine learning based models to perform the classification tasks. In neural network based models, it can be parted into time-dependent and time-independent models. Finally, we use cross-validation to evaluate our model performance, tune the model parameters and adjust the input features we opt for back and forth to obtain our best model.

Data Preprocessing

The given training data set include following features:

- Sniffer location
- Created time (the time visitors show up on the sniffer location)

Considering that the existing features are not enough, in order to increase training accuracy, we create more features based on the given training data:

- Staying time

  We use create time to calculate how long in seconds the visitor spent on each location. Moreover, to make training performance better, we also do some data preprocess on the staying time:

  - Outlier removal and alleviation

    We use boxplots to detect the outliers present in the dataset. Boxplots depict the distribution of the data in terms of quartiles.

    First, we calculate the first and third quartile (Q1 and Q3), evaluate the interquartile range, IQR = Q3 - Q1, and then estimate the lower bound (the lower bound = Q1*1.5 * IQR), and the upper bound (upper bound = Q3*1.5* IQR). After calculating the lower bound and upper bound, we replace the data points that lie outside of the lower and the upper bound with lower and upper bound.

  - Standardization

    Standardization of a dataset is a common requirement for many machine learning tasks. Therefore, we use the function StandardScaler() from sklearn.preprocessing to normalize the dataset. This operation is performed feature-wise in an independent way.

- Location record

Location record feature records places visitor had visited. This feature will be an 1x14 vector. If the visitor had visited at least one time, we will mark the corresponding element in the vector to 1.

- Visitor preference

  We assume that different types of visitors have different preferences of booths. Fig. 2 and Fig. 3 show the marathon exhibition map, which was provided by AIdea and 2018 Marathon Expo. Based on the map, we can categorize booth locations into 7 types record in Table 1.

Fig. 2: Marathon exhibition map provided by AIdea.

Fig. 3: Marathon exhibition map provided by 2018 Marathon Expo.

	Booth locations
Type1	#1
Type2	#3
Type3	#2, #4, #5, #6
Type4	#7
Type5	#8, #13
Type6	#9, #10, #11
Type7	#12, #14

Table 1: 7 types of booth locations based on Fig. 2, 3.

With the help of Table 1, we can create extra feature to perform the classification task. For example, we can record the number of times visitors visited that type of booths.

Modeling & Evaluation

This part will introduce two bases of models:

- Neural network based models

  In this part, we want to know if the time-related input feature makes a difference with the predictive results. Our design consideration is quite straightforward: we divide the classification task into time-related and time-unrelated problems, simply building time-dependent and time-independent models to make an experiment.

  - Time-dependent models

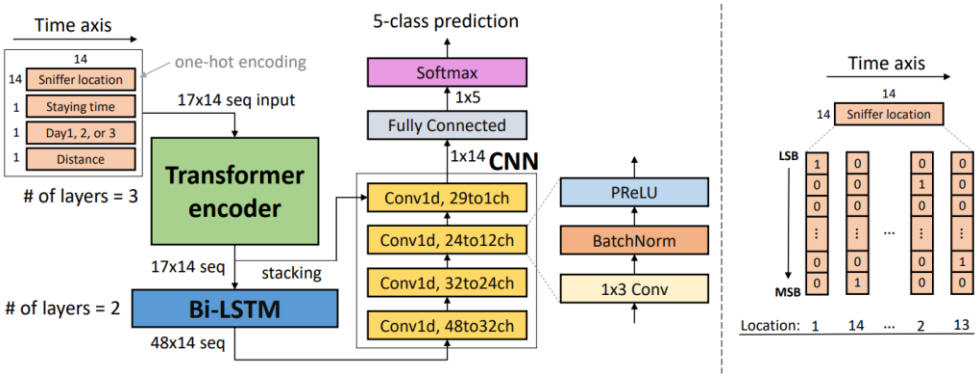


Fig. 4: Model architecture of our time-dependent model.

1. Model architecture

   Fig. 4 shows the model architecture of our time-dependent model. The proposed model is named as Transformer + LSTM. Inputs consist of time-related features, including sniffer location, staying time, which day visitors participated the activity and the distance between each sniffer

location. The above features are in chronological order. Fig. 4 on the right shows the schematic diagram of sniffer locations encoded by one-hot encoding. If the feature is not encoded, it would greatly affect the model performance.

Transformer + LSTM is comprised of a Transformer encoder, bidirectional LSTM and CNN. Transformer encoder enables the model learn the range of the receptive field automatically according to the different length of the input. Different from the traditional RNN, **Transformer is able to view all input sequence** for each component of the sequence due to its self-attention mechanism. Thus, its output features have much timing information. Next, **bidirectional LSTM can further help us find the relationship on the time axis among the sequence**. Finally, **CNN is mainly used to extract more high level feature and reduce dimension**.

2. Evaluation

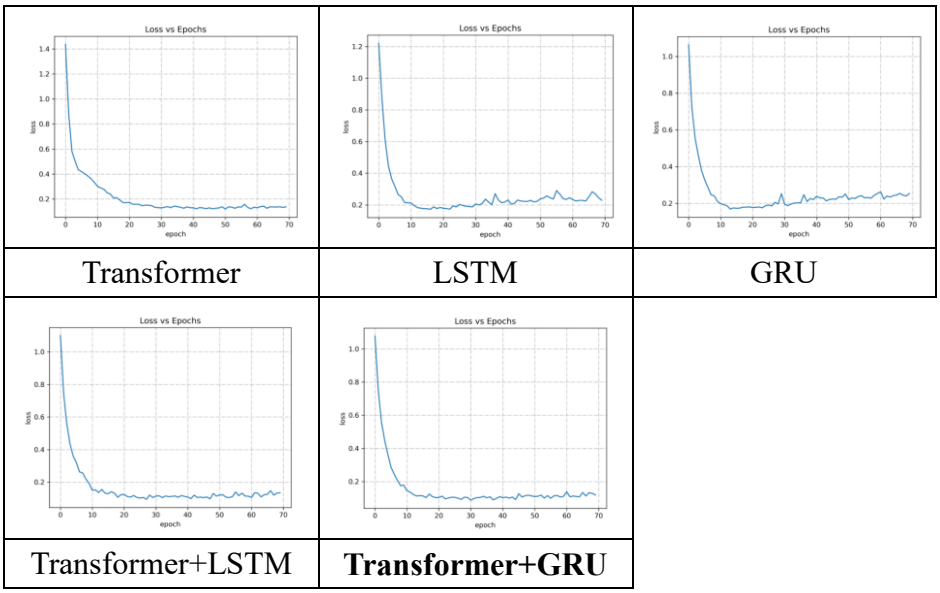


Fig. 5: Validation loss vs. epochs of time-dependent models.

	Validation Loss	Testing Loss (AIdea)
Transformer	0.120	0.12555
LSTM	0.173	0.17214
GRU	0.169	0.16984
Transformer+LSTM	0.094	0.09455
Transformer+GRU	**0.081**	**0.07988**

Table 2: Results of time-dependent models.

Fig. 5 shows validation loss vs. epochs across five different time-dependent models. The objective function is cross-entropy loss. **We use 10-fold cross-validation to evaluate the model performance.** Table 2 records the ablation study for different combinations of the three models: Transformer, LSTM and GRU. Our **best performance occurs in Transformer + GRU**, which will be discussed in the next section. It is also can be seen that LSTM and GRU are inclined to overfitting. **With an addition of Transformer, the overfitting situation can be alleviated**.

3. Discussion

   I think the reason why GRU performs better than LSTM is model complexity. As we know, LSTM has three gates: input gate, output gate and forget gate to make it memorable. While GRU only two gates: update gate and output gate. The update gate combines the input gate and forget gate from LSTM to control it memory. Thus, theoretically, GRU has less one third parameters than LSTM. The loss curve in Fig. 5 **implies that the model is apt to overfitting**, suggesting that **the dataset may not be complex or the model may be complex**. Considering the situation, GRU might have the advantage to learn better than LSTM. That's why Transformer + GRU outperforms Transformer + LSTM in this case.

■ Time-independent models

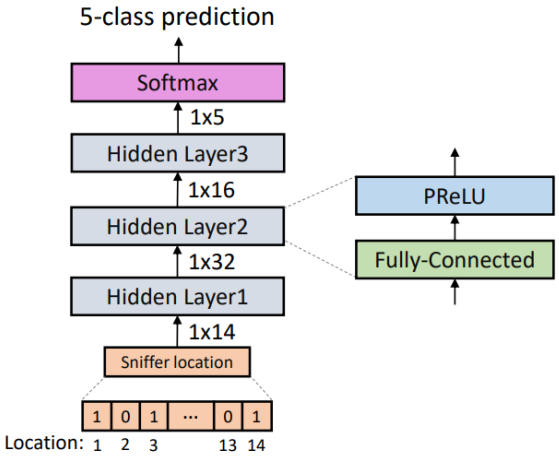


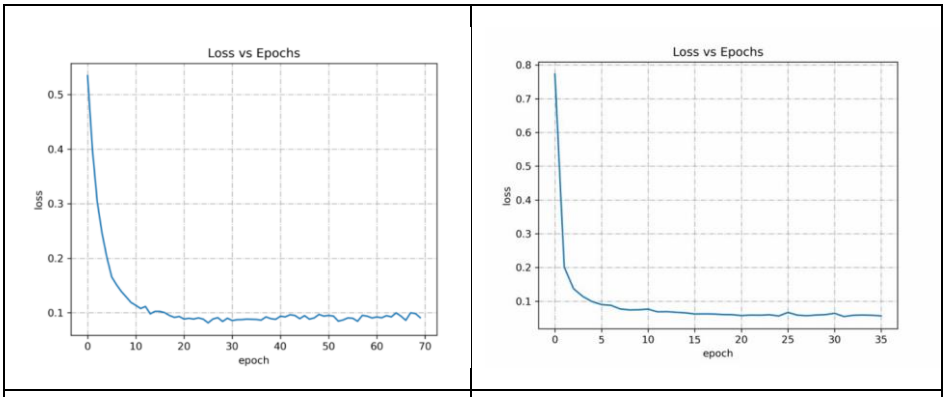
Fig. 6: Model architecture of our time-independent model.

1. Model architecture

   Fig. 6 shows the model architecture of the time-independent model. Since the network consists of fully-connected layers, we named it as FC-NN. Input feature records if sniffer locations had been visited as Fig. 6 shows (feature "location record"). Only three fully-connected layers are in the proposed model to prevent overfitting, and the activation function after

fully-connected layer is PReLU.

2. Evaluation



CNN	FC-NN

Fig. 7: Validation loss vs. epochs of time-independent models.

	Validation Loss	Testing Loss (AIdea)
CNN	0.081	0.08329
FC-NN	**0.043**	**0.04719**

Table 3: Results of time-independent models.

Fig. 7 shows validation loss vs. epochs across two time-dependent models. **We use 10-fold cross-validation to evaluate the model performance.** Table 3 records final results of two models. **Our best performance occurs in FC-NN**: **testing loss = 0.04719.** From the experimenting process, **we found that the model depth cannot be too deep, or the overfitting would be occurred**, further verifying that the statistic of the dataset is not complicated.

3. Discussion

From the evaluation section, it can be observed that FC-NN outperforms CNN. I think it is related to the network property. As we know, filters in **CNN are commonly shared within the same channel**. Basically, CNN is mostly used in complicated applications with large amount of data, such as image-related applications: super resolution, image deblurring, and so on. In this classification task, we **only have limited data**, so it seems that it's **not proper to use too much convolution layers**.

Compared with time-dependent models in the previous section, we can find that there is a large performance gap (testing loss: 0.07988-0.04719=0.03269) between them. From the result, we can conclude that **there is little relationship between the time-related feature and the**

predictive result. i.e., the type of visitors. Namely, **the features in the dataset are less time-related**, so time-independent models outperform time-dependent models.

- Conventional machine learning based models
  - XGBoost
    1. Description

       XGBoost is composed of many simple decision trees. Boosting aims to fix the worse prediction of old decision tree, and then combine them into a model. We generate some time-independent features to fit this model and predict the test accuracy of 5 classes. The features include location record, staying time, visitor preference and visitor cluster. Feature "visitor cluster" is choose the largest visit times according to 7 types of booth locations.

    2. Evaluation

Location record	✓	✓	✓	✓	✓	✓	✓	✓
Staying time		✓			✓	✓		✓
Visitor preference			✓		✓		✓	✓
Visitor cluster				✓		✓	✓	✓
Train accuracy	0.9925	0.9978	0.9938	0.9926	**0.9981**	0.9977	0.9933	0.9978
Train loss	0.0350	0.0174	0.0282	0.0337	**0.0132**	0.0145	0.0282	0.0135
Validation accuracy	0.9777	0.9702	**0.9818**	0.9752	0.9800	0.9693	0.9811	0.9795
Validation loss	0.0712	0.0938	**0.0596**	0.0717	0.0723	0.0923	0.0601	0.0736
AIdea Test loss	0.0852	0.1295	**0.0838**	0.0855	0.0873	0.1046	0.0848	0.0905

Table. 4: Ablation study for four different features in XGBoost

Table 4 shows the ablation study for four different features in XGBoost model, helping us know the influence to the model performance among the four features. Additionally, XGBoost provides "feature importance

plot" to support user how to choose importance features relative to classification significantly, as Fig. 8 shows.

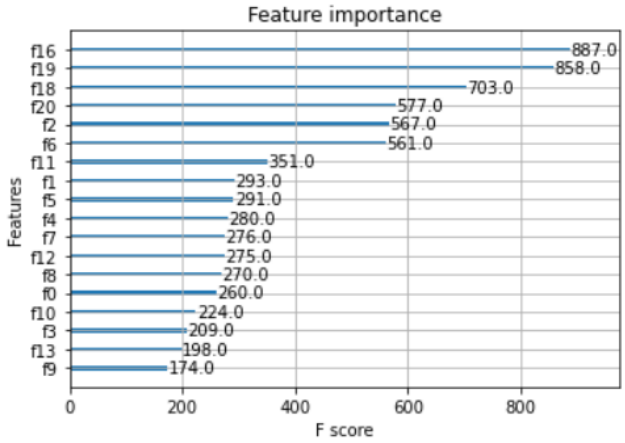


Fig. 8: Feature importance example.

3. Discussion

   From the result of Table 4, we choose **the feature "Location record" as our baseline feature. It can reach loss = 0.085 on Aidea testing.** We can find that features with location record and visitor preference achieve the best performance of validation accuracy and validation loss, it can reach loss = 0.083 on Aidea testing. Although model for features with location record, staying time and visitor preference can reach the best performance of train loss and train accuracy, but it's validation performance is not outstanding, so as to model for all features. Hence, **we can infer that feature "location record" and "visitor preference" contribute to classification significantly of this model.**

Summary

In this classification tasks, basically, we consider if the dataset is time-related to the predictive results, so we analyze two kinds of neural network based models: time-dependent and time-independent models. Among two kinds of models, Transformer + GRU and FC-NN perform the best, and their testing loss on AIdea is 0.07988 and 0.04719 respectively. From experimental results, we can conclude that complicated models could degrade the performance given the limited data, and the features in the dataset might be less time-related to the predictive results.

In the last part, we utilize the conventional machine learning method: XGBoost to make an extra experiment. From the ablation study of the four time-independent features, the feature "location record" and "visitor preference" contribute to the classification significantly, especially for the feature "location record", which further verifies that the

excellent performance for FC-NN in time-independent model. In a nutshell, from this project, we realize that the analyzing models with different properties and data preprocessing are utmost important. Doing such experiments, we understand that which models and features are suitable for this classification task. Finally, we achieve the lowest loss = 0.04719 in FC-NN model on AIdea as showed in Fig. 9.

Fig. 9: Our best testing loss on AIdea.