Multi-modal Movie Box Office Prediction

using poster, review along with tabular data

Team 24
Soyoung Yoon, Kyumin Park, Jooyeon Kim, Hangil Park
https://github.com/victory-jooyon/MM-movie-box-office-prediction

1. Introduction

A. Problem

On tasks like movie box office prediction or sentiment classification, we tend to look at one type of dataset source. However, recent studies ([1], [2], [3]) address the limitations of single-feature training and show that using a mixture of data types on training together improves the performance of the model. Specifically, work on extending transformers for image and text has advanced ([7]), and even pre-training methodologies for images and text has been addressed ([8] [9]). There is also work on taking all of audio, image, and text into account ([10], [11]). Therefore, by extending the idea of [7], we try to extend and improve the prediction of movie revenue by looking at both textual data (numerical indicators, textual features) and image data (movie posters). By doing so, since we can look at more information at once, we expect that the model performs better than just looking at single input, such as looking at only textual, or poster data.

Single-input model approach has limitations when training
→ Introducing a multi-input model approach for both image and text understanding

B. Motivation

In class, we learned about image captioning or visual question answering, which used a vision model along with an nlp model to solve tasks which was surely interesting. When our teammates gathered together for the team project, there were many candidates, but soon we were able to conclude that actually, everyone loves movies. Predicting movie revenue is especially important for movie workers (production companies, directors, etc). We thought that implementing & experimenting on a multi-modal model would be a great way to apply what we learned from the class, since we can experiment/preprocess both image and text data. Also, we thought that using multi-modal methodology for prediction would greatly help, or be a valid way, to improve the accuracy for the prediction.

2. Methodology

A. Approach

Use three feature-extractor models - one for movie posters, one for IMDB tabular data, one for movie overview from TMDB dataset. Then, taking into account all of the hidden state vectors from multiple feature extractor models, we finally get the prediction score which indicates whether the movie will succeed or not. As a prediction step we extend the model with MLP taking aggregated features as an input, and expected probability of success.

B. Baselines

We plan to utilize several pretrained feature extractor fit to each type of data. For image data, we use pretrained ResNet. Since ResNet[4] has good performance on general feature extraction, we can use ResNet as a poster feature extractor. We utilize the pretrained torchvision resnet18 model.

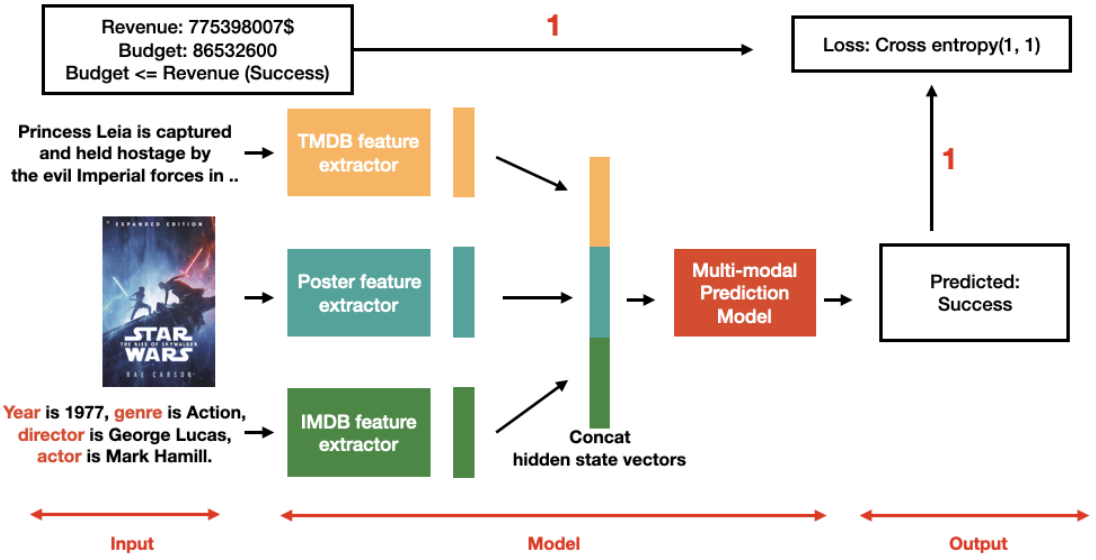
For linguistic data such as TMDB movie overview, we can use pretrained BERT ([5]) to represent each review into feature vector. BERT sequence classification task uses the first output vector in order to determine sequence characteristics. This task indicates that the output vector from BERT can represent sequence property, enabling us to use BERT as a feature extractor.

For tabular data (release year, genre, director) from IMDB dataset, we plan to preprocess tabular data into a sentence form (e.g. year is 1977, genre is Action, director is George Lucas, actor is Mark Hamil). Then, we use pretrained BERT as a representation model of our preprocessed tabular data. We have tried switching the feature extractor to a simple MLP, but using pretrained BERT as backbone model showed better accuracy.

A. Overview

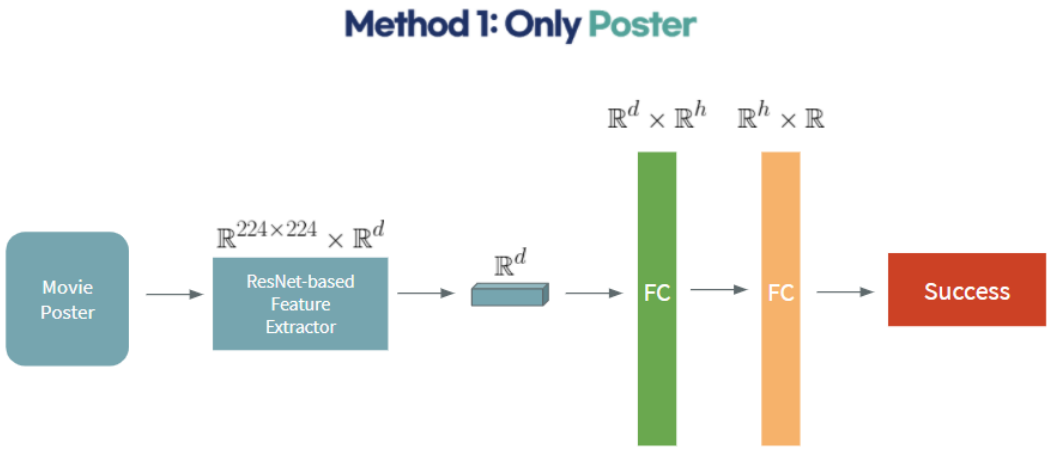
In the following section **we introduce our multi-modal movie success inference model**. We first introduce feature extractors for three data - poster, overview and extra movie information, then we propose our multi-modal

model. All of our implementations are based on pytorch. Especially, we used the huggingface transformers library for implementing most of our model [13]. We turned the gradient backpropagation off for three feature extractors since we use the already pretrained model (Resnet18 and BERT), and only trained the parameters for the fully connected layer when training. We also experimented on turning the gradients on for pretrained model, but turning them off showed better accuracy.



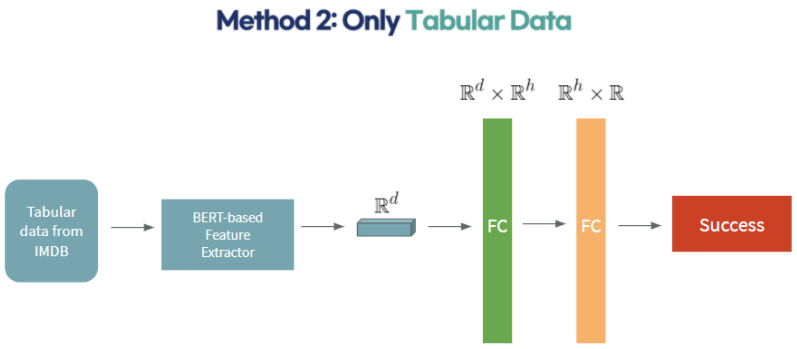
B. Poster feature extractor

For image data (which are movie posters), we make a feature extractor based on Resnet, and forward it with two fully connected layers to predict the success.



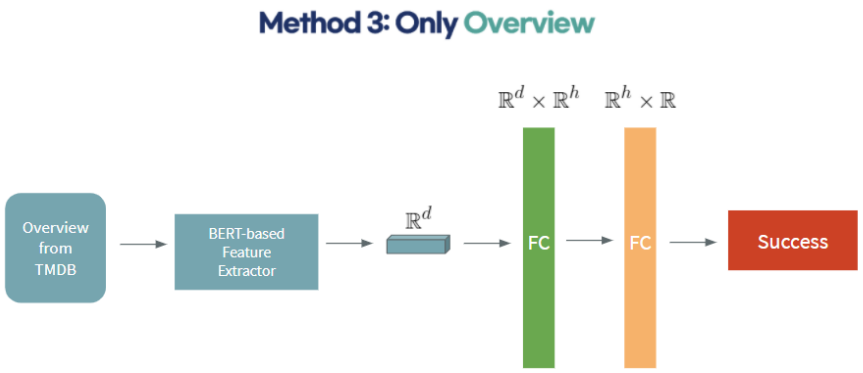
C. Overview feature extractor

For a single model with only tabular data from IMDB, We extract features by pretrained BERT model, and forward it with the same fully connected layer.



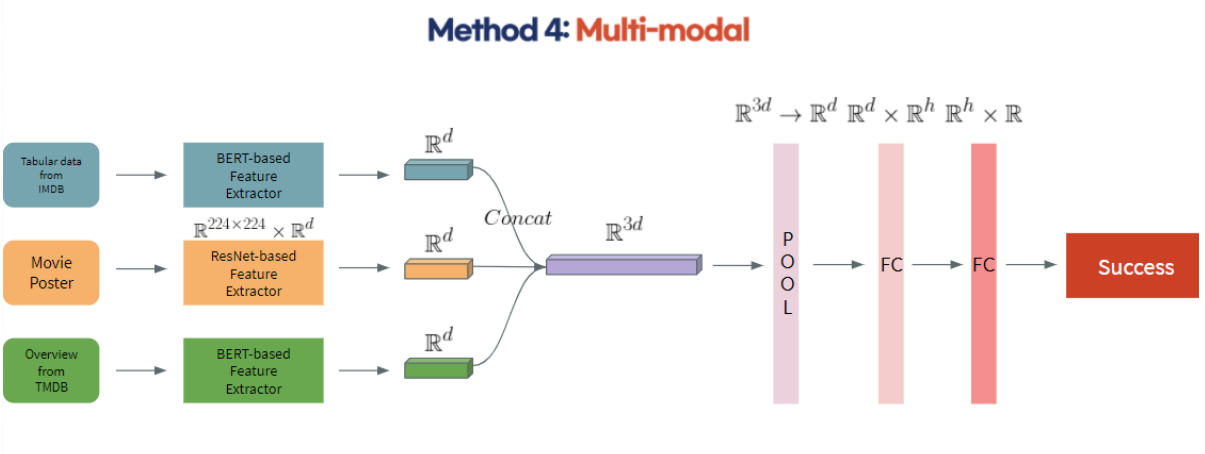
D. Overview feature extractor

For text features by TMDB, we also extract it with pretrained BERT, and forward it with fully connected layers.



E. Multimodal

This is our proposed multi-modal structure. We use the same feature extractor to output the same hidden vectors for all three data types. What is different is that we concatenate the three hidden vectors and forward it to the fully connected layer.



3. Experiment

A. Dataset

We will use the validation dataset to tune hyperparameters such as random seed, weight decay, learning rate, # of perceptrons for fully connected layers, etc.

- Use release year, director, main actor, and main genre from IMDB dataset. (https://www.imdb.com/interfaces/)
    - With director and main actor data, we made a particular ID for the person rather than using its name.

- Use poster image, overview, budget and revenue from TMDB. (https://www.themoviedb.org/documentation/api)
    - Crawled TMDB using the open API

We divide the available dataset into train, dev, and test sets, in a ratio of 8 : 1 : 1.

At first, we tried to use public datasets from Kaggle such as [12], but after validating the data, we found out that data having both budget and revenue data was too scarce, about 1000 movies while the dataset had 45000 movies total. Therefore, we crawled the data by ourselves. We crawled over 10 million movies, and after validation, we were able to work with about 7500 movies, which was 7 times larger than the public dataset.

B. Evaluation

The ground truth value of the gross of the movie is written as a "revenue" column on the TMDB dataset.
We evaluate the model on the validation & test set and get the output result of the "revenue" of the Input movie feature. We use "revenue" as the target feature.

At first, we tried predicting the raw value itself. This means, we tried to evaluate the performance of our model based on how close the model predictions are compared to the ground truth value (gross of the movie). But by this way, it was hard to correctly measure the effectiveness of the model. Therefore, as the professor commented out on the presentation, we change our evaluation criterion to a new evaluation strategy, by labeling (dividing) the predicted output into two classes: failure and success.

Then, we turn our classification task to predict whether the movie was a failure or success. We tag the data sample in the following rules: it is tagged "failure" if the revenue was smaller than the budget (revenue < budget), and tagged "success" if the outcome revenue was bigger or same as the breakeven point.(if revenue >= budget). By this way we could measure the model's accuracy. Before changing the strategy, we used the root mean squared error (RMSE) to calculate the loss. After changing the evaluation criterion to a classification task, we use the cross entropy loss when training and measure the accuracy for the test and valid set.

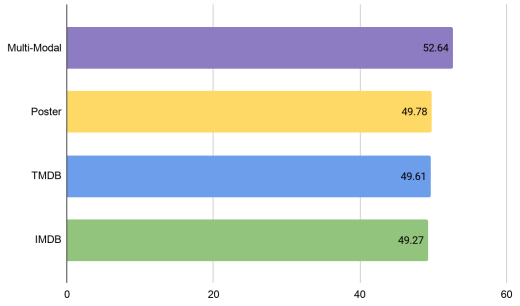
We train for 50 epochs, using the Adam optimizer with weight decay of 1e-04. For evaluating the test and valid set, we use the batch size of 32, and for the training set, we use the batch size of 16. We use a learning rate of 0.0005. All experiments are run on a single gpu of V100 model.

We evaluate our method on a total of 4 variations: one for evaluating accuracy of our proposed multi-modal, and 3 for evaluating our baseline models. Our three baseline models are the ones that use a single feature to predict the movie revenue. Comparing the accuracy of multi-modal and single-modal shows the effectiveness of our multi-modal compared with using only the single features. The three baseline model is the model with using:
1. only the IMDB features,
2. only TMDB features, and
3.only movie poster images.

By comparing accuracy results within those three, we can also find out which feature best contributed to the overall performance of predicting the total venue of the movie. We expect to prove the effectiveness of looking at multiple types of data by the increase in accuracy compared with best-performing single model and multi-model.

4. Results

A. Results



Above chart summarizes our prediction result on the binary-labeled movie dataset, compared with three single-modal baselines. Result shows that our multi-modal prediction model outperforms any other single-modal models, in a sense of test accuracy. Among the single-modal models, prediction with poster showed the best performance with test accuracy of 52.44%.

	Multimodal	Poster	TMDB	IMDB
Precision	0.61	0.45	0.44	0.51
Recall	0.58	0.60	0.55	1.0
F1 Score	0.59	0.51	0.49	0.68

We also report the precision, recall and F1 score for our prediction as additional metrics. These three statistical results locate around 0.5~0.6, which is around their accuracy value. Also, the multi-modal model reported better scores than single modal models, similarly to accuracy results. One exception is IMDB score, which reported a high precision score. This phenomenon may come from lack of feature representativity from IMDB data. This may be interpreted that the result from IMDB is biased, predicting only a single label.

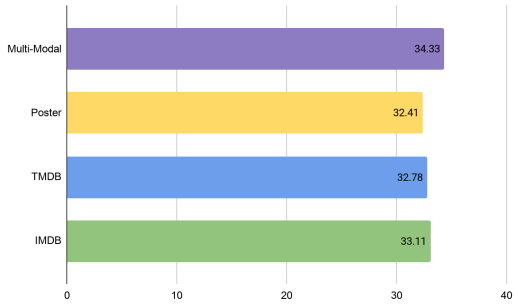


Table 2 shows the prediction result from a three-labeled task, which is divided into minus profit, profit lower than budget and profit more than budget. Following the tendency of the binary label task, our multi-modal model reported better accuracy than the other baseline models.

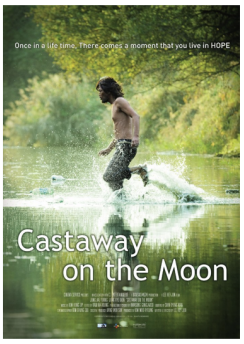
B. Analysis

The result showed our hypothesis of 'multi-modal model will have better performance on predicting the movie's success compared to single-feature model' is true. This is evidenced from higher test accuracy of multi-modal than any of single-feature models. Also, precision/recall/F1 score result shows that the models are not biased to one label. Since the gap between precision and recall factor is not large, we can conclude that the model can predict both failed and successful movies with similar performance.

Between the single-feature models, giving the poster image as the input generated the best performance. We assume it is because

- The difference of the backbone models. Poster image used the ResNet18 as the backbone feature extractor, while IMDB and TMDB used the BERT-base model as the backbone feature extractor. The ResNet18 model has a total of 11M parameters, while the Bert-base model has a total of 110M parameters - which is 10 times larger than the ResNet model. Compared with the fact that our task is relatively simple, we hypothesize the ineffectiveness of the bert-based model as the inherent over-complex structure of the backbone model. Maybe converting the feature extractor to a simpler model may exhibit better performance.
- The data from IMDB was not appropriate enough for training with BERT. We assumed that it is because BERT could not tokenize the names of actors and directors properly. Therefore, we changed the names of actors and directors to IDs, which would be much simpler and easier for BERT to analyze.
- Variance of poster data is larger than text data. Since each movie provides a highly variant quality of poster while their overview is similar, model prediction would depend on the poster more than other features.

Following figures show some examples inferred from our multi-modal model. Examples display that our model can predict movie success from non-direct information. Also, these figures, both predicting movie 'Castaway on the moon' with different posters, can be interpreted that the quality of poster affects model prediction, since probability of success increased when the poster was in seemingly better quality.



Label	Test Acc.
Success	54.52
Fail	45.48

Label	Test Acc.
Success	51.41
Fail	48.59

5. Discussion

A. Challenges

- Though IMDB and TMDB were big enough for training. However, data with both budget and revenue data were scarce. Korean movies, for instance, do not provide budget and revenue, which distracts those movies to be included in the dataset. If our goal was predicting the vote average (star rate) using regression, it would have been a better-performing model.

B. Future Work

- We expect an increase in accuracy and model performance by adding more related feature extractor. Thinking about it, people consider various factors (Review from a peer, looking at movie video advertisements) We expect our prediction to be more effective if we consider more features into account.
- If we explore more complex and better ways to concat the three feature vectors, we may expect an increase in accuracy. We have experimented on just concating the three vectors, applying max pooling after prediction, and applying average pooling after prediction. Among those, using average pooling got the best results. Maybe trying similar perturbations on how to concat the three hidden vector output would help increase the accuracy. For example, doing a weighted sum of the feature vectors before applying pooling may result in better accuracy.

C. Deliverables

- Crawled Dataset
  Data uploaded online having IMDB and TMDB related together and at the same time having budget and revenue data was below 1000 objects(rows). To make our model better, we implemented a crawler by ourselves, and collected over 7000 objects(rows). We plan to open this dataset later.

- Open-source Project
  We are opening our project public in Github for proving reproduction.
  (https://github.com/victory-jooyon/MM-movie-box-office-prediction)

6. Contributions

- Soyoung
  - Code
    - Implement TMDB feature extractor
    - Experiment on ablation studies (max pooling of concat layer, average pooling, change imdb model to mlp, add more layer to FC layer …) and report the result
    - Run training on various seeds and report result
  - Data
    - Run tmdb dataset crawling implemented by Jooyon
  - Report
    - Write evaluation, introduction, motivation, implementation details, analysis & future work.

    - Presentation
        - Present the final presentation. Made the overview picture. (multimodal model)
- Jooyon
    - Code & Data
        - Collect open dataset for movie data from Kaggle
            - Validated the data from Kaggle
            - Preprocessed data from Kaggle
                - Changing strings (ex. actor name) to integers (ex. actor ids)
            - Investigated on the data to discover patterns, analyze, and summarize the datasets (Exploratory data analysis)
        - Implement crawler for TMDB
            - Preprocessed data from TMDB
                - Getting the main genre (movies can belong to multiple genres)
                - Getting the main actor
        - Assist with implementing dataset loader
    - Report
        - Illustrated about the dataset and assisted on introduction and discussion

- Hangil
    - Code
        - Implement baseline models and the multimodal model.
    - Data
        - Assist with crawling data for TMDB.
    - Final presentation
        - Mainly work on motivation and methodology parts.
        - Illustrates each of our baselines and the multimodal model.
    - Report
        - Mainly work on methodology, analysis and discussion part.

- Kyumin
    - Code
        - Build Training pipeline
    - Train
        - Run train & report result
    - Report
        - Work on result / Analysis section

7. Reference

[1] Sierra, S., & González, F.A. (2018). Combining Textual and Visual Representations for Multimodal Author Profiling: Notebook for PAN at CLEF 2018.

[2] Christopher Bonnett, <Classifying e-commerce products based on images and text>, 2016-06-26, <http://cbonnett.github.io/Insight.html>, 2020-11-25

[3] Nicolas Audebert Catherine Herold Kuider Slimani Cédric Vidal. (2019). Multimodal deep networks for text and image-based document classification.

[4] K. He, X. Zhang, S. Ren and J. Sun. (2015). Deep Residual Learning for Image Recognition.

[5] Devlin, Jacob & Chang, Ming-Wei & Lee, Kenton & Toutanova, Kristina. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

[6] Pengcheng Yin∗ Graham Neubig, Wen-tau Yih Sebastian Riedel (2020). TABERT: Pretraining for Joint Understanding of Textual and Tabular Data

[7] Douwe Kiela et al., (2019) Supervised Multimodal Bitransformers for Classifying Images and Text, https://arxiv.org/pdf/1909.02950

[8] Weijie Su et al.. (2020) VL-BERT: Pre-Training of Generic Visual-Linguistic Representations, https://arxiv.org/pdf/1908.08530

[9] Jiasen Lu et al., (2019) ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks, https://arxiv.org/abs/1908.02265

[10] Yao-Hung Hubert Tsai et al., (2019) Multimodal Transformer for Unaligned Multimodal Language Sequences, https://arxiv.org/pdf/1906.00295

[11] Wasifur Rahman et al ., (2020) Integrating Multimodal Information in Large Pretrained Transformers, https://www.aclweb.org/anthology/2020.acl-main.214.pdf

[12] Kaggle, The Movies Dataset, https://www.kaggle.com/rounakbanik/the-movies-dataset

[13] The huggingface transformers library https://github.com/huggingface/transformers