Muti-label Movie and TV-series Genre Classification merging Resnet and LSTM Model using Title Poster and Plot Summary from IMDb

Mohammad Sabik Irbaz, 160041004
Computer Science and Engineering
Islamic University of Technology
Dhaka, Bangladesh
sabikirbaz@iut-dhaka.edu

Abir Azad, 160041024
Computer Science and Engineering
Islamic University of Technology
Dhaka, Bangladesh
abirazad@iut-dhaka.edu

Fardin Ahsan, 160041059
Computer Science and Engineering
Islamic University of Technology
Dhaka, Bangladesh
fardinahsan@iut-dhaka.edu

A. H. M. Rezaul Karim, 160041022
Computer Science and Engineering
Islamic University of Technology
Dhaka, Bangladesh
rezaulkarim@iut-dhaka.edu

Ahmad Imam, 160041054
Computer Science and Engineering
Islamic University of Technology
Dhaka, Bangladesh
ahmadimam@iut-dhaka.edu

I. Introduction

Multi-class classification problems are revised through and through for a long time in the domain of Machine Learning where each instance is labeled with a single label. Modern classification problems often involve the prediction of multiple labels simultaneously associated with a single instance. In our experimentation, we wanted to implement a robust model that can handle one such Multi-label classification problem. Genre Classification for movies and TV series is one of the tasks that is being performed by movie experts from the beginning of the industries. But if we can effectively build a model for generating the correct genres for a particular movie it can automate the reviewing process. For every movie, we can find their summary plot which is actually a series of texts containing information from the perspective of the machine. This directly falls into the multi-label text classification problem. We extracted a dataset of movies with their plot summary collected from IMDb and trained our model with the dataset. Then again there were some cases where only a plot summary could not predict efficiently, like whether a movie is an *'Animation'* or not, this usually is not conveyed in the plot summaries. For handling such cases we thought of using the title poster of the movies as the visual information of the posters might fetch such details about the movies. Thus in our project, we had to combine a NLP model and a vision based model together and create an architecture that can predict the movie genres with a distinctive accuracy. That's how, we tried to build a robust model using spatial and temporal information.

II. Background Study and Related Works

Genre Classification is not a new concept in machine learning. The work of Prateek Joshi primarily inspired us where he used a simple NLP model to handle such a problem and got around 40% accuracy. [1] This motivated us to use a deep neural network model for predicting the genres. We found the work of Druvil Shah who approached using Convolutional Neural Network (CNN) over the movie posters for genre prediction. In his work, he achieved around *53%* accuracy even after using a relatively smaller dataset. [2]

We implemented all our codes using Pytorch and FastAI library, [3] a layered API built on top of Pytorch. For our NLP part, we used the Universal Language Model Fine-tuning (ULMFiT) proposed by Jeremy Howard. [4] ULMFiT is an effective transfer learning method that can be applied to any task in NLP, and introduce techniques that are key for fine-tuning a language model. FastAI has an English model with an AWD-LSTM architecture over Wikitext-103. [5] We used this model to further fine-tune our own model. And for integrating the poster images with the architecture we used ResNet-50. [6] ResNet-50 is a convolutional neural network that is 50 layers deep. We loaded a pre-trained version of the network trained on more than a million images from the ImageNet database. [7] The pre-trained network can classify images into 1000 object categories, such as the keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images.

III. Methodology

We scraped the title poster, plot summary and genres from IMDb. After that, the image data and text data was preprocessed while the genres were used as labels. Finally, we used a pre-trained Resnet50 model for image data, a pre-trained LSTM model to encode and train the text model and merged them with some pooling and custom layers to get the final output.

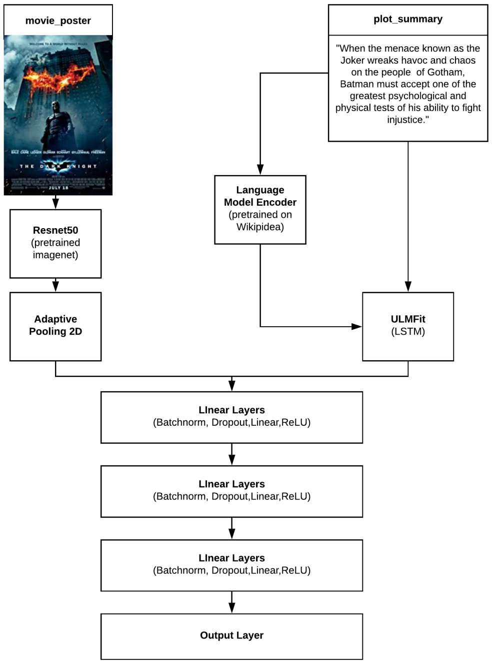
movie_poster
plot_summary
"When the menace known as the Joker wreaks havoc and chaos on the people of Gotham, Batman must accept one of the greatest psychological and physical tests of his ability to fight injustice."
Language Model Encoder (pretrained on Wikipidea)
Resnet50 (pretrained imagenet)
Adaptive Pooling 2D
ULMFit (LSTM)
LInear Layers (Batchnorm, Dropout,Linear,ReLU)
LInear Layers (Batchnorm, Dropout,Linear,ReLU)
LInear Layers (Batchnorm, Dropout,Linear,ReLU)
Output Layer

Fig. 1. Model Architeture

A. Data Collection

IMDb is a source of a reliable dataset for movies and TV series. All the submissions and suggestions about the title posters, plot summaries and genres go through a reviewing process and experts either deny or accept the submission. So, the dataset is labeled with domain experts. These are not subjected to copyright if not used for commercial reasons.

B. Preprocessing

Since we will be using Resnet50 pre-trained on Imagenet for image data, we normalized the image data according to their statistics. In case of the NLP part before applying any algorithm we need to process the simple texts. At first we changed the raw texts to lists of words, or tokens through tokenization. Then we transformed these tokens into numbers, a process known as numericalization. These numbers were then passed to embedding layers that will convert them in arrays of floats before passing them through a model.

C. Model Architecture

The image data is fed into a pre-trained Resnet50 model followed by an adaptive pooling 2D layer to adapt to different sizes of the image. The text data is fed into a language model encoder that was pre-trained on the Wikipedia corpus and generates an encoder. The text data is again fed into an LSTM model where the trained Language Model encoder is used for classification.

The LSTM model and Adaptive pooling Layer [8] are concatenated into a linear custom layer. The linear layer is followed by another two custom hidden layers and an output layer for evaluation. For more insight, we can look into Figure 1.

IV. Experimental Analysis

A. Dataset Description

The data was collected by diligent web scraping. There are 2,500 instances in the dataset and 21 genres as label in total. Each instance can have one or more genres as label. The data distribution according to genres is shown in **Figure 2**. Each instance of the dataset contains a title poster, a plot summary for feature extraction and multiple genres as labels.

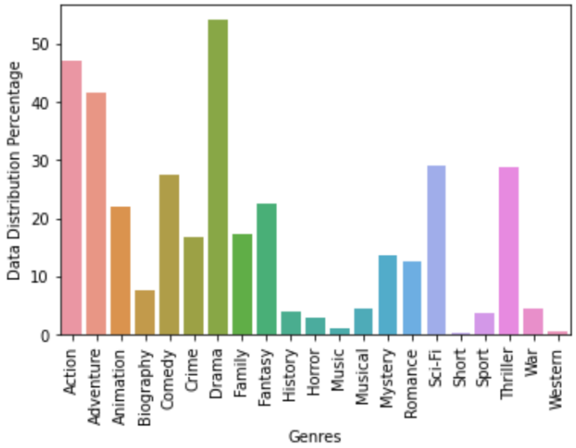

Fig. 2. Data Distribution of Genres

B. Training and Testing

After preprocessing and shuffling the dataset randomly, it was split into train and test set in 80-20 ration in a stratified way. At first, we trained and fine tuned the image data on pretrained Resnet50 and text data on pretrained LSTM separately. Then we merged the models, added some custom linear layers and trained on the whole model.

C. Result Analysis

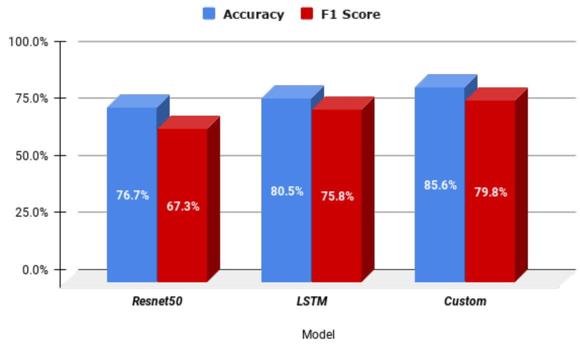

Fig. 3. Accuracy analysis

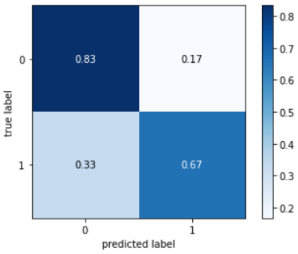

Fig. 4. Confusion Matrix for Action Genre

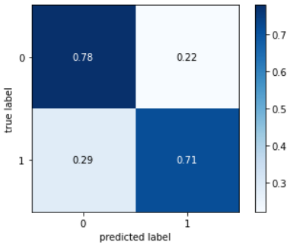

Fig. 5. Confusion Matrix for Adventure Genre

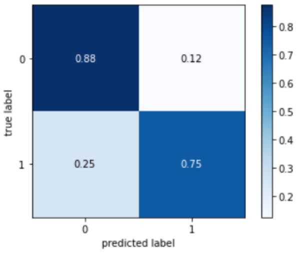

Fig. 6. Confusion Matrix for Animation Genre

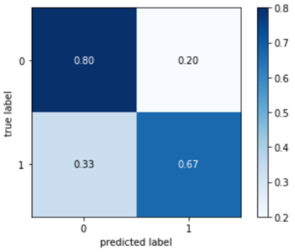

Fig. 7. Confusion Matrix for Drama Genre

We had to tune the learning rate to get the best result in case of 3 models. We used accuracy and F1-Score as evaluation metric. In all of the models we used Adam Optimizer to adapt the weights and One-vs-Rest method to classify multiple labels. Our neural network model finally gives 21 binary outputs for each genres in the output layer. In case of merged model, we added some linear custom models with batch normalization, dropout regularization and ReLU as activation function. In our output layer, we used sigmoid activation function to get the binary outputs.

From Figure 3, we can see that we got 76% accuracy in Resnet50 model when we trained with just the title posters, 80% accuracy in LSTM Model when we trained with just the plot summaries and 85% accuracy when we trained with both models and custom layers. The reason behind this difference can be - it is not possible to identify the genres just by looking at the plot or the poster. For example, we can't say if a movie or series is of 'Animation' genre just by reading the plot summary. Even the experts cannot do that properly. But looking at a title poster, we all can certainly say if it is of 'Animation' genre. What we tried is to merge both experience and learn from them simultaneously.

We plotted a confusion matrix for each of models to see which genre can be predicted more precisely. Figure 4,5,6 and 7 show us the confusion matrix for Action, Animation, Adventure and Drama Genre.

V. Conclusion and Future Work

Our experimentation was an approach towards building a deep neural network architecture which uses both temporal and spatial information. Our model showed significant performance in solving this multi-label genre classification problem. We were able to successfully combine pre-trained model with our existing architecture to get more accurate result. In future, we wish to work further to fine-tune the model and achieve state of the art accuracy.

References

[1] "https://www.analyticsvidhya.com/blog/2019/04/predicting-movie-genres-nlp-multi-label-classification/,"

[2] "https://towardsdatascience.com/cnn-approach-for-predicting-movie-genre-from-posters-95f122f88bc2,"

[3] J. Howard and S. Gugger, "Fastai: A layered api for deep learning," *Information*, vol. 11, p. 108, 02 2020.

[4] J. Howard and S. Ruder, "Fine-tuned language models for text classification," *CoRR*, vol. abs/1801.06146, 2018.

[5] . Stephen Merity et al., "Wikitext-103,"

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[8] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.