

## CS5242 Project: Film's Genre Classification with Deep Learning

### **Group 20**

Sashankh Chengavalli Kumar (A0162363J)

Tran Ba Phong (A0228533E)

Wong Wei Hao (A0262004X)

## Project Overview

Multi-class, single label classification problem, predicting genre of a film given the plot

Project captured within 3 notebooks:

**(1) Data\_Collection\_Cleaning\_And\_Analysis.ipynb**

- Outputs cleaned dataset stored in Zipped folder of Google Drive after scraping, to ensure convenience for consumption during model development phase

**(2) Baseline\_Model.ipynb**

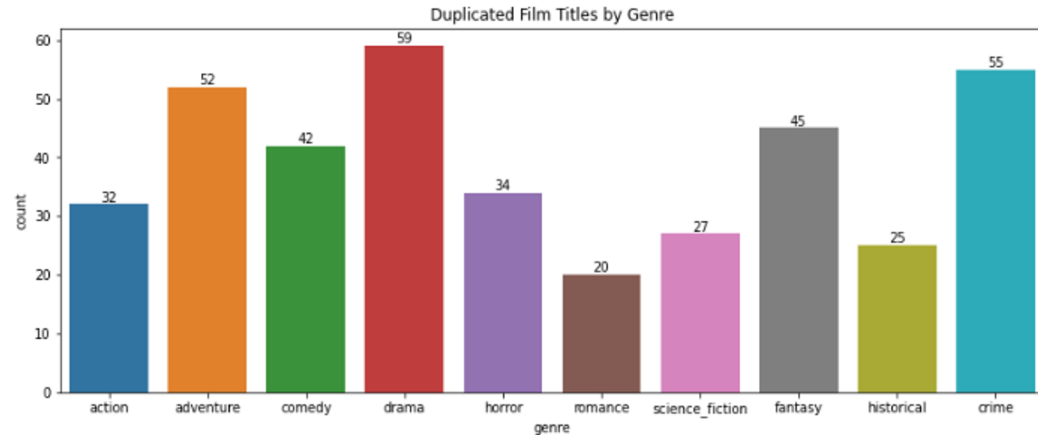
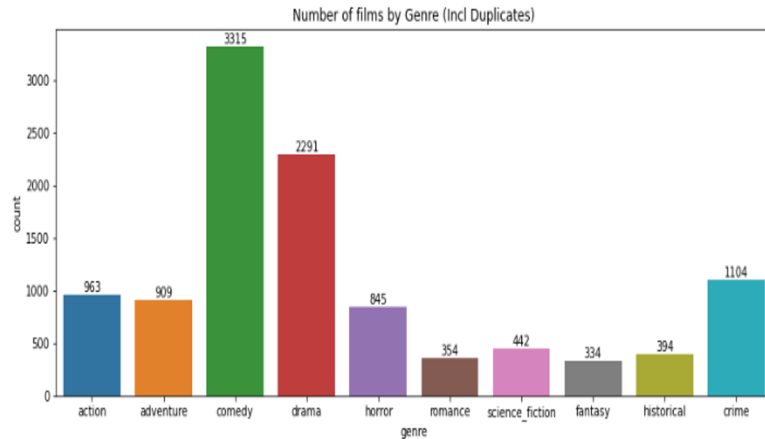
**(3) Deep\_Learning\_Model.ipynb**

- Using CNN, RNN, LSTM, Self-attention, and Transformer to classify the film's genres

## Data Collection, Exploration and Pre-processing

Web Scraping (film titles & genres) from Wikipedia using:

- requests and BeautifulSoup libraries
- American, British, Canadian, Australian, New Zealand, Hong Kong, South Korean and Chinese films

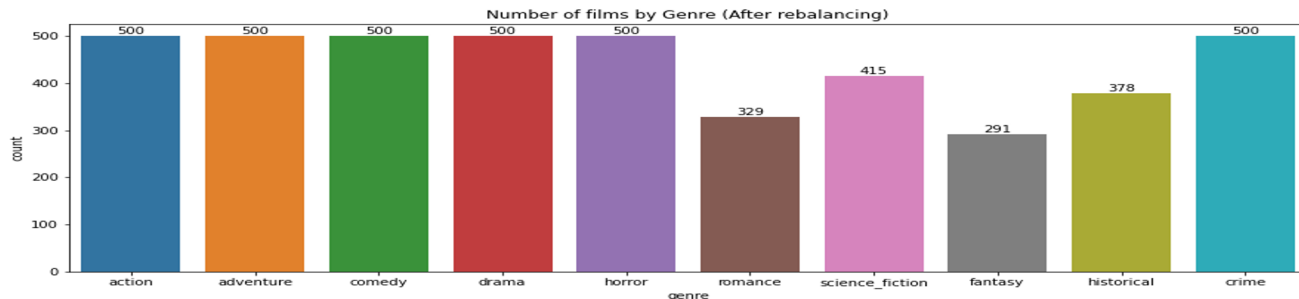


Total Number of films scraped: 10952 (Including duplicates)

- The number of films will increase when code is re-executed in the future

## Prior to Scraping for film plots :

- (1) Removed duplicated films by title
- (2) Rebalanced each genre to have **500** films (for those having more than 500 films), randomly chosen (reproducible) using `df.groupby('genre').apply(pd.DataFrame.sample,n=500,random_state=1)`



## Begin Scraping for Film Plots:

- (1) using Wikipedia and Wikipedia API libraries (try and except approach)
  - Note that there might be some cases whereby the functions are unable to fetch plots even if it exists within page, the number of films for genres with lesser than **500** films may differ slightly when the code is re-executed

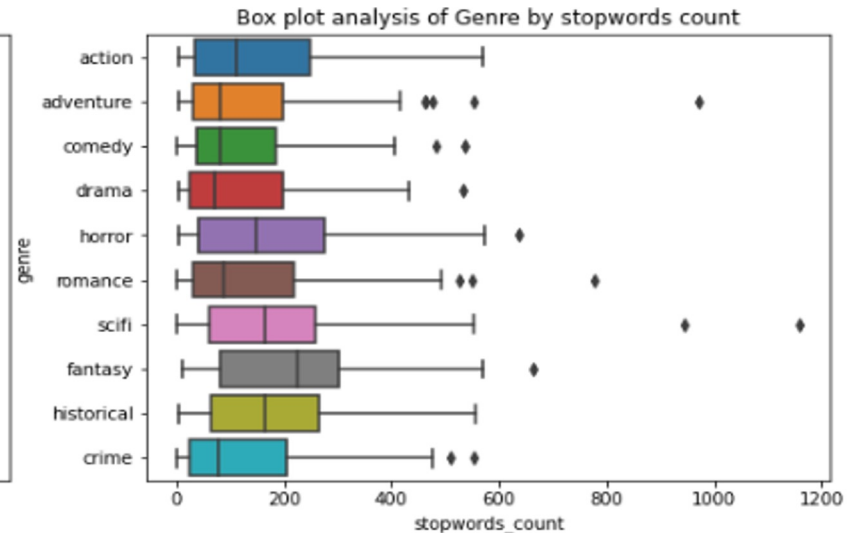
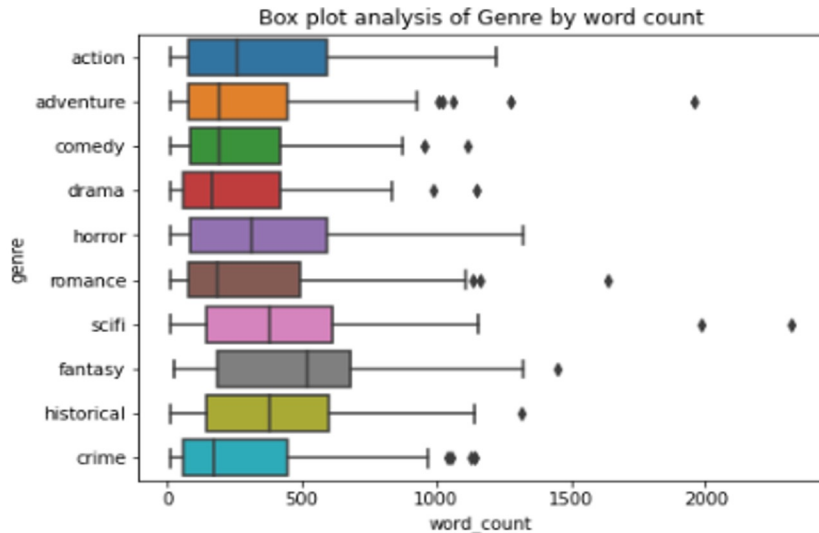
## After Scraping for Film Plots, removed films with:

- (1) null/na plots
- (2) "Page with no Section on Plot"
- (3) "Error" plots
- (4) Plots with word count < 10 to prevent situations in CNN1d where plot length is < kernel size

## Removed punctuations from plots

## Data Collection, Exploration and Pre-processing

Distribution of word count across different genres is highly similar to the distribution of stopwords (nltk) count (approximately 40% of plot length) - Dataset (1)



# Data Collection, Exploration and Pre-processing

**Final Dataset 1** (Zipped Folder):

`!gdown 10fAKL5gOYjG0WcNlwZnZ1X9_n8pciT19`

`path = "/content/data_full3.zip"`

`initial_df = pd.DataFrame()`

`with zipfile.ZipFile('/content/data_full3.zip') as z:`

```
    for name in z.namelist():
        if name.endswith(".csv"):
            print(f'Loading data from {name}...')
            x = pd.read_csv(z.open(name))
            print(f'Loading completed from {name}...')
            initial_df = pd.concat([initial_df, x[['genre', 'plot']]], axis=0, ignore_index=True)
    print("Dataframe (df) ready to be used!")
```

`initial_df`

**Final Dataset 2** (Zipped Folder):

`!gdown 1dhTF48AKpPBEq0ZeSstOSsS17A7ktmzl`

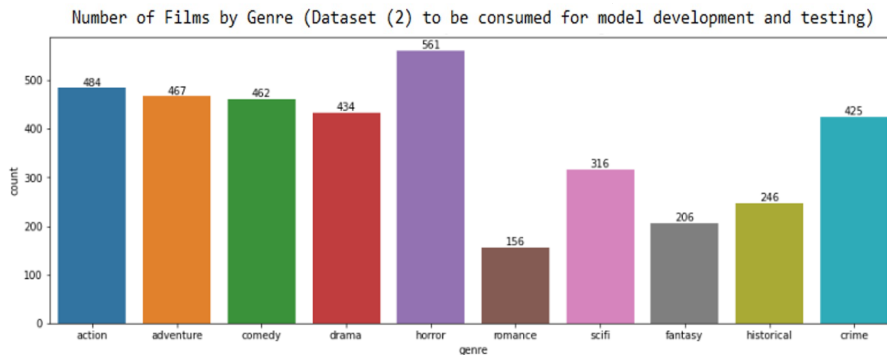
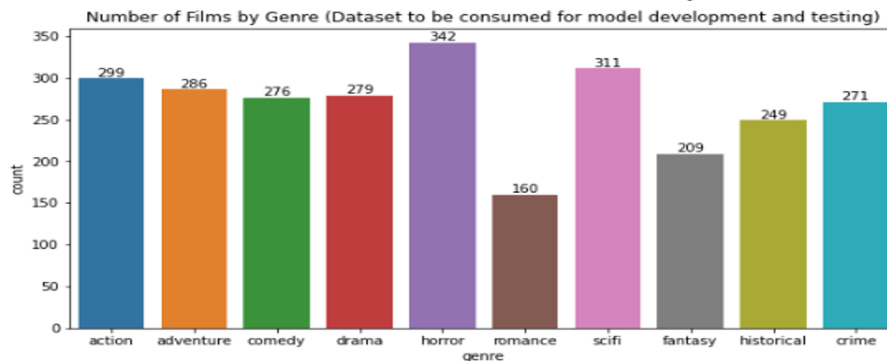
`path = "/content/data_full4.zip"`

`initial_df = pd.DataFrame()`

`with zipfile.ZipFile('/content/data_full4.zip') as z:`

```
    for name in z.namelist():
        if name.endswith(".csv"):
            print(f'Loading data from {name}...')
            x = pd.read_csv(z.open(name))
            print(f'Loading completed from {name}...')
            initial_df_2 = pd.concat([initial_df_2, x[['genre', 'plot']]], axis=0, ignore_index=True)
    print("Dataframe (df) ready to be used!")
```

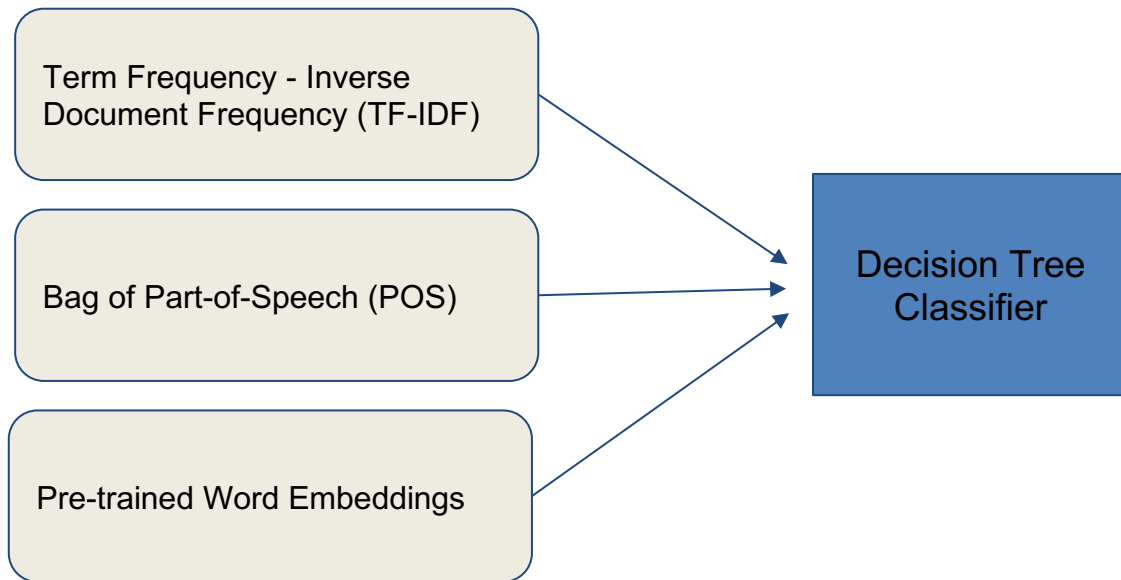
`initial_df_2`



## Traditional Machine Learning Baselines (Model Based)

Baseline as in: [https://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=3547887](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3547887)

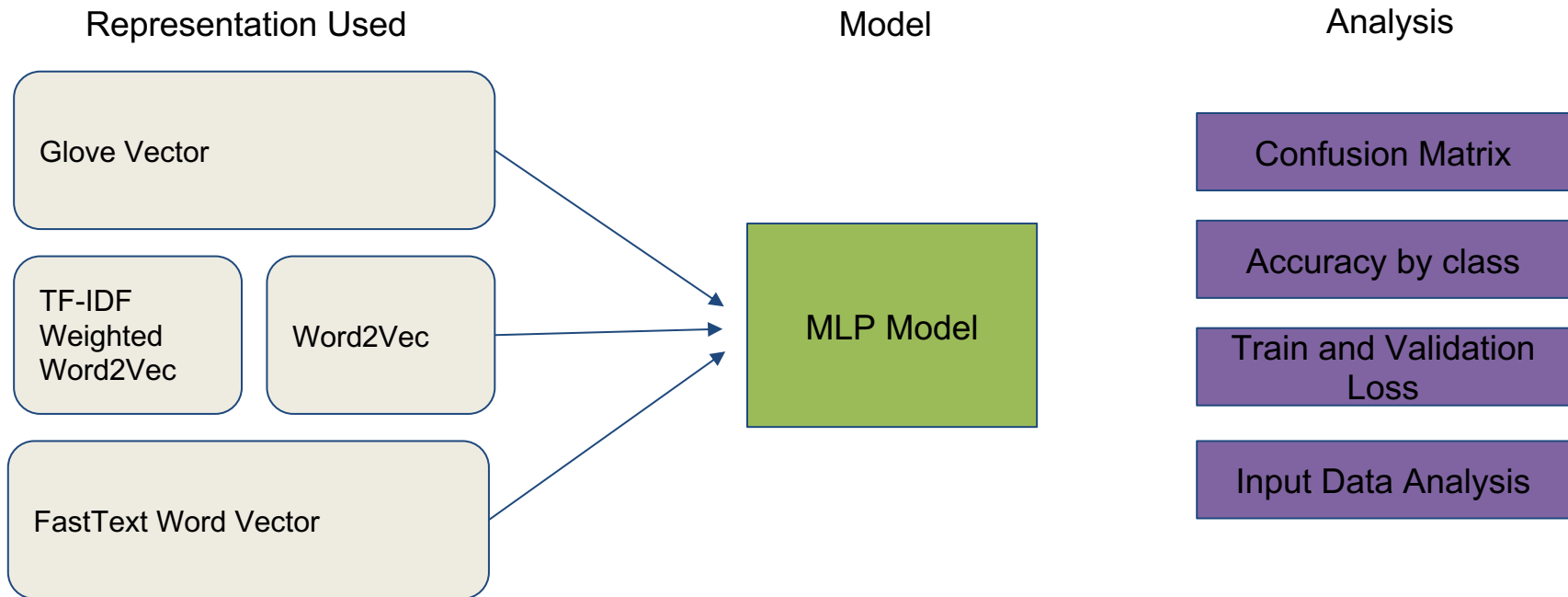
### Representation Used



### K-Fold Cross Validation Approach

TF-IDF	34.8%
POS	20.9%
Word Emb	33.9%

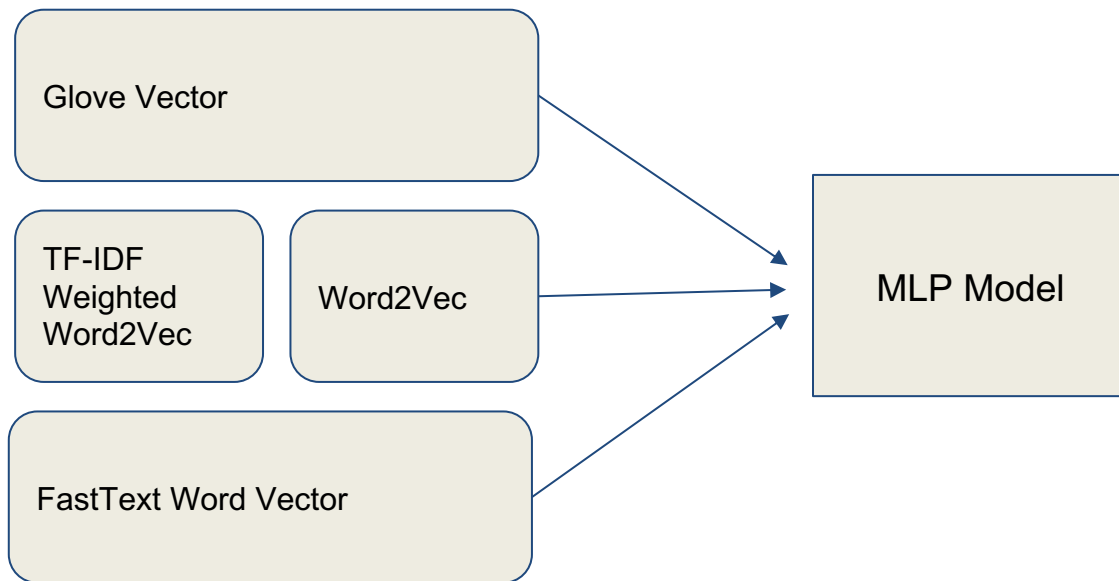
## Deep Learning 1: Multi-Layer Perceptron





## Deep Learning 1: Multi-Layer Perceptron

### Representation Used



10 class	42%
4 class	63.8%

# Deep Learning 1: Multi-Layer Perceptron

## Highlights

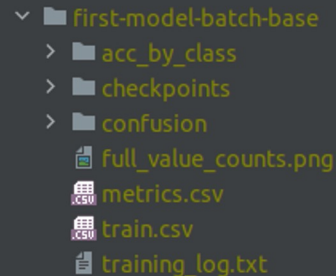
1. Custom framework to run experiments seamlessly
2. Implementation of functionality from scratch
  - a. TF-IDF
  - b. Model Parsing
3. Integrating checkpointing, embeddings, down-sampling of data, and more!
4. Single configuration to define the whole model
5. Generates output animation and plots

```
{  
  "data_path": "source_data/data_full3.zip",  
  "model_name": "first-model-batch-base",  
  "structure": [  
    "l_50_125", "d_0.3", "r", "l_125_125", "d_0.3", "r",  
    "l_125_150", "d_0.3", "r", "l_150_110", "d_0.3", "r",  
    "l_110_75", "l_75_10", "s_0"],  
  "lr": 3e-4,  
  "epochs": 250,  
  "input_dim": 50,  
  "batch_size": 16,  
  "simple": true,  
  "embedding_type": "glove-wiki-gigaword-50",  
  "down_sample": {  
    "drama": 0.0,  
    "comedy": 0.0  
  }  
},
```

# Deep Learning 1: Multi-Layer Perceptron

## Highlights

1. Custom framework to run experiments seamlessly
2. Implementation of functionality from scratch
  - a. TF-IDF
  - b. Model Parsing
3. Integrating checkpointing, embeddings, down-sampling of data, and more!
4. Single configuration to define the whole model
5. Generates output animation and plots



```
▼ first-model-batch-base
  > acc_by_class
  > checkpoints
  > confusion
  full_value_counts.png
  metrics.csv
  train.csv
  training_log.txt
```

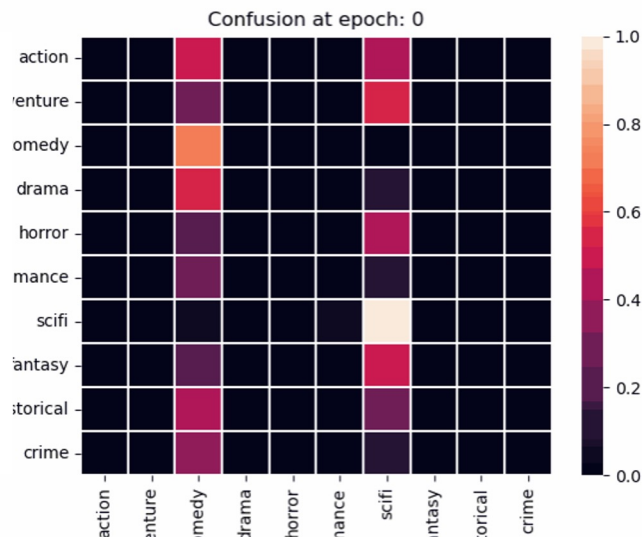
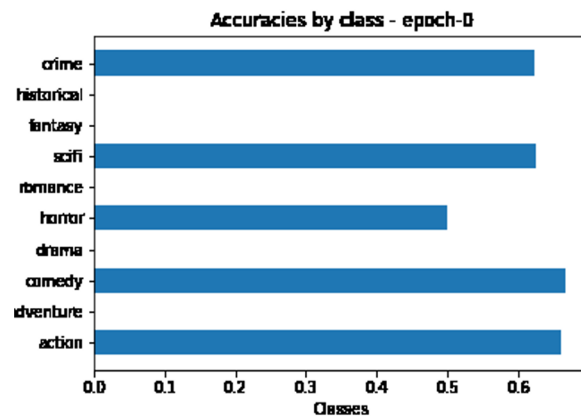
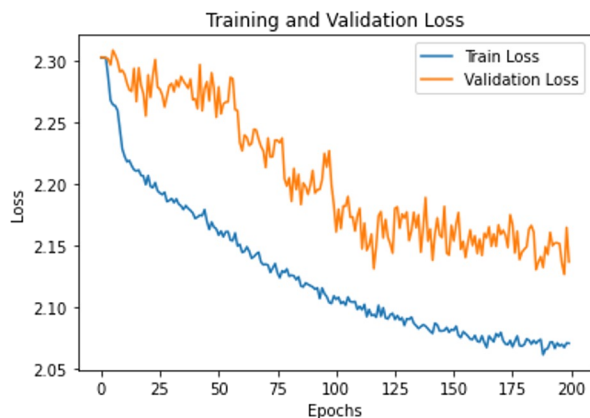
## Deep Learning 1: Multi-Layer Perceptron Results

Model	Best Accuracy	Comments
4 Class MLP	63.8%	Glove 300d
10 Class MLP	42%	Glove 100d

### Observations:

1. Beats uniform random guess reasonably (10% for 10 class clf)
2. Mean vector not a good enough representation of plot
3. Topic differences sometimes do not correlate with genre differences!
4. Dataset size and skewness potential causes of poor performance
  - a. 10 classes too many and too similar!
5. Drop out has a great effect!

# Deep Learning 1: Multi-Layer Perceptron Results



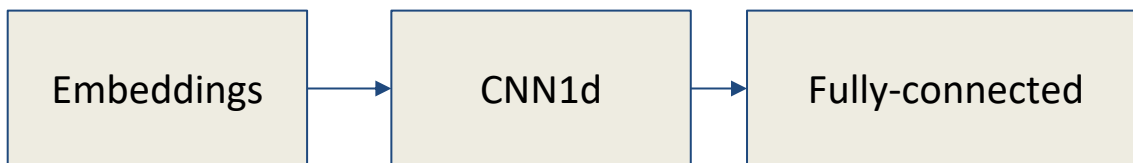
# Deep Learning (CNN1d, RNN, LSTM, Attention and Transformer)

## Data preparation:

1. Using Final Dataset 1 zipped file, containing 10 genres and 2413 plots
1. Tokenize of all the plots
1. **(Optional)** Remove escapes, remove punctuation, lemmatization and stemming are further explored to check performance
1. **(Optional)** Tripping length of plots and using batched are further explored

# Deep Learning (CNN1d)

## Architecture



# Deep Learning (CNN1d)

## Default Parameters

Settings:

- Batch = 1
- No padding of each plot
- No pretrained embeddings
- Adadelata optimizer
- Three CNN1d layers
- Learning rate = 0.01
- No clipping of plot length
- No clipping number of genres



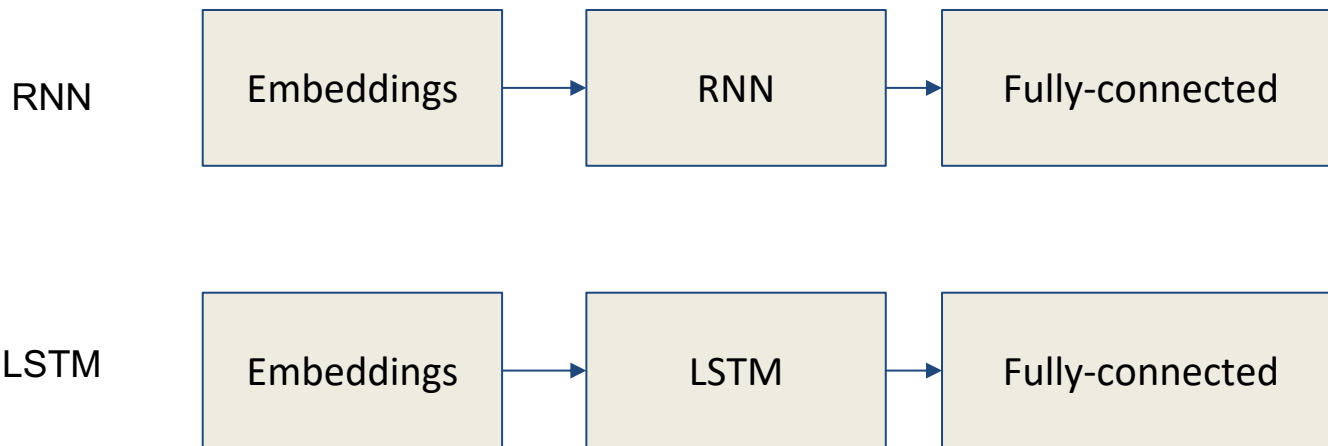
# Deep Learning (CNN1d)

## Changing Parameters (showing highlighted models)

Model Name	Modification from default parameters	Validation Accuracy
CNN1	Default	43.12%
CNN2	Glove 6B.100d pretrained	57.99%
CNN3	Glove 6B.100d pretrained, Higher learning rate	58.36%
CNN4	Glove 6B.100d pretrained, Adam optimization	43.87%
CNN6	Glove 6B.100d pretrained, Dropout, High Learning rate	58.36%
CNN9	Glove 6B.100d pretrained, Dropout, High Learning rate, Clipping dataset size	57.14%
CNN22	Glove 6B.100d pretrained, Dropout, High Learning rate, lemma and stem data	50.19%

# Deep Learning (RNN + LSTM)

## Architecture



# Deep Learning (RNN + LSTM)

## Performance compared to CNN1d

Model Name	Modification from default parameters	Validation Accuracy
CNN1d	Glove 6B.100d pretrained	<b>57.99%</b>
RNN	Glove 6B.100d pretrained	<b>34.57%</b>
LSTM	Glove 6B.100d pretrained	<b>45.35%</b>

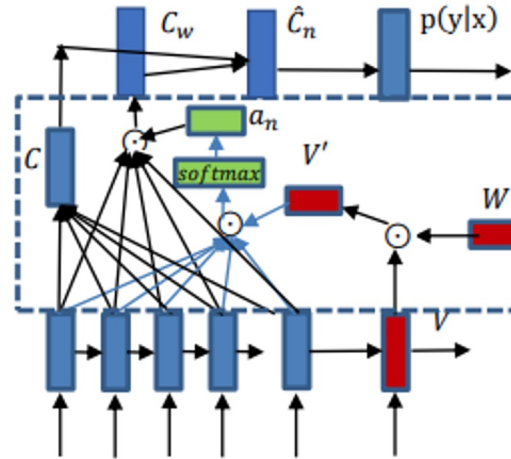
Some observations:

1. Pretrained-model significantly improves the model performance
2. RNN and LSTM seems perform not well compared to CNN1d. It could be that the network is not too big to able to capture the context (even though the number of parameters is roughly same as CNN1d). Or the film's plot is meaningful if we only capture in a neighbor of text rather than capturing all the words of plot (as in RNN and LSTM).

# Deep Learning (RNN-Attention + LSTM-Attention)

Network adapted from paper “A self-attention Based LSTM Network for Text Classification”

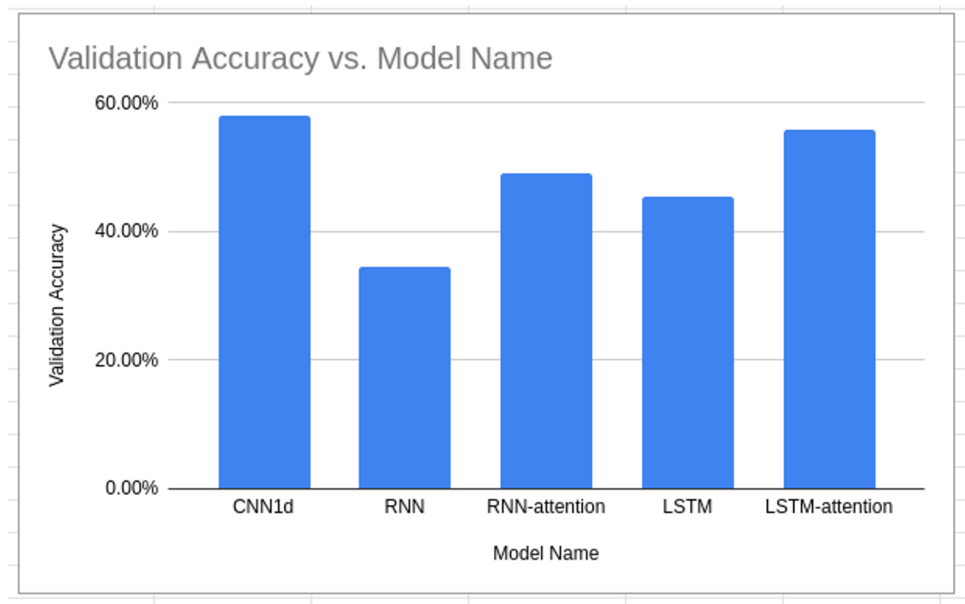
<https://iopscience.iop.org/article/10.1088/1742-6596/1207/1/012008/pdf>



**Figure 1.** Algorithmic framework

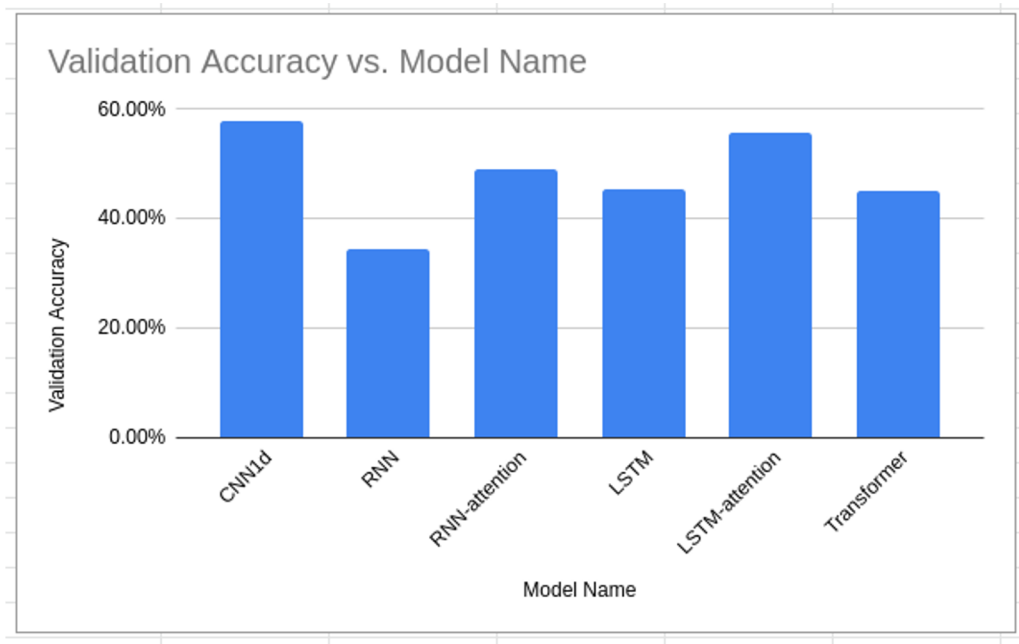
# Deep Learning (RNN-Attention + LSTM-Attention)

## Performance compared to previous networks (with Glove embeddings)

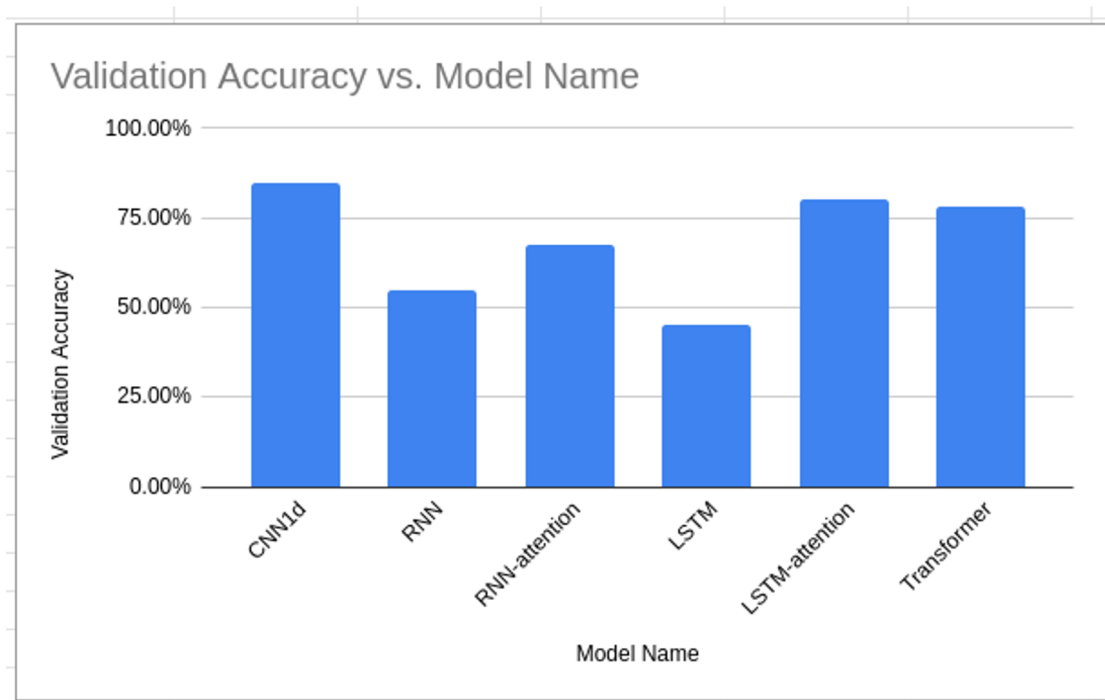


# Deep Learning (Transformer)

Network adapted from paper link (<https://blog.paperspace.com/transformers-text-classification/>). Here we show performance



# Deep Learning (New 4d dataset)



# Summary

In this project we did film genre classification. Some comments about the project:

1. Data collection:
  - a. Collect film's genres from different countries and use 10 genres for classification
2. MLP models
  - a. Mean vector not a sufficient representation of the plot (word-word relations lost)
  - b. Unable to differentiate genres if the topics discussed are not distinct.
3. Deep Learning models:
  - a. For text-classification, a model captures the neighbor relationship (CNN, Self-attention and Transformer) generally performs better than a model of temporal relationship (e.g., RNN and LSTM)