Movie Genre Prediction by Its Synopsis

Artem Razdyakonov

January 2023

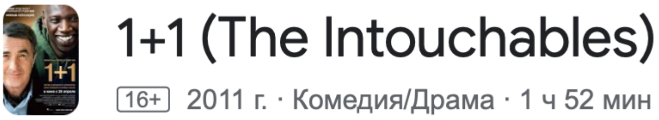
Abstract

This document is the report for NLP course project and will provide you insights about different ways to classify movies as either comedies or dramas based on their synopsis. Here is a link to this project code: https://github.com/rtm-1dyakonov/nlp_course.

1 Introduction

The task of determining the genre of a film based on its description has long ceased to be a challenge for machine and deep learning algorithms. Even the straightforward BoW can reach high score. It is not surprising since key words, for instance, of horror genre are quite different from key words of family film.

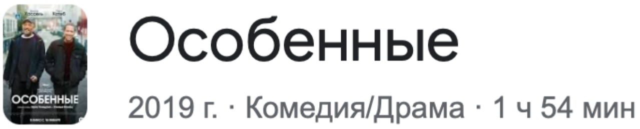
The moment when the task is to identify similarly close genres, complexity appears. Specifically, if you try to google a comedy film, almost every film will contain both comedy and drama genre in its description. Examples of this are presented on Fig. 1 – 4. Therefore, this research aims to check how well do deep learning algorithms cope with the task of distinguish between comedies and dramas.



1+1 (The Intouchables)

16+ 2011 г. · Комедия/Драма · 1 ч 52 мин

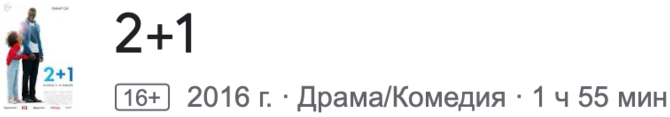
Figure 1: The Intouchables genres.



Особенные

2019 г. · Комедия/Драма · 1 ч 54 мин

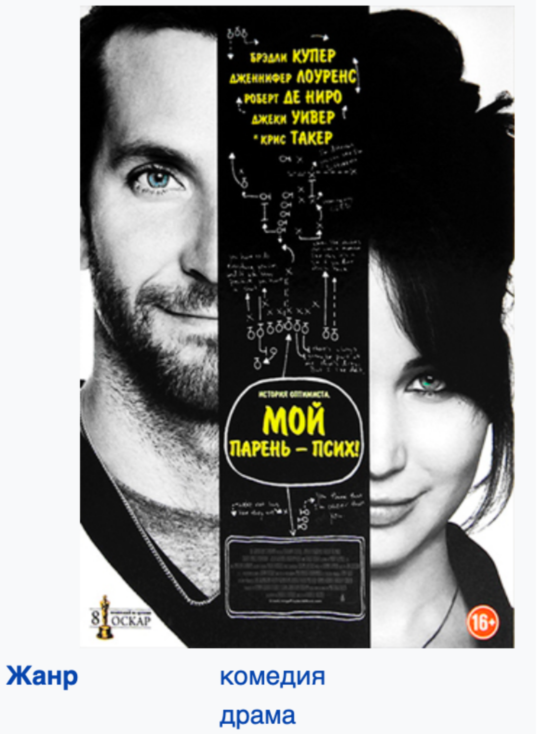
Figure 2: The Specials genres.



2+1

16+ 2016 г. · Драма/Комедия · 1 ч 55 мин

Figure 3: Two Is a Family genres.



Жанр	комедия драма

Figure 4: Silver Linings Playbook genres.

1.1 Team

Artem Razdyakonov has performed research and prepared this document.

2 Related Work

The thing is there are very few works of such specifics. Most of the time the main goal of movie genre prediction projects is to train model, which will be able to distinguish several different genres, instead of a couple of close genres.

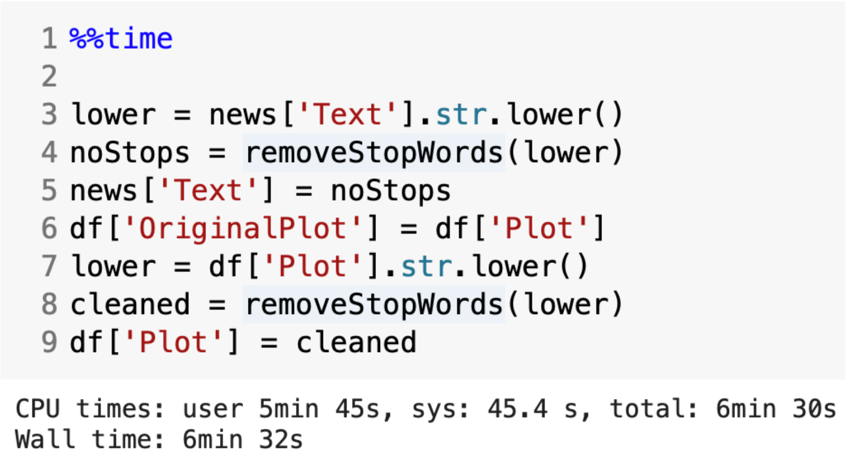
Quan Hoang [1] studied the problem of predicting genres from movie plot and used different methods like Recurrent Neural Networks and Word2Vec+XGBoost are used text classification, also K-binary transformation and probabilistic classification were employed to tackle the multi label problem. He attained a high F1-score of 0.56 along with a hit rate of 80%.

A. Blackstock and M. Spitz [2] attempt to classify movie scripts by building a logistic regression model using NLP-related features extracted from the scripts such as the ratio of descriptive words to nominals or the ratio of dialogues frames to non-dialogue frames. The experiment is done on a small dataset with only 399 scripts and the best subset of features achieves an F1-score of 0.56.

Ka wing Ho [3] investigates different methods to classify movies’ genres based on synopsis. The methods examined include One-Vs-All Support Vector Machines (SVM), Multi-label K-nearest neighbor (KNN), Parametric mixture model (PMM) and Neural network. The experiment is limited to only predicting only 10 most popular genres, including action, adventure, comedy, crime, documentary, drama, family, romance, short films, and thrillers. Overall, SVM achieves the highest F1-score of 0.55.

3 Model Description

Before any experiment with neural network models all movies’ plots were gone through preprocessing functions to make all words lowercase and get rid of stopwords. The code is represented on Fig. 5.

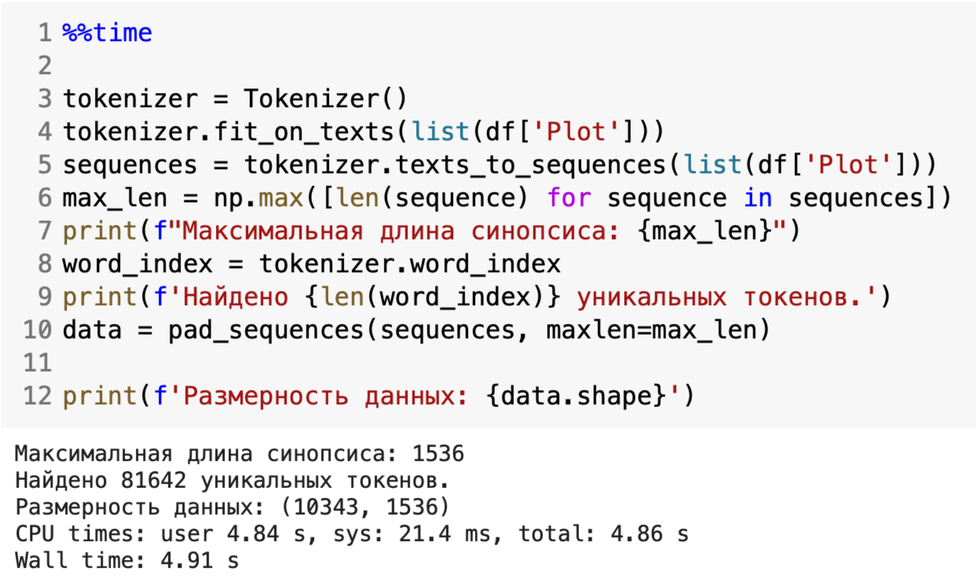


```
%%time

lower = news['Text'].str.lower()
noStops = removeStopWords(lower)
news['Text'] = noStops
df['OriginalPlot'] = df['Plot']
lower = df['Plot'].str.lower()
cleaned = removeStopWords(lower)
df['Plot'] = cleaned
```

```
CPU times: user 5min 45s, sys: 45.4 s, total: 6min 30s
Wall time: 6min 32s
```

Figure 5: Preprocessing of plots.

After the preprocessing stage all plots have been tokenized with Tokenizer from keras.preprocessing.text with padding each sequence up to the maximum length of plot vector, which is shown on Fig. 6.



```
%%time

tokenizer = Tokenizer()
tokenizer.fit_on_texts(list(df['Plot']))
sequences = tokenizer.texts_to_sequences(list(df['Plot']))
max_len = np.max([len(sequence) for sequence in sequences])
print(f"Максимальная длина синопсиса: {max_len}")
word_index = tokenizer.word_index
print(f'Найдено {len(word_index)} уникальных токенов.')
data = pad_sequences(sequences, maxlen=max_len)

print(f'Размерность данных: {data.shape}')
```

```
Максимальная длина синопсиса: 1536
Найдено 81642 уникальных токенов.
Размерность данных: (10343, 1536)
CPU times: user 4.84 s, sys: 21.4 ms, total: 4.86 s
Wall time: 4.91 s
```

Figure 6: Using Tokenizer to create sequences.

A CNN-based approach, a hybrid CNN-LSTM approach, and an ensemble of the two previous ones are applied to the problem.

4 Dataset

For training and testing purposes two datasets were used, which are Wikipedia Movie Plots[1] and News Category Dataset[2]. The last one was used as supplemental training data taken from the news category dataset, which contains headlines and short descriptions of Huffington Post articles, in addition to categories that include comedy. News descriptions are quite short, so they were concatenated with headlines to add information. Examples of each dataset are presented on Fig. 7 and Fig. 8. You can see the distribution of classes in Wikipedia Movie Plots on the Tab. 1, and the distribution of classes in News Category Dataset on the Tab. 2

[1] The Wikipedia Movie Plots link: https://www.kaggle.com/datasets/jrobischon/wikipedia-movie-plots

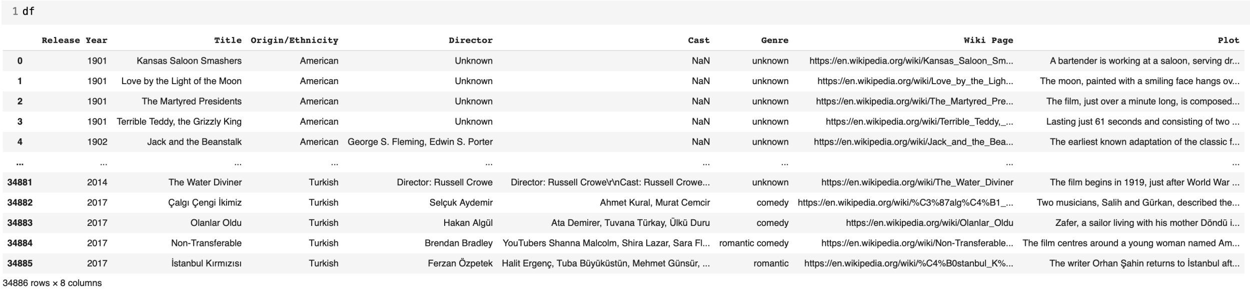
[2] The News Category Dataset link: https://huggingface.co/datasets/khalidalt/HuffPost/blob/main/News_Category_Dataset_v2.json

Genre	Plots count
drama	5964
comedy	4379

Table 1: Distribution of classes in Wikipedia Movie Plots.

Genre	News count
comedy	5175
other	5000

Table 2: Distribution of classes in News Category Dataset.

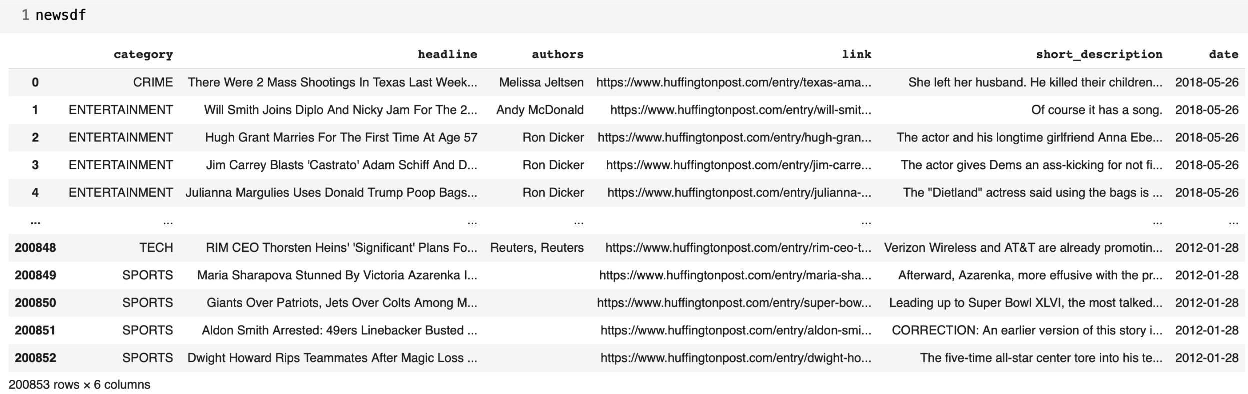


```
df
```

	Release Year	Title	Origin/Ethnicity	Director	Cast	Genre	Wiki Page	Plot
0	1901	Kansas Saloon Smashers	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/Kansas_Saloon_Sm...	A bartender is working at a saloon, serving dr...
1	1901	Love by the Light of the Moon	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/Love_by_the_Ligh...	The moon, painted with a smiling face hangs ov...
2	1901	The Martyred Presidents	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/The_Martyred_Pre...	The film, just over a minute long, is composed...
3	1901	Terrible Teddy, the Grizzly King	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/Terrible_Teddy,_...	Lasting just 61 seconds and consisting of two ...
4	1902	Jack and the Beanstalk	American	George S. Fleming, Edwin S. Porter	NaN	unknown	https://en.wikipedia.org/wiki/Jack_and_the_Bea...	The earliest known adaptation of the classic f...
...	...	...	...	...	...	...	...	...
34881	2014	The Water Diviner	Turkish	Director: Russell Crowe	Director: Russell Crowe\r\nCast: Russell Crowe...	unknown	https://en.wikipedia.org/wiki/The_Water_Diviner	The film begins in 1919, just after World War ...
34882	2017	Çalgı Çengi İkimiz	Turkish	Selçuk Aydemir	Ahmet Kural, Murat Cemcir	comedy	https://en.wikipedia.org/wiki/%C3%87alg%C4%B1_...	Two musicians, Salih and Gürkan, described the...
34883	2017	Olanlar Oldu	Turkish	Hakan Algül	Ata Demirer, Tuvana Türkay, Ülkü Duru	comedy	https://en.wikipedia.org/wiki/Olanlar_Oldu	Zafer, a sailor living with his mother Döndü i...
34884	2017	Non-Transferable	Turkish	Brendan Bradley	YouTubers Shanna Malcolm, Shira Lazar, Sara Fl...	romantic comedy	https://en.wikipedia.org/wiki/Non-Transferable...	The film centres around a young woman named Am...
34885	2017	İstanbul Kırmızısı	Turkish	Ferzan Özpetek	Halit Ergenç, Tuba Büyüküstün, Mehmet Günsür, ...	romantic	https://en.wikipedia.org/wiki/%C4%B0stanbul_K%...	The writer Orhan Şahin returns to İstanbul aft...

34886 rows × 8 columns

Figure 7: Wikipedia Movie Plots dataframe.



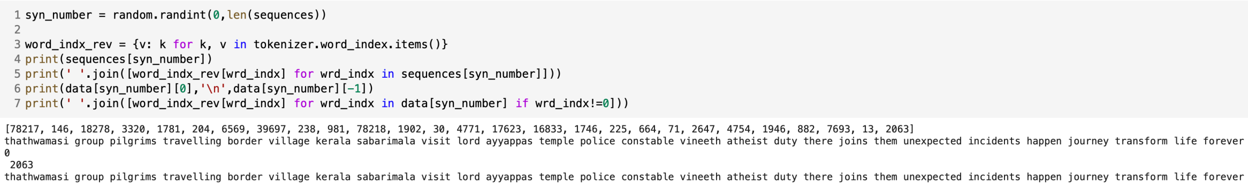
```
newsdf
```

	category	headline	authors	link	short_description	date
0	CRIME	There Were 2 Mass Shootings In Texas Last Week...	Melissa Jeltsen	https://www.huffingtonpost.com/entry/texas-ama...	She left her husband. He killed their children...	2018-05-26
1	ENTERTAINMENT	Will Smith Joins Diplo And Nicky Jam For The 2...	Andy McDonald	https://www.huffingtonpost.com/entry/will-smit...	Of course it has a song.	2018-05-26
2	ENTERTAINMENT	Hugh Grant Marries For The First Time At Age 57	Ron Dicker	https://www.huffingtonpost.com/entry/hugh-gran...	The actor and his longtime girlfriend Anna Ebe...	2018-05-26
3	ENTERTAINMENT	Jim Carrey Blasts 'Castrato' Adam Schiff And D...	Ron Dicker	https://www.huffingtonpost.com/entry/jim-carre...	The actor gives Dems an ass-kicking for not fi...	2018-05-26
4	ENTERTAINMENT	Julianna Margulies Uses Donald Trump Poop Bags...	Ron Dicker	https://www.huffingtonpost.com/entry/julianna-...	The "Dietland" actress said using the bags is ...	2018-05-26
...	...	...	...	...	...	...
200848	TECH	RIM CEO Thorsten Heins' 'Significant' Plans Fo...	Reuters, Reuters	https://www.huffingtonpost.com/entry/rim-ceo-t...	Verizon Wireless and AT&T are already promotin...	2012-01-28
200849	SPORTS	Maria Sharapova Stunned By Victoria Azarenka I...		https://www.huffingtonpost.com/entry/maria-sha...	Afterward, Azarenka, more effusive with the pr...	2012-01-28
200850	SPORTS	Giants Over Patriots, Jets Over Colts Among M...		https://www.huffingtonpost.com/entry/super-bow...	Leading up to Super Bowl XLVI, the most talked...	2012-01-28
200851	SPORTS	Aldon Smith Arrested: 49ers Linebacker Busted ...		https://www.huffingtonpost.com/entry/aldon-smi...	CORRECTION: An earlier version of this story i...	2012-01-28
200852	SPORTS	Dwight Howard Rips Teammates After Magic Loss ...		https://www.huffingtonpost.com/entry/dwight-ho...	The five-time all-star center tore into his te...	2012-01-28

200853 rows × 6 columns

Figure 8: News Category Dataset dataframe.

On Fig. 9 you can see the result of using Tokenizer's dictionary to transform text to vector and vice.



```
syn_number = random.randint(0,len(sequences))

word_indx_rev = {v: k for k, v in tokenizer.word_index.items()}
print(sequences[syn_number])
print(' '.join([word_indx_rev[wrd_indx] for wrd_indx in sequences[syn_number]]))
print(data[syn_number][0],'\n',data[syn_number][-1])
print(' '.join([word_indx_rev[wrd_indx] for wrd_indx in data[syn_number] if wrd_indx!=0]))
```

```
[78217, 146, 18278, 3320, 1781, 204, 6569, 39697, 238, 981, 78218, 1902, 30, 4771, 17623, 16833, 1746, 225, 664, 71, 2647, 4754, 1946, 882, 7693, 13, 2063]
thathwamasi group pilgrims travelling border village kerala sabarimala visit lord ayyappas temple police constable vineeth atheist duty there joins them unexpected incidents happen journey transform life forever
0
 2063
thathwamasi group pilgrims travelling border village kerala sabarimala visit lord ayyappas temple police constable vineeth atheist duty there joins them unexpected incidents happen journey transform life forever
```

Figure 9: Demonstration of Tokenizer.

The data is divided into training and validating sets as follows – 80% for training and 20% for validation.

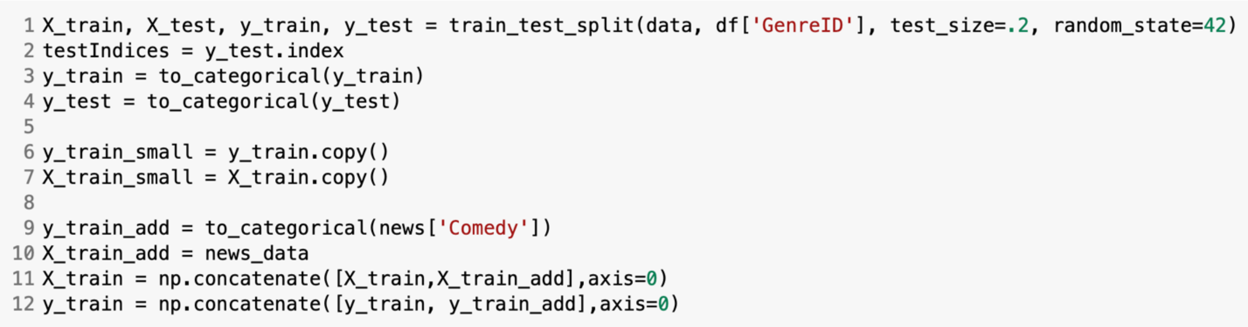
5 Experiments

5.1 Metrics

To evaluate the quality of all algorithms, a standard Accuracy metric will be used since our dataset is fairly balanced.

5.2 Experiment Setup

Firstly, we split the data into training and validating sets. Additionally, separate training sets for training data enhanced with news data and training data consisting only of movie data were created. It is presented on Fig. 10. Moreover, the GloVE 6B 100d[3] word embeddings were loaded in hopes that they will enhance the accuracy and training speed of the neural network models. After that 2 Keras embedding layers were created to convert the texts to embeddings. One uses Glove, and the other does not.



```
X_train, X_test, y_train, y_test = train_test_split(data, df['GenreID'], test_size=.2, random_state=42)
testIndices = y_test.index
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)

y_train_small = y_train.copy()
X_train_small = X_train.copy()

y_train_add = to_categorical(news['Comedy'])
X_train_add = news_data
X_train = np.concatenate([X_train,X_train_add],axis=0)
y_train = np.concatenate([y_train, y_train_add],axis=0)
```

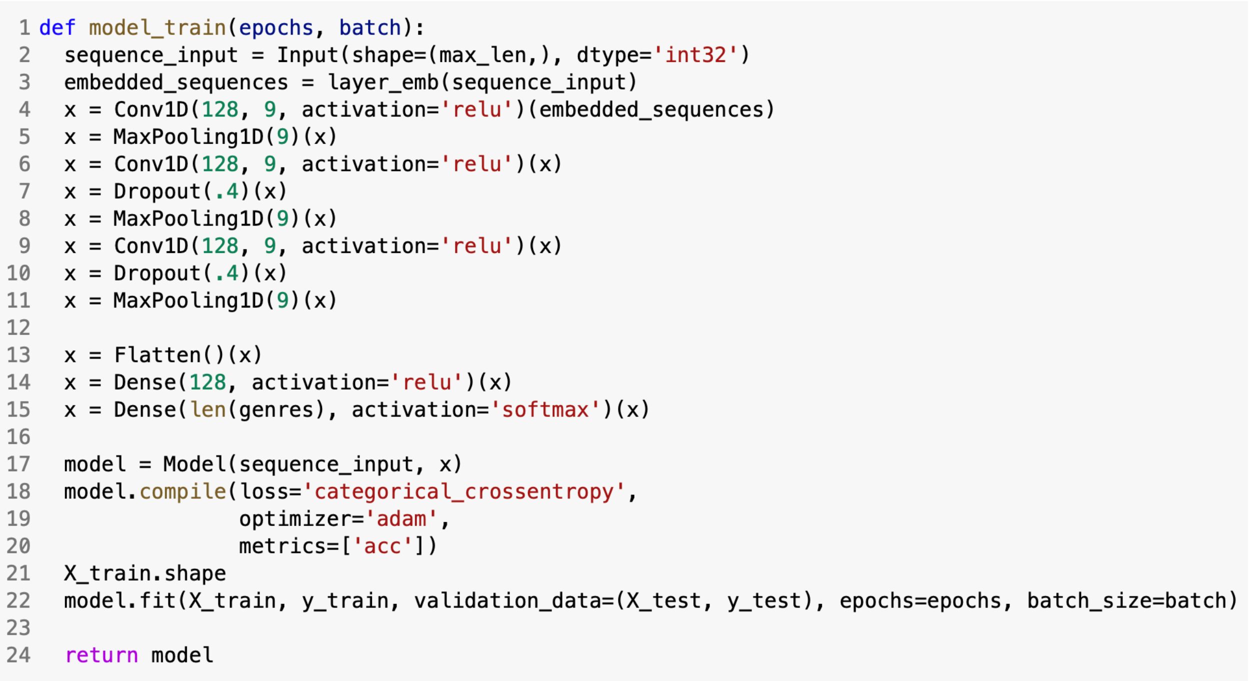
Figure 10: Splitting dataset with and without extra news data.

Secondly, 3 different CNN models with 3 layers were trained for 10 epochs and 128 batch size.

1. Without GloVE encodings, news and films data.
2. With GloVE encodings, news and movies data.
3. With GloVE encodings, only movies data.

[3] The GloVE 6B 100d link: https://www.kaggle.com/datasets/rtatman/glove-global-vectors-for-word-representation

The example of a CNN model is presented on Fig. 11.



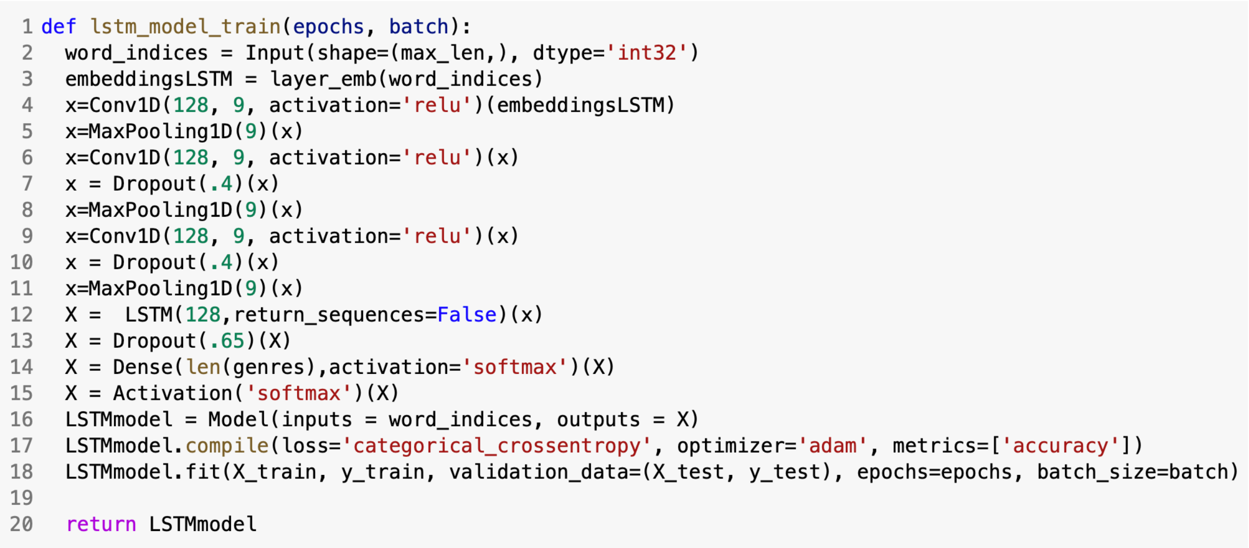
```python
def model_train(epochs, batch):
  sequence_input = Input(shape=(max_len,), dtype='int32')
  embedded_sequences = layer_emb(sequence_input)
  x = Conv1D(128, 9, activation='relu')(embedded_sequences)
  x = MaxPooling1D(9)(x)
  x = Conv1D(128, 9, activation='relu')(x)
  x = Dropout(.4)(x)
  x = MaxPooling1D(9)(x)
  x = Conv1D(128, 9, activation='relu')(x)
  x = Dropout(.4)(x)
  x = MaxPooling1D(9)(x)

  x = Flatten()(x)
  x = Dense(128, activation='relu')(x)
  x = Dense(len(genres), activation='softmax')(x)

  model = Model(sequence_input, x)
  model.compile(loss='categorical_crossentropy',
                optimizer='adam',
                metrics=['acc'])
  X_train.shape
  model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=epochs, batch_size=batch)

  return model
```

Figure 11: CNN model.

Thirdly, 2 different LSTM-CNN models were trained for 10 epochs and 128 batch size:

1. With GloVE encodings, news and movies data.
2. With GloVE encodings, only movies data.

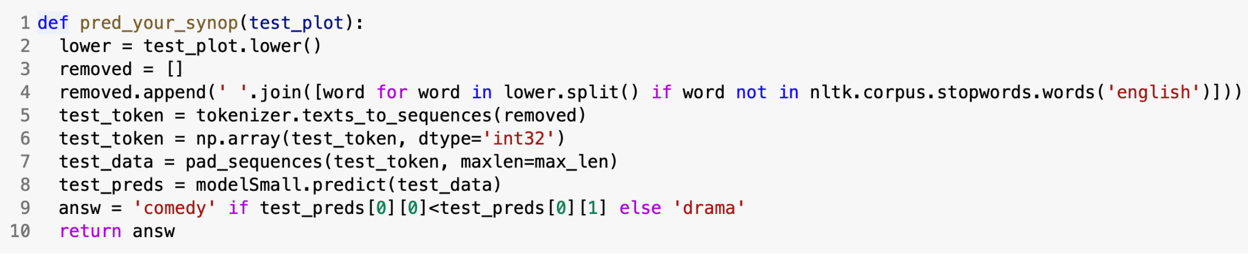
The example of a LSTM-CNN model is presented on Fig. 12.



```python
def lstm_model_train(epochs, batch):
  word_indices = Input(shape=(max_len,), dtype='int32')
  embeddingsLSTM = layer_emb(word_indices)
  x=Conv1D(128, 9, activation='relu')(embeddingsLSTM)
  x=MaxPooling1D(9)(x)
  x=Conv1D(128, 9, activation='relu')(x)
  x = Dropout(.4)(x)
  x=MaxPooling1D(9)(x)
  x=Conv1D(128, 9, activation='relu')(x)
  x = Dropout(.4)(x)
  x=MaxPooling1D(9)(x)
  X =  LSTM(128,return_sequences=False)(x)
  X = Dropout(.65)(X)
  X = Dense(len(genres),activation='softmax')(X)
  X = Activation('softmax')(X)
  LSTMmodel = Model(inputs = word_indices, outputs = X)
  LSTMmodel.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
  LSTMmodel.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=epochs, batch_size=batch)

  return LSTMmodel
```

Figure 12: LSTM-CNN model.

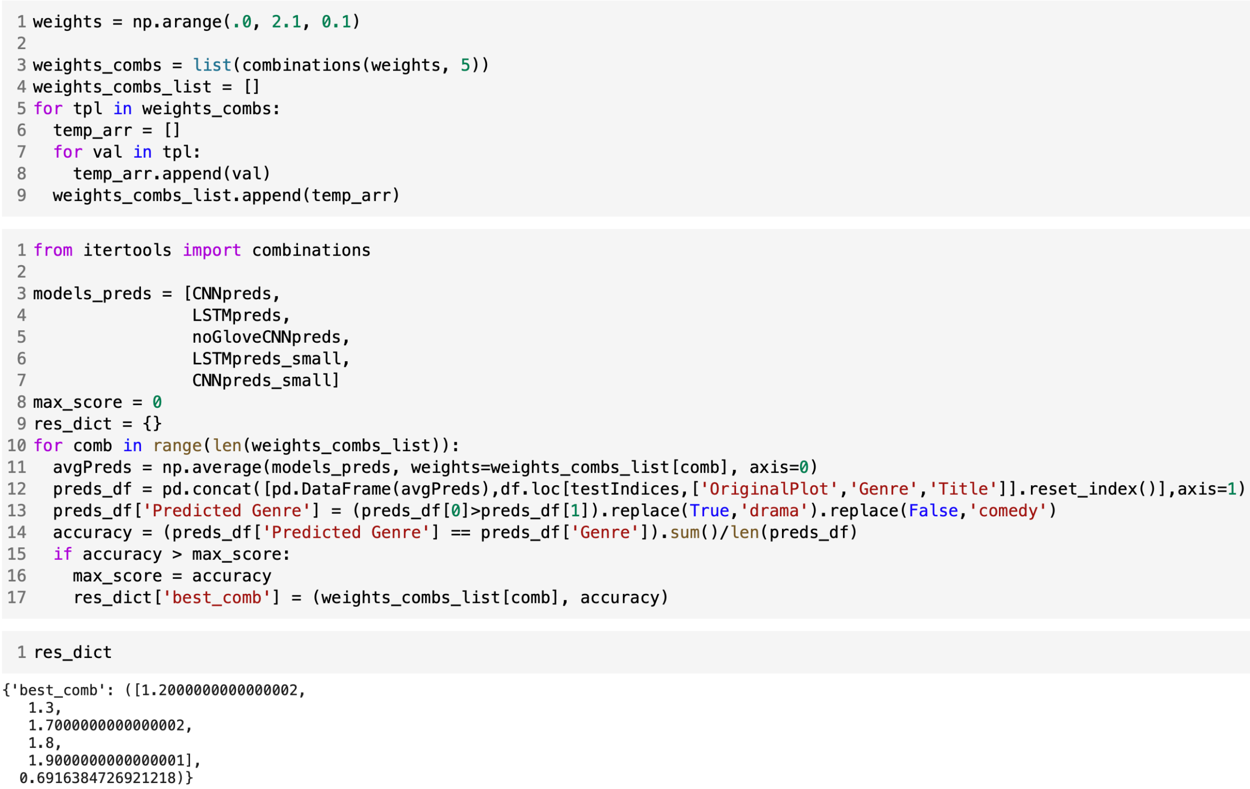
Additionally, special function was created to transform any input plot into vector and send it into model. This function is presented on Fig. 13.



```python
def pred_your_synop(test_plot):
  lower = test_plot.lower()
  removed = []
  removed.append(' '.join([word for word in lower.split() if word not in nltk.corpus.stopwords.words('english')]))
  test_token = tokenizer.texts_to_sequences(removed)
  test_token = np.array(test_token, dtype='int32')
  test_data = pad_sequences(test_token, maxlen=max_len)
  test_preds = modelSmall.predict(test_data)
  answ = 'comedy' if test_preds[0][0]<test_preds[0][1] else 'drama'
  return answ
```

Figure 13: Function to vectorize input text and make predictions

Finally, the ensemble of all 5 models mentioned above was created. To determine the best combination of weights of each model special algorithm was prepared. It is shown on Fig. 14.



```python
weights = np.arange(.0, 2.1, 0.1)

weights_combs = list(combinations(weights, 5))
weights_combs_list = []
for tpl in weights_combs:
  temp_arr = []
  for val in tpl:
    temp_arr.append(val)
  weights_combs_list.append(temp_arr)
```

```python
from itertools import combinations

models_preds = [CNNpreds,
                LSTMpreds,
                noGloveCNNpreds,
                LSTMpreds_small,
                CNNpreds_small]
max_score = 0
res_dict = {}
for comb in range(len(weights_combs_list)):
  avgPreds = np.average(models_preds, weights=weights_combs_list[comb], axis=0)
  preds_df = pd.concat([pd.DataFrame(avgPreds),df.loc[testIndices,['OriginalPlot','Genre','Title']].reset_index()],axis=1)
  preds_df['Predicted Genre'] = (preds_df[0]>preds_df[1]).replace(True,'drama').replace(False,'comedy')
  accuracy = (preds_df['Predicted Genre'] == preds_df['Genre']).sum()/len(preds_df)
  if accuracy > max_score:
    max_score = accuracy
    res_dict['best_comb'] = (weights_combs_list[comb], accuracy)
```

```python
res_dict
```

```
{'best_comb': ([1.2000000000000002,
   1.3,
   1.7000000000000002,
   1.8,
   1.9000000000000001],
  0.6916384726921218)}
```

Figure 14: Algorithm finding the best weights for ensemble.

6 Results

The final metrics of CNN models with training set are presented on the Tab. 3.

Model	Accuracy
CNN, no GloVE encodings, news and films data	0.66
CNN, GloVE encodings, news and movies data	0.65
CNN, GloVE encodings, only movies data	0.86

Table 3: Accuracy scores of CNN models.

The final metrics of LSTM-CNN models with training set are presented on the Tab. 4.

Model	Accuracy
LSTM, GloVE encodings, news and films data	0.62
LSTM, GloVE encodings, only movies data	0.8

Table 4: Accuracy scores of LSTM-CNN models.

The accuracy scores of all 5 models with validation data are shown on the Tab. 5 below.

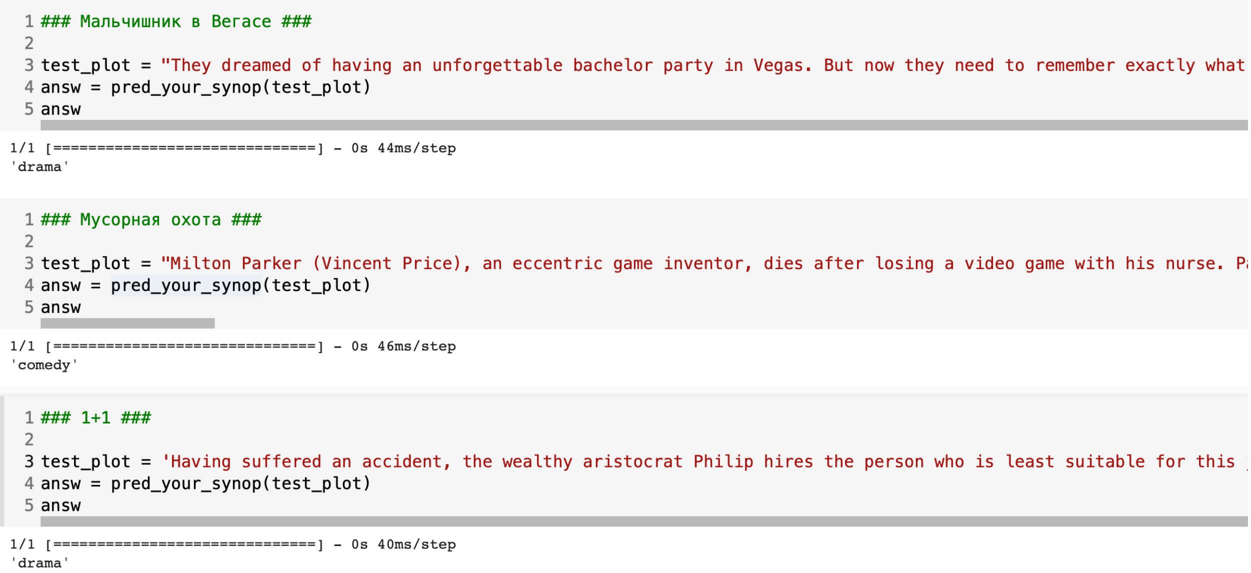
Model	Accuracy
CNN, GloVE encodings, news and movies data	0.623
LSTM, GloVE encodings, news and films data	0.657
CNN, no GloVE encodings, news and films data	0.681
LSTM, GloVE encodings, only movies data	0.656
CNN, GloVE encodings, only movies data	0.666

Table 5: Accuracy scores of models with validation set.

As shown on Fig. 14, the highest accuracy score achieved with ensemble is 1.2 percentage point higher than the best individual model.

7 Conclusion

Eventually, the preparation and processing of datasets with news and short film plots during the execution of this course project. The final result of this work can be considered several models, which are capable of determine whether the input plot is comedy or drama. The example of using your own plot as input is shown on. Fig. 15.



```
### Мальчишник в Вегасе ###

test_plot = "They dreamed of having an unforgettable bachelor party in Vegas. But now they need to remember exactly what
answ = pred_your_synop(test_plot)
answ
```

```
1/1 [==============================] - 0s 44ms/step
'drama'
```

```
### Мусорная охота ###

test_plot = "Milton Parker (Vincent Price), an eccentric game inventor, dies after losing a video game with his nurse. Pa
answ = pred_your_synop(test_plot)
answ
```

```
1/1 [==============================] - 0s 46ms/step
'comedy'
```

```
### 1+1 ###

test_plot = 'Having suffered an accident, the wealthy aristocrat Philip hires the person who is least suitable for this
answ = pred_your_synop(test_plot)
answ
```

```
1/1 [==============================] - 0s 40ms/step
'drama'
```

Figure 15: Predictions with your own plots.

References

[1] Quan Hoang. Predicting Movie Genres Based on Plot Summaries, 2018.

[2] Alex Blackstock and Matt Spitz. Classifying movie scripts by genre with a memm using nlp-based features, 2008.

[3] Ka wing Ho. Movies genres classification by synopsis, 2011.