HOW DO YOU FEEL, MY DEAR

Fabio Brambilla - 978983

ABSTRACT

The project aims at studying fictional scripts of several movies and TV series under the emotional profile focussing on how this emotional profile changes in time along the evolution of the movie story and how it is affected by the various relations among the different characters.

For the purposes of the project it has been decided to use a categorical representation of emotions.

DATA SOURCES

The datasets used to obtain the data required for the project are:

- Emotion detection from Text: available on the kaggle.com website. This dataset contains 40000 tweets classified into 13 different emotions.

- Cornell Movie-Dialogs Corpus: obtained via the convokit library. containing an extensive metadata-rich collection of fiction conversations extracted from raw film scripts.

EMOTION PREDICTION APPROACH

- Dataset preprocessing
- Text vectorization
- Model building
- Model training and testing
- Evaluation

EMOTION DATASET PREPROCESSING

- Removal of unnecessary columns: Removing the column of tweets ids.
- Removal of all rows in the table with 'angry', 'empty', 'boredom', 'enthusiasm' sentiment.
- Text cleaning: use the text_hammer library to perform text cleaning.
- Dataset split for train, test and validation.

```
emotion_dataset.drop(['tweet_id'], axis=1, inplace=True)
```

```
em = emotion_dataset[emotion_dataset.sentiment != 'anger']
em = em[em.sentiment != 'empty']
em = em[em.sentiment != 'boredom']
em = em[em.sentiment != 'enthusiasm']
emotion_dataset = em
```

```
def text_preprocessing(df, col_name):
    column = col_name
    df[column] = df[column].apply(lambda x:str(x).lower())
    df[column] = df[column].apply(lambda x: th.cont_exp(x))
    df[column] = df[column].apply(lambda x: th.remove_emails(x))
    df[column] = df[column].apply(lambda x: th.remove_html_tags(x))
    df[column] = df[column].apply(lambda x: th.remove_special_chars(x))
    df[column] = df[column].apply(lambda x: th.remove_accented_chars(x))
    df[column] = df[column].apply(lambda x: th.make_base(x))
    return(df)
```

```
train_emotion_dataset, val_emotion_dataset, test_emotion_dataset = train_val_test(emotion_dataset)
```

TEXT VECTORIZATION

- Tokenising text using the tokenizer object provided by the Keras library.
- Padding tokenized text using the pad_sequence method provided by the Keras library.
- Vectorize padded texts using Glove word embedding.

```
from keras.preprocessing.text import Tokenizer
num_words = 15000
tokenizer=Tokenizer(num_words,lower=True)
df_total = pd.concat([train_emotion_dataset['text'], test_emotion_dataset.text], axis = 0)
tokenizer.fit_on_texts(df_total)
```

```
X_train=tokenizer.texts_to_sequences(train_emotion_dataset['text'])
X_train_pad=pad_sequences(X_train,maxlen= 300,padding='post')
X_test = tokenizer.texts_to_sequences(test_emotion_dataset.text)
X_test_pad = pad_sequences(X_test, maxlen = 300, padding = 'post')
X_val = tokenizer.texts_to_sequences(val_emotion_dataset.text)
X_val_pad = pad_sequences(X_val, maxlen = 300, padding = 'post')
```

```
import gensim.downloader as api
glove_gensim  = api.load('glove-wiki-gigaword-100')
```

MODEL BUILDING

- Build a sequential model provided by keras library. A bi-LSTM model was used in order to be able to retain word memory in succession and thus have meaningful word sequences.

- Five other models provided by the sklearn library were also used to test their effectiveness:

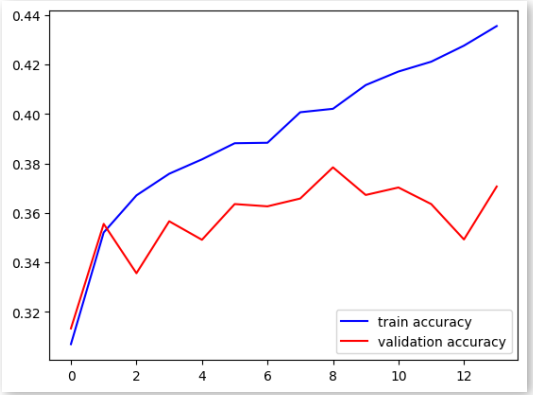
  - Logistic Regression
  - MLP
  - Decision Tree
  - Random Forest
  - K-Nearest Neighbors

```
EMBEDDING_DIM = vector_size
class_num = 9
model = Sequential()
model.add(Embedding(input_dim = num_words,
 output_dim = EMBEDDING_DIM,
 input_length= X_train_pad.shape[1],
 weights = [gensim_weight_matrix],trainable = False))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(100,return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(200,return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(100,return_sequences=False)))
model.add(Dense(class_num, activation = 'softmax'))
model.compile(loss = 'categorical_crossentropy', optimizer = 'adam',metrics = 'accuracy')
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
```

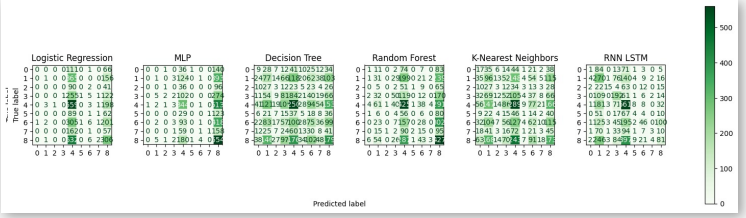
MODEL TRAINING AND TESTING

- The models were trained using 70% of the texts for each class.
- The Bi-LSTM model was trained using a maximum of 25 epochs, with a batch size of 120 and an early stopping patience of 5. Results after training showed that the model started overfitting around the tenth epoch, stopping at the thirteenth epoch with a training accuracy = 43% and a validation accuracy = 37%.
- The other 5 models in the sklearn library were trained with the default parameters.



Model	Accuracy	Avg Precision (macro)	Avg Recall (macro)	Avg F1-score (macro)	Avg Precision (weighted)	Avg Recall (weighted)	Avg F1-score (weighted)
Logistic Regression	0.254712	0.210287	0.126583	0.079188	0.267017	0.254712	0.156971
MLP	0.262827	0.175594	0.132326	0.085666	0.205055	0.262827	0.166659
Decision Tree	0.179843	0.126005	0.127276	0.126189	0.181283	0.179843	0.180006
Random Forest	0.251832	0.133022	0.138277	0.116635	0.193228	0.251832	0.196658
K-Nearest Neighbors	0.182199	0.120863	0.123943	0.118914	0.169798	0.182199	0.173093
RNN LSTM	0.307068	0.297926	0.223032	0.193635	0.307216	0.307068	0.248814

	Precision
Logistic Regression	0.239529
MLP	0.156806
Decision Tree	0.180366
Random Forest	0.239005
K-Nearest Neighbors	0.181152
RNN LSTM	0.319372



EVALUATION

- The evaluation of all six models showed that the most accurate prediction was achieved by the Bi-LSTM model with an accuracy of 31% for 9 classes on the test dataset.

- Preprocessing
- Emotion prediction

SCRIPT DATASET

SCRIPT DATASET PREPROCESSING

- The object containing the datasets for scripts, actors and utterances is obtained using the convokit library.
- The three datasets are merged and only a few useful columns are retained.
- Once the dataset containing the useful information for analysis purposes has been obtained, the scripts must be cleaned and vectorised. This is done with the same method used for the emotion dataset adding a new column 'cleaned_text'.

	id	conversation_id	meta.movie_idx_x	speaker	meta.character_name	meta.gender	reply_to	text	cleaned_text
0	L1045	L1044	m0	u0	BIANCA	f	L1044	They do not!	they do not
1	L985	L984	m0	u0	BIANCA	f	L984	I hope so.	i hope so
2	L925	L924	m0	u0	BIANCA	f	L924	Let's go.	let us go
3	L872	L870	m0	u0	BIANCA	f	L871	Okay -- you're gonna need to learn how to lie.	okay you re go to need to learn how to lie

EMOTION PREDICTION

- The prediction of emotion was performed on all texts in the dataframe with all previously trained models. For each model, a column was added to the dataframe whose value corresponds to the emotion predicted by that model on the text.

decision_tree	MLP	random_forest	logistic_regression	k_n_n	rnn LSTM
8	4	4	4	4	4
3	4	8	4	8	4
4	4	4	4	4	4
8	8	4	4	8	4
8	8	8	8	4	4

EMOTIONAL PROFILE

- First of all, since the script dataset is not sorted, the dataset is sorted by 'id' and 'conversation_id'. In this way, the entire script of a film is sorted by time sequence and it is easier to extract the rows of the time sequence of a character's script.

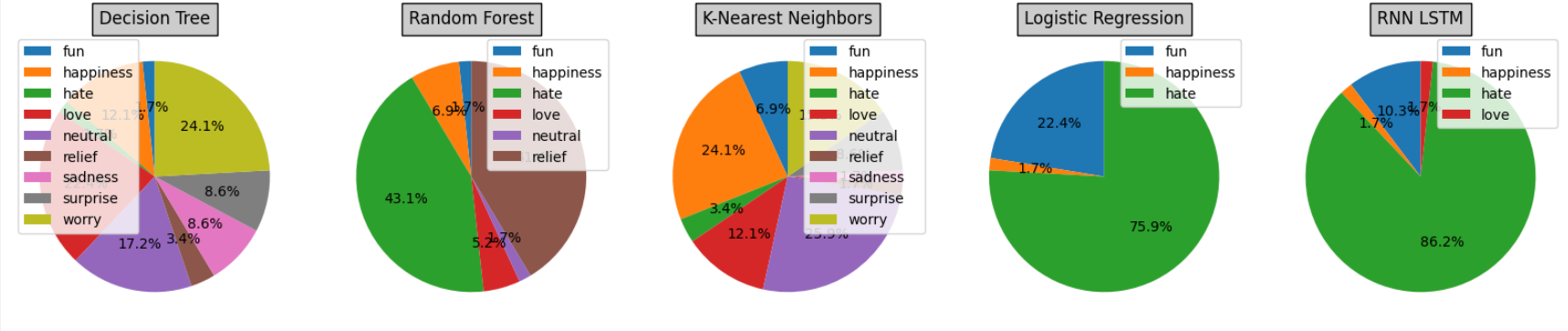
- Subsequently, the speaker id is used to extract all the rows of that specific character and create a new dataframe with all his rows.

	id	conversation_id	meta.movie_idx_x	speaker	meta.character_name	meta.gender	reply_to
0	L49	L49	m0	u0	BIANCA	f	None
1	L50	L49	m0	u3	CHASTITY	?	L49
2	L51	L49	m0	u0	BIANCA	f	L50
3	L59	L59	m0	u9	PATRICK	m	None
4	L60	L59	m0	u8	MISS PERKY	?	L59

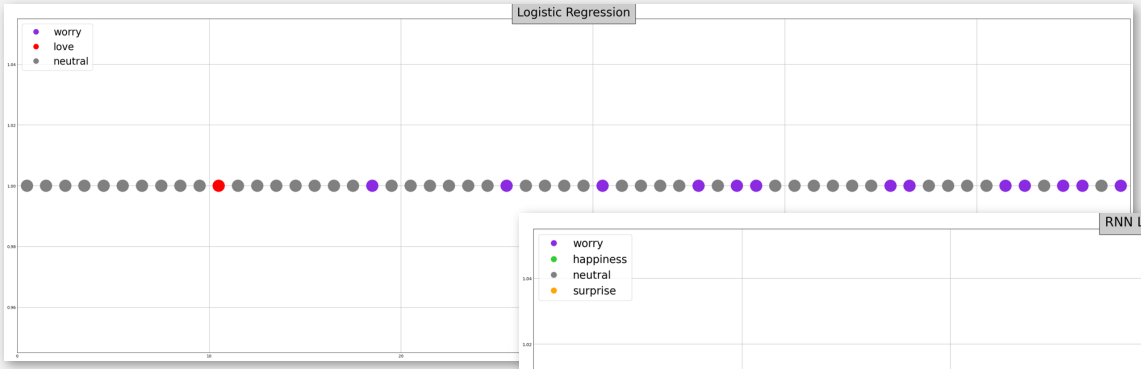
EMOTION VISUALIZATION

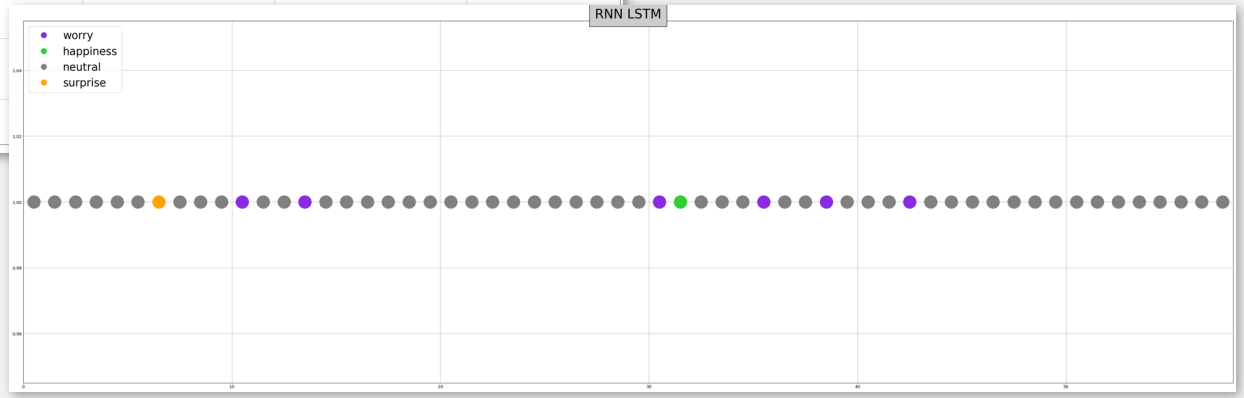
As far as the visualisation of the actor's emotional state is concerned, it was chosen to observe:

- The actor's general emotional state as a percentage based on his lines
- The actor's emotional state during the course of the film
- Which emotions prevail during the exchange of lines with another character
- What are the prevailing emotions according to the gender of the character he/she is interacting with

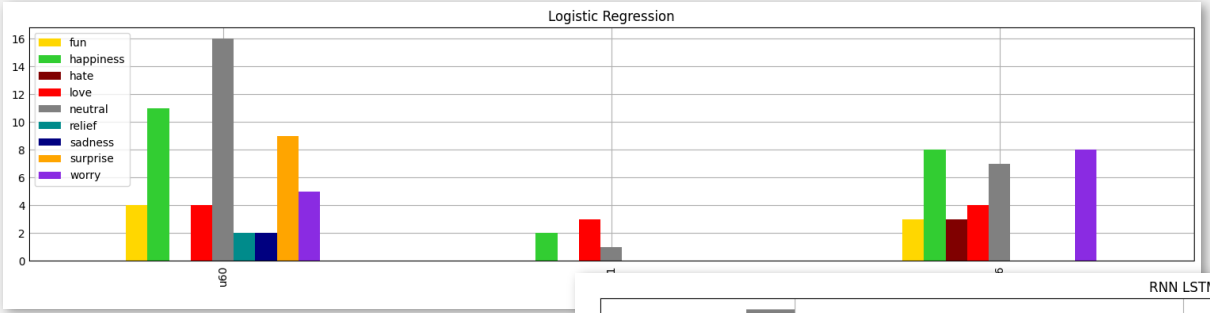


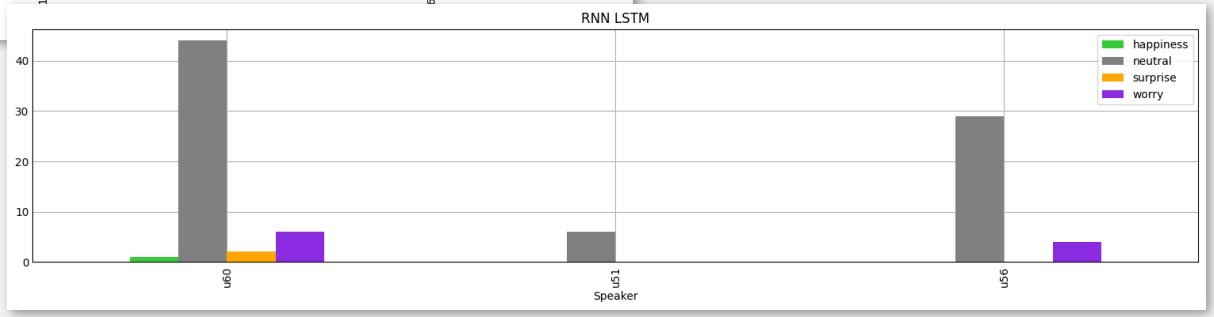
ACTOR'S GENERAL EMOTIONAL STATE AS A PERCENTAGE BASED ON HIS LINES



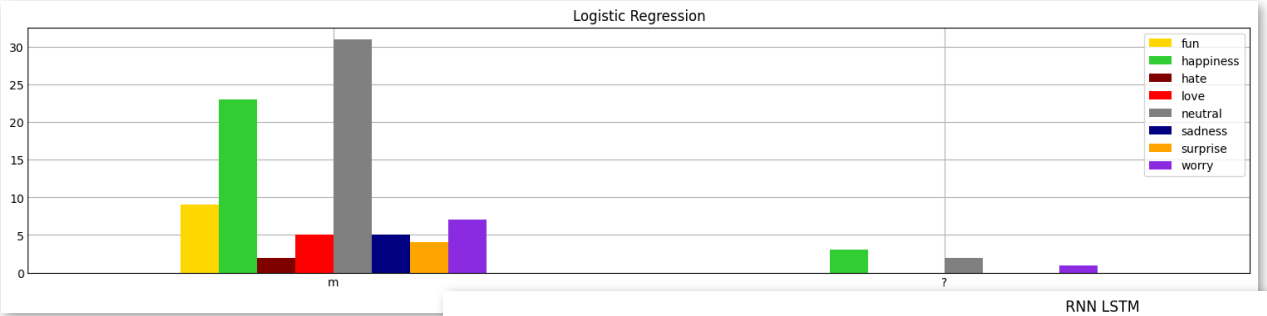


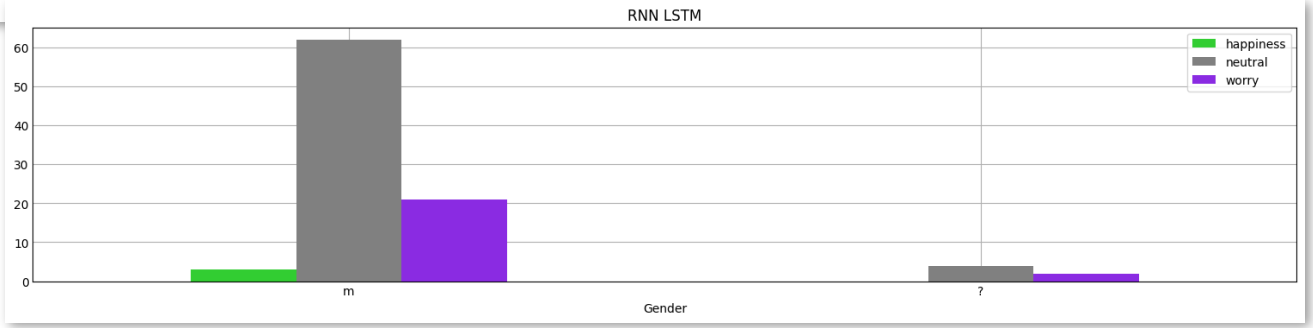
ACTOR'S EMOTIONAL STATE DURING THE COURSE OF THE FILM





WHICH EMOTIONS PREVAIL DURING THE EXCHANGE OF LINES WITH ANOTHER CHARACTER





WHAT ARE THE PREVAILING EMOTIONS ACCORDING TO THE GENDER OF THE CHARACTER HE/SHE IS INTERACTING WITH