ITCS-6100 – Big Data Analytics for Competitive Advantage

Project

Detecting and Rating Humor and Offense

Project Report
submitted by

Nisha Ramrakhyani
Anisha Kakwani
Punit Mashruwala

At
The University Of North Carolina at Charlotte

Detecting and Rating Humor and Offense

Current Date: 27/12/2020

Abstract:

Humor, like most figurative language, poses interesting linguistic challenges to NLP, due to its emphasis on multiple word senses, cultural knowledge, and pragmatic competence. Humor is an essential but most fascinating element in personal communication. Building computational models to discover the structures of humor and recognize humor remains a challenge because the reaction that make people laugh can hardly be generalized. Factors like age, gender, and socio-economic status are known to have an impact on the perception of a joke. In this project, we worked on a dataset with labels and ratings from a balanced set of age groups from 18-70. The annotators also represented a variety of genders, political stances, and income levels. Our task was to predict whether the text is humorous, if it is humorous, what is the humor rating and humor controversy whereas if the text is offensive, how offensive it is. And our approach towards solving this problem was building a model that learns to identify humorous jokes from almost 8000 labeled instances. We have worked on a real-world dataset that is too dynamic and our model tries to mimic human brain and deduces if the text is humorous or offensive.

1. Introduction:

This is the problem statement of a competition in CodaLab. It is divided into 4 tasks-

In Task 1a, we are supposed to predict if a text is humorous. We have used multiple models to predict if a text is humorous and the model that gave the highest accuracy is Colbert. It gave an accuracy of 91.5% and F-Score of 93.15% putting us at the 15th position on the leaderboard of competition that has around 50 participants on the leaderboard and 250 total participants.

Task 1a Humor Detection						
#	User	Entries	Date of Last Entry	Team Name	F-Score ▲	Accuracy ▲
15	Nisha_Ramrakhyani	102	12/26/20	Team NAP	0.9315 (20)	0.9150 (15)

In Task 1b, we are supposed to predict if the text is classed as humorous, how humorous it is (for an average user). The values vary between 0 and 5. We have used Neural network model with word embedding and reLu, sigmoid and linear to predict humor_rating. And the metric used for this task is RMSE. We have got the RMSE of 0.88 with Neural network model putting us at the 22nd position on the leaderboard of competition that has around 250 participants.

Task 1b Average Humor Score					
#	User	Entries	Date of Last Entry	Team Name	RMSE ▲
22	Nisha_Ramrakhyani	102	12/26/20	Team NAP	0.8846 (22)

In Task 1c, if the text is classed as humorous, we are supposed to predict if the humor rating would be considered controversial i.e., the variance of the rating between annotators is higher than the median. (This is a binary task). We have used Neural network model with word embedding and reLu, sigmoid and linear layers to predict humor_controversy. The metric used for this task is accuracy. We have got the accuracy of 53.3% with Neural network model putting us at the 12th position on the leaderboard of competition that has around 250 participants.

Task 1c Humor Controversy						
#	User	Entries	Date of Last Entry	Team Name	F-Score ▲	Accuracy ▲
12	Nisha_Ramrakhyani	103	12/27/20	Team NAP	0.3763 (23)	0.5332 (12)

In Task 2 we are supposed to predict how generally offensive a text is for users. We have used neural network model with word embedding and reLu, sigmoid and linear layers to predict the average offensiveness score. The metric used for this task is RMSE and we have got the RMSE of 0.918 with Neural network model putting us at the 18th position on the leaderboard of competition that has around 250 participants

Task 2 Average Offensiveness Score					
#	User	Entries	Date of Last Entry	Team Name	RMSE ▲
18	Nisha_Ramrakhyani	103	12/27/20	Team NAP	0.9190 (17)

2. Is the text humorous ? If yes, how humorous it is? Is the text offensive? If yes, how offensive is it ?

Humor is a highly subjective phenomenon, a piece of text may be humorous for one person and offensive for the other depending on various factors like age, gender, and socio-economic status. These factors have an impact on the perception of a joke. In this task, we must work on a dataset collected from a balanced set of age groups and a variety of genders, political stances, and income levels. We are supposed to analyze if the text is humorous and if it is humorous, how humorous it is, or if it is offensive, how offensive it is. To solve this problem, we have built models that perform text classification and have achieved an accuracy of 91.15% for humor detection using Colbert model.

3. Data from Codalab

CodaLab has provided training and test datasets in **Train.csv** and **public_dev.csv** respectively.

<u>Train.csv:</u>

This is the training dataset with 8000 records. In this data, we have 6 columns**: ID, Text, is_humor, humor_rating, humor_controversy, offense rating.**

This data was collected by asking people of different age and gender the following question:

- Is the intention of this text to be funny? (Rating from 0 to 1)

- [If so] How humorous do you find the text? (Rating from 0 to 5)
- Is this text generally offensive*? (Rating from 0 to 1)
- [If so] How generally offensive is the text? (Rating from 0 to 5)

Note: Annotators were not asked questions 2 and 4 if they did not answer 'yes' to the previous question.

Data format:

id	text	is_humor	humor_rating	humor_controversy	offense_rating
1	Text for the first row	1	1.126	0	3.098
2	Text for the second row	0		1	1.282
3	Text for the third row	1	3.983	1	1.644

Dealing with the missing data

As we can see in the above-mentioned table, we do have some missing data in the humor_rating and humor_controversy columns. If the value for is_humor is 0, the humor_rating and humor_controversy should be 0 as well. So, we have filled the missing data with '0'.

<u>public_dev.csv</u>

This is the test dataset with 1000 records. In this dataset, we have 2 columns: ID and Text.

From this data we need to predict the values for 4 columns (is_humor, humor_rating, humor_controversy, offense_rating) in respective 4 tasks.

Data format:

id	text
1	Text for the first row
2	Text for the second row
3	Text for the third row

4. Approach and Experiments

4.1 Motivation behind approaches

Initially we started experimenting with different models- Naïve Bayes, Random-forest and neural network model for the text classification on Kaggle short jokes dataset while we were waiting to get the training dataset from CodaLab. We were familiar with these algorithms as they were discussed in the class and the Kaggle dataset of short jokes gave an accuracy of more than 93% with these algorithms.

After getting the dataset from CodaLab, we used the same models that we built for the Kaggle dataset and we managed to get an accuracy of 70% - 80% with these models. However, we were looking for a better approach for our problem statement. After conducting a few experiments and discussing the results with professor, he suggested to play around with Neural networks having word embeddings, different activation models, different number of layers and Colbert model. We started incorporating professor's suggestions into our approaches and connected with him on a weekly basis. We built different models- Linear SVC, GRU, LSTM and all of them gave an accuracy of about 70-80%.

We got the accuracy of more than 80% for humor detection with Colbert model when we tested it for the first time and that is how we were motivated to use Colbert model for our problem statement. We experimented with different parameters, changed number of epochs, batch-size, etc. and we managed to achieve an accuracy of 91.5% in the final week of our course.

Also, during the lectures, at the time of group presentations , we learned the significance of keeping and removing stop words during text classification from one of the groups and that is how we were motivated to research in this area. In our case, they are useful, so we have not removed them.

4.2. Brief details of the models and methods used:

Naïve Bayes Model:

We started by lemmatizing the dataset and then vectorizing it using tf-idf and then trained the dataset on the Naïve Bayes classifier

Model Name	Accuracy (Task 1a)	RMSE (Task 1b)	Accuracy (Task 1c)	RMSE (Task 2)
Naïve Bayes model	79.3%	-	-	-

Neural Network Model (without word embeddings):

In this model, we started by incorporating the google-news model from keras and built the neural network with 3 layers. 1st was the hub layer, 2nd was dense layer with 16 neurons and reLu activation, last layer was the dense layer with 1 neuron.

Model Name	Accuracy (Task 1a)	RMSE (Task 1b)	Accuracy (Task 1c)	RMSE (Task 2)
Neural Network model (without word embedding)	78.6%	-	-	-

Linear SVC Model:

We started by replacing the special symbols from the data, stemming the words and then vectorizing them with the features finally training the model using the Linear SVC classifier.

Model Name	Accuracy (Task 1a)	RMSE (Task 1b)	Accuracy (Task 1c)	RMSE (Task 2)
Linear SVC model	58.3%	-	-	-

Neural Network Model (with word embeddings):

In this model, we tokenized the words including embeddings (glove) and created model with 4 layers - 1st was Embedding layer, 2nd was Dense layer with 32 neurons and reLu activation, 3rd was Dense layer with 16 neurons and reLu activation and last was dense layer with 1 neuron.

Model Name	Accuracy (Task 1a)	RMSE (Task 1b)	Accuracy (Task 1c)	RMSE (Task 2)
Neural Network model (with word embeddings)	83.50%	0.88	53.32	0.9190

LSTM Model:

In LSTM model we first tokenize, lemmatize, and stem the words using PorterStemmer from the tokenized word from the text sentence. We created the model with 4 layers - 1st was Embedding layer, 2nd was LSTM layer, 3rd was Dense layer with 6 neurons and reLu activation, and the last was dense layer with 1 neuron.

Model Name	Accuracy (Task 1a)	RMSE (Task 1b)	Accuracy (Task 1c)	RMSE (Task 2)
LSTM model	76%	-	51.8	-

GRU Model:

In this model we tokenized the data and added padding to the texts to feed it into the GRU model, the model consisted of 5 layers. 1st was Embedding layer from keras, 2nd was a layer of Bi-directional GRU, 3rd layer was dense layer with 12 neurons and reLu activation, 4th layer was a Dense layer with 6 neurons and reLu activation and the last layer was dense layer with 1 neuron and sigmoid activation.

Model Name	Accuracy (Task 1a)	RMSE (Task 1b)	Accuracy (Task 1c)	RMSE (Task 2)
GRU	63.7%	0.93	51.17	1.2

Colbert Model:

This model gave us the **highest accuracy** for Task 1a**.** We have used bert-based-uncased to tokenize the data which uses BertTokenizer. Then we have processed the data to feed it to model made with berth input layer along with one relu and one sigmoid layer. In total, this model has 9 layers in which 6 of them are keras layer and other layer are global average layer.

Model Name	Accuracy (Task 1a)	RMSE (Task 1b)	Accuracy (Task 1c)	RMSE (Task 2)
Colbert model	92.75%	1.13	51.8%	1.2

4.3 Brief details of experiments performed:

Sr no.	Model	Parameters changed	No. of experiments
1	Naïve Bayes	Layers, Number of Features	5
2	Neural Network (without word embeddings)	Layers, Epochs, Activation functions	5
3	Linear SVC	Layers, Number of Features	5
4	Neural Network (with word embeddings)	Epochs, Layers, Activation functions, Embeddings, Stop words, Embedding dimensions, Max length,	20
5	LSTM	Layer, Embedding dimensions, Max length, Vocabulary size	5
6	GRU	Layers	5
7	Colbert	Regex, Epochs, Vocabulary size	10

<table>
<tr><th colspan="6">Short description of the task</th></tr>
<tr><th>Date</th><th>Measure you use</th><th>Value of the measure</th><th>Method</th><th>No. of experiments</th><th>Comments</th></tr>
<tr><td>14th Oct 2020 - 21st Oct 2020</td><td>Loss and Accuracy</td><td>KAGGLE - [0.28042107820510864, 0.9264500141143799]
CODALAB -[1.4476656913757324, 0.7666666507720947]</td><td>Text Classification using NN - Tensor flow Hub</td><td>2</td><td>The observation we found after applying NN on the Coda-lab Trial data set is that Loss is comparatively higher than the Kaggle data set. Also, we have not used all the features available in the Coda-Lab data set</td></tr>
<tr><td>21st Oct 2020 – 28th Oct 2020</td><td>Accuracy</td><td>Using Most-frequent strategy to fill missing data on all columns:
Logistics Regression:
Accuracy: 96.667%
Random Forrest Regression:
Accuracy: 92.63%

Using Median strategy to fill missing data on all columns:
Logistics Regression:
Accuracy: 91.667%
Random Forrest Regression:
Accuracy: 94.45%

Using Most-frequent strategy to fill missing data on important columns:
Logistics Regression:
Accuracy: 98.33%
Random Forrest Regression:
Accuracy: 95.31%

Using Median strategy to fill missing data on important columns:
Logistics Regression:
Accuracy: 85%

Random Forrest Regression:</td><td>K-fold cross-validation using logistic regression and random forest regression</td><td>15</td><td>- After analyzing the NN model we observed the correlation between humor and offense values in the dataset

-Using k-fold cross-validation on coda-lab data the accuracy of prediction was improved but we still must change the way we deal with missing data</td></tr>
</table>

		Accuracy**:** 95.31%			
29th Oct 2020 – 04th Nov 2020	Accuracy, R2	**Colbert-Using-BERT-Sentence-Embedding-for-Humor-Detection** Accuracy: 94% **Predicting Kaggle features(200K records) using Codalab data(60 records)** Accuracy:56%	Linear Regression, Logistic Regression, and least-squares method	12	-Replicating the ColBert notebook gave us an accuracy of 94% when we used 5000 rows for training the data and 1000 rows for testing the data. -After predicting the values for the Kaggle dataset, we noticed that the values were very vague since the code lab data that we used was too less to get the proper prediction.
5th Nov 2020 – 18th Nov 2020	Accuracy, F-score	**Naïve Bayes model:** Accuracy:79.3% F-Score:85.43% **Neural Network model:** Accuracy: 78.6% F-Score: 81% **Linear SVC model:** Accuracy: 58.3 % F-Score: 62%	Naïve Bayes, Neural Network using TensorFlow hub and Linear SVC	15	After performing these experiments, we tested the models on the test data and submitted the same on Codalab, so far, we have secured 7th position on the leader board based on our best submission
19th Nov 2020 – 25th Nov 2020	Accuracy, F-score, RMSE	**Word Embeddings with Neural Network:** **Task1a** Accuracy: 81.3% F-Score: 85.51% **Task1b** RMSE : 0.93 **Task1c** Accuracy: 52.68% F-Score: 24.68% **Task 2** RMSE: 1.20 **Colbert model:** Accuracy: 80.8%	Neural Network with Word Embeddings, Colbert	12	We tried implementing the neural network model with word embeddings for different tasks and secured the following position on the leaderboard Task1a: 11th Task1b: 4th Task1c: 3rd Task2: 4th We also tried the Colbert model, but the accuracy was comparatively less than that of the neural network model

		F-Score: 82.51%			
26th Nov 2020 – 02nd Dec 2020	Accuracy, RMSE	**LSTM model:** **Task1a** Accuracy: 76% F-Score: 82% **Task1c** Accuracy : 51.8% F-Score: 50.6% **GRU (GATED RECURRENT UNIT) model:** **Task1a** Accuracy: 63.7 % F-Score: 72.68 % **Task1b** RMSE : 0.93 **Task1c** Accuracy: 51.17% F-Score: 44.60% **Task 2** RMSE: 1.20 **Word Embeddings with Neural Network:** **Task1a** Accuracy: 82.30% F-Score: 86.40% **Task1b** RMSE : 1.8 **Task1c** Accuracy: 51.27% F-Score: 44.60% **Task 2** RMSE: 1.57	Neural Network with Word Embeddings, GRU, and LSTM.	15	After implementing the neural network model with word embeddings and changing the parameters, we managed to get better accuracy for Task 1a. We tried implementing the tasks with LSTM and GRU models, but the accuracy was comparatively less than that of the neural network model.

03rd Dec 2020 – 9th Dec 2020	Accuracy, RMSE	**Word Embeddings with Neural Network:** **Task1a** Accuracy: 83.5% F-Score: 87.3% **Task1b** RMSE : 0.88 **Task1c** Accuracy: 53.32% **Task 2** RMSE: 0.919 **Colbert model** **Task1a** Accuracy: 86.3% F-Score: 89.22%	Neural Network with Word Embeddings, Colbert model.	30	Though our position on leaderboard has not changed much, we have been able to improve the scores for all the tasks this week by experimenting with various parameters in Neural network model and Colbert model.
10th Dec 2020 – 27th Dec 2020	Accuracy, RMSE	**Word Embeddings with Neural Network:** **Task1b** RMSE : 0.88 **Task1c** Accuracy: 53.32% **Task 2** RMSE: 0.919 **Colbert model** **Task1a** Accuracy: 91.5%	Neural Network with Word Embeddings, Colbert model.	30	We have managed to achieve below positions on the leaderboard (Total participants: 233): Task 1a: 15 Task 1b: 22 Task 1c: 12 Task 2: 18

5. Discussion and Related Work

Before working on the dataset provided by Codalab, we had started to work on a Kaggle dataset of short jokes as we were waiting for the training data of Codalab to work on. We managed to achieve an accuracy of 98% using the Logistic Regression model. We had also tried other different models like Neural Network, Linear Regression, Random Forest, Naive Bayes.

6. Conclusions and Open Problems

We have managed to achieve an accuracy of 91.5 % and F-Score of 93.15% in detecting how humorous a text is. It is quite difficult to work with humor. We are working towards improving our model to achieve better accuracy in predicting average humor score and average offensive score. Our work is placed at below link on GitHub:

Our data is available at: https://github.com/Anisha-Kakwani/Hahackathon---Detecting-Rating-Humor-Offense/tree/master

Our code is available at https://github.com/Anisha-Kakwani/Hahackathon---Detecting-Rating-Humor-Offense/tree/eighth-week(2ndDec-9thDec)

7. Contributions and Conflict of Interest Declarations.

Individual Contributions

	Nisha Ramrakhyani	Anisha Kakwani	Punit Mashruwala
WEEK 1: (14th Oct – 21st Oct)	-Participated in an initial discussion on how to proceed with the project -Referred 'Text Classification' Notebook which was shared by Professor on Shared Drive - **Data Preparation**: Imported the dataset, converted the categorical attributes to numerical values, partitioned the data into 'Training', 'Testing" and 'Validation' sets. - Submitted Temporary output file in Coda-lab competition.	- Participated in an initial discussion on how to proceed with the project -Referred 'Text Classification' Notebook which was shared by Professor on Shared Drive - **Building the Model**: Using the prepared data set, created the layers of NN using Tensor-flow and Keras, and then trained the model.	- Participated in an initial discussion on how to proceed with the project -Referred 'Text Classification' Notebook which was shared by Professor on Shared Drive - **Evaluating the model**: Used the trained model against the test dataset and against the Coda-lab dataset, compared the accuracy of the model for both the dataset and plotted the same for visualization.

WEEK 2 (21st Oct – 28th Oct)	- Participated in the strategy discussion - Performed k-fold and Logistics Regression using different strategies to fill missing data - Using 'Median' and 'Most-Frequent' Strategies to fill the missing data performed Random Forest Regression - Worked on project report simultaneously	- Participated in the strategy discussion - Analyzed NN model to find out co-relation between humor and offense - Performed various random seed experiments on Logistics Regression and Random Forest Regression - Worked on project report simultaneously	- Participated in the strategy discussion - Implemented Feature selection to shortlist the important features. - Performed Cross-Validation using Shuffle-split method from SKLearn on Coda-lab dataset - Worked on project report simultaneously
WEEK 3 (29th Oct-04th Nov)	--Predicted other features of the Kaggle dataset using Codalab dataset using Linear Regression. - Had a Discussion with Professor regarding the project impediment - **Data Collection:** Made Google forms for the survey and sent them across the contacts. -Explored the new training data of Codalab.	- Replicated the ColBert notebook Colbert-Using-BERT-Sentence-Embedding-for-Humor-Detection notebook. - Had a Discussion with Professor regarding the project impediment -**Data Collection:** Sent Google forms for the survey across the contacts.	-Predicted other features of the Kaggle dataset using the Codalab dataset using the LinearRegression and least-squares method. - Had a Discussion with Professor regarding the project impediment -**Data Collection:** Sent Google forms for the survey across the contacts.
WEEK 4 & 5 (05th Nov- 18th Nov)	-Used Tf-IDF vectorization and built a Naïve Bayes model for the new training data. -Tested the model against evaluation data released by Codalab. - Worked on project report and presentation simultaneously. -Had a discussion with Professor to discuss further steps.	-Built Neural Network model using Tensor-flow and Keras for the new training data. -Tested the model against evaluation data released by Codalab. - Worked on project report and presentation simultaneously. -Had a discussion with Professor to discuss further steps.	-Used Tfidf Vectorizer and built Linear SVC model for the new training data. -Tested the model against evaluation data released by Codalab. - Worked on project report and presentation simultaneously. -Had a discussion with Professor to discuss further steps.

WEEK 6 (19th Nov-25th Nov)	-Implemented Colbert model to predict task 1a -Tested the model against evaluation data released by Codalab. - Submitted 3 files on the Codalab leaderboard. - Worked on a project report simultaneously.	-Built Neural Network model using word embeddings to predict task 1b -Tested the model against evaluation data released by Codalab. - Submitted 1 file on the Codalab leaderboard. - Worked on a project report simultaneously.	- Built Neural Network model using word embeddings to predict task 1c & task 2 -Tested the model against evaluation data released by Codalab. - Submitted 3 files on the Codalab leaderboard. - Worked on a project report simultaneously.
WEEK 7 (26th Nov-02nd Dec)	-Implemented LSTM model to predict task 1a and task 1c. -Tested the model against evaluation data released by Codalab. - Worked on project reports and presentations simultaneously.	-Implemented a NN model using word embeddings(GloVe) to predict task 1a. -Tested the model against evaluation data released by Codalab. - Worked on project reports and presentations simultaneously.	-Implemented the GRU model to predict task 1a and task 1c. -Implemented the NN model to predict task 1a, task 1b, task 1c, and task 2. - Worked on project reports and presentations simultaneously.
WEEK 8 (03rd Dec-9th Dec)	-Performed Exploratory data analysis of dataset. - Had discussion with professor for further approach -Performed multiple experiments by changing different parameters on BERT and NN model for task 1a, task 1b, task 1c and Task 2. - Submitted 10 files on the Codalab leaderboard.	-Performed Error Analysis of training data for NN model - Had discussion with professor for further approach -Performed multiple experiments by using different word embeddings and different parameters in NN model for task 1a, task 1b, task 1c and Task 2. - Submitted 10 files on the Codalab leaderboard.	-Performed Error Analysis of training data for CNN and LSTM models - Had discussion with professor for further approach -Performed multiple experiments by changing different parameters on Colbert and NN model for task1a, task1b, task1c and Task 2. -Submitted 10 files on the Codalab leaderboard. - Worked on project reports and presentations simultaneously.

	- Worked on project reports and presentations simultaneously.	- Worked on project reports and presentations simultaneously.	
WEEK 9 (10th Dec – 24th Dec)	- Performed experiments on Colbert model - Worked on project reports simultaneously	- Performed experiments on Colbert model - Worked on project reports simultaneously	- Performed experiments on Colbert model - Worked on project reports simultaneously

Conflict of Interest Resolution:

There were not any conflicts within the team regarding any of the approaches. All the suggestions given by Professor and each team member were welcomed. Everyone in the team was equally involved and we used to have a weekly connect to keep everyone in the team on the same page. We also connected with Professor weekly to discuss the project impediments and brainstorm the approach for the week.

8. REFERENCES

https://www.tensorflow.org/hub/tutorials/tf2_text_classification

https://arxiv.org/pdf/2004.12765.pdf

https://medium.com/analytics-vidhya/text-classification-using-word-embeddings-and-deep-learning-in-python-classifying-tweets-from-6fe644fcfc81

https://github.com/manashpratim/Sarcasm-Detection/blob/master/Sarcasm_Detection.ipynb

https://www.youtube.com/watch?v=-8XmD2zsFBI

https://medium.com/analytics-vidhya/author-multi-class-text-classification-using-bidirectional-lstm-keras-c9a533a1cc4a

https://towardsdatascience.com/bert-text-classification-using-pytorch-723dfb8b6b5b

https://towardsdatascience.com/nlp-performance-of-different-word-embeddings-on-text-classification-de648c6262b

https://www.kaggle.com/moradnejad/200k-short-texts-for-humor-detection

9. APPENDIX

Notebook of Week 1:

text_classification_pdf
.pdf

Notebook of Week 2:

k-fold-notebook.pdf

Notebooks of week 3:

ColBERT
model-Copy1 - Jupyte

Predicting_Kaggle_fro
m_Codalab - Jupyter I

predicting_kaggle -
Jupyter Notebook.pdf

Notebooks of week 4 & 5:

Linear-SVC.pdf

Naive_Bayes.pdf

Neural Network
week5.pdf

Notebooks of week 6

ColBERT model -
Jupyter Notebook.pdf

NN-all-tasks -
Jupyter Notebook.pdf

Notebooks of week 7

Is-Humor_LSTM -
Jupyter Notebook.pdf

NN_with_remove_sto
p_words - Jupyter No

GRU.pdf

Notebooks of week 8

ColBERT model
updated- Jupyter Not

NN_with_WordEmbe
ddings - Jupyter Note

Notebooks for final week

ColBERT model- Humor Detection

NN_with_WordEmbeddings-Average humor ratings, offensive score, humor controversy

ColBERT model - Jupyter Notebook.pdf

NN_with_WordEmbeddings - Jupyter Note