A report on

EMOTION DETECTION FROM TEXT JOURNAL ENTRIES

By:

Kumarakrishna Valeti (2021A7PS2617G)

At

Ultrahive Healthcare Pvt Ltd A Practice School – 1 Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

Under the Guidance of

MENTORS - Dr. Laxminarayana Yashaswy Akella and Mr. Aditya Yadandla

FIC – Dr. D Sriram

Acknowledgement

I am incredibly indebted to Mr. Laxminarayana Yashaswy Akella and Mr. Aditya Yadandla, for their precious guidance and encouragement. Without their guidance and support, the report would have been incomplete.

I would also like to express gratitude towards my Faculty Mentor Dr. D. Sriram for his support and constant monitoring of the work on this project and Practice School division for providing the fantastic opportunity.

Lastly, I want to thank my Friends and Family, who were a constant support system to me.

Executive Summary

This report provides an update on the ongoing development of an emotion detection model for integration into the Masth app, a mental health support application designed for teenagers. The objective of this project is to enhance the app's functionality by incorporating a machine learning model capable of analyzing text journal entries and providing feedback on the expressed emotions.

Various models like Logistic Regression, Random forests and Long Short Term Memory have been implemented to accomplish this task. Tf-Idf vectorizer and GloVe have been used to process the words and obtain the relevant vector representations required by the models.

Finally Api's have been developed for the models and have been tested with a mock app, before being sent for final integration into the MASTH Guru app.

Both Random Forest and LSTM models have provided similar accuracy.

Introduction

In this project, I implemented AI and Deep Learning models for emotion detection from text. I explored techniques like Logistic Regression, Random Forests, and LSTM to achieve accurate results. Integrating GloVe word embeddings and TF-IDF vectorizer enhanced the models' performance in comprehending emotions expressed in text journal entries.

Crucially, I developed APIs to seamlessly integrate these models with the Masth app. These APIs facilitated smooth communication, allowing the app to receive journal entries from users and return precise emotion predictions as valuable feedback.

The significance of these models and APIs lies in their diverse applications in the field of Natural Language Processing (NLP). They enable sentiment analysis, providing businesses with insights into customer feelings and enhancing decision-making. In mental health support, the emotion detection system assists teenagers by offering personalized feedback and encouraging emotional expression.

Moreover, NLP-powered chatbots and virtual assistants deliver more natural and contextually relevant interactions. Additionally, emotion detection contributes to personalized content and product recommendations, enhancing user satisfaction and loyalty.

Overall, this project's implementation of AI and Deep Learning models and APIs holds immense promise in improving emotional awareness and supporting mental well-being across various domains.

Objective

The primary objective of this project is to successfully integrate an emotion detection system into the Masth app, thereby providing teenagers with a powerful tool to enhance self-awareness, facilitate emotional expression, and receive personalized mental health support.

By accurately analysing the emotions expressed in text journal entries, the system will enable users to gain insights into their emotional patterns, allowing for a deeper understanding of their mental well-being.

This information can be utilized not only within the Masth app but also to enhance other features, such as a movie recommender system that suggests films based on the detected emotions.

Additionally, the emotion detection system can be leveraged to notify the need for professional intervention, such as connecting users with a counsellor when necessary.

Ultimately, the successful implementation of the emotion detection system will empower teenagers to better navigate their emotional landscapes, promote their mental health, and provide them with the necessary support and resources tailored to their emotional needs.

PROJECT DESIGN

Dataset Selection and Analysis

We started by choosing Emotions for NLP dataset on Kaggle before merging it with Emotion from Text dataset to create a custom dataset suitable for our requirements.

Initially the training dataset had 1600 entries, but after merging and refactoring the labels of the 2 datasets we managed to obtain a training datasets with ~33000 entries. This helped increase the accuracy of the models significantly. In the dataset the text has been classified into 6 emotions namely – joy, sadness, anger, fear, love and surprise.

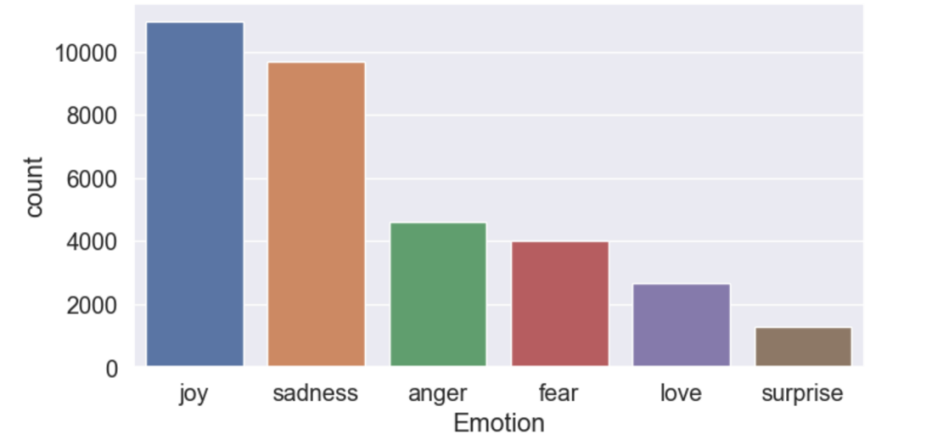
This is a sample view of the training dataset.

```
In [3]:
#print first 5 rows
df_train.head()
```

Out[3]:

	Text	Emotion
0	i didnt feel humiliated	sadness
1	i can go from feeling so hopeless to so damned...	sadness
2	im grabbing a minute to post i feel greedy wrong	anger
3	i am ever feeling nostalgic about the fireplac...	love
4	i am feeling grouchy	anger

The distribution of the text into the 6 emotion categories is as follows :



Data Cleaning

The same text classified into different emotions and can cause problems while training the model hence duplicate values need to be eliminated.

An example of duplicate values :

In [15]:
```
df_train[df_train['Text'].duplicated() == True]
```

Out[15]:

	Text	Emotion
5067	i feel on the verge of tears from weariness i ...	joy
6133	i still feel a craving for sweet food	love
6563	i tend to stop breathing when i m feeling stre...	anger
7623	i was intensely conscious of how much cash i h...	sadness
7685	im still not sure why reilly feels the need to...	surprise
8246	i am not amazing or great at photography but i...	love

In [16]:
```
df_train[df_train['Text'] == df_train.iloc[5067]['Text']]
```

Out[16]:

	Text	Emotion
1501	i feel on the verge of tears from weariness i ...	love
5067	i feel on the verge of tears from weariness i ...	joy

Code to remove duplicate entries:

In [17]:
```
index = df_train[df_train['Text'].duplicated() == True].index
df_train.drop(index, axis = 0, inplace = True)
df_train.reset_index(inplace=True, drop = True)
```

In [18]:
```
df_train[df_train['Text'].duplicated() == True]
```

Out[18]:

	Text	Emotion

Data Pre-processing

Data pre-processing plays a crucial role in preparing the text journal entries for the emotion detection models. The following steps were undertaken for data pre-processing:

Text Cleaning: The raw text data from the journal entries was subjected to cleaning procedures to remove unnecessary characters, symbols, and formatting issues. This step involved removing punctuation marks, special characters, and any HTML tags if present.

Stop Word Removal: Stop words, such as common words like "the," "and," or "is," were eliminated from the tokenized text. These words typically do not carry significant meaning and can potentially introduce noise to the analysis.

Lemmatization: Lemmatization is the process of reducing words to their base or root form, known as lemmas. This step helps to normalize the variations of words and reduce the dimensionality of the data. For example, lemmatization would convert words like "running" and "ran" to their lemma "run".

Model Training

Once the data pre-processing steps were completed, the next stage in the project involved training the emotion detection models. Several models were considered and implemented, including

Logistic Regression, Random Forests, and Long Short-Term Memory (LSTM) networks.
For the Logistic Regression and Random Forest models, the pre-processed text data was transformed using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique. TF-IDF assigns weights to each word in the text based on its frequency within a document and inversely to its frequency across the entire corpus. This transformation allowed the models to capture the importance of words within a specific context while reducing the impact of commonly occurring words.

For the LSTM (Long Short-Term Memory) model, the text data was enriched using GloVe (Global Vectors for Word Representation) embeddings. GloVe embeddings provide vector representations for words based on their co-occurrence patterns in a large corpus of text. This technique helps capture the semantic relationships between words and enables the LSTM model to understand the meaning and context of the text entries.

Due to computational limitations the LSTM model was only trained for 20 epochs, but in presence of enough GPU resources the LSTM model could be trained upto 50 epochs. The LSTM model implemented in this project consisted of three layers of bidirectional LSTM units.

The LSTM model used was a three layer bidirectional lstm, and the glove vectors used was the 200 dimensional set consisting of 1 billion entries.

MODEL EVALUATION and PERFORMANCE

F1-score was used to calculate accuracy for all the three models.

The trained emotion detection models, including Logistic Regression, Random Forest, and a three-layer bidirectional LSTM, demonstrated promising results in classifying emotions from text journal entries. In all three models the increase in size of training dataset contributed significantly to accuracy.

Logistic Regression achieved an accuracy of 91% after merging of dataset showing a significant rise in accuracy compared to the previous 86%

Random Forest, being an ensemble of decision trees, yielded an accuracy of 97% after merging of dataset showing a significant rise in accuracy compared to the previous 88%

The LSTM model, leveraging GloVe embeddings and bidirectional architecture, outperformed the other models, achieving an accuracy of 97% after merging of dataset showing a rise in accuracy compared to the previous 93%. The LSTM model was trained only for 20 epochs due to computational limitations, but further training upto 50 epochs keeping overfitting in mind could increase the accuracy further.

While the accuracy of the emotion detection models on the test dataset appears high, it is essential to acknowledge their limitations when applied to real-life diary entries. In real-life scenarios, the models may occasionally predict faulty or inaccurate emotions.

API Development

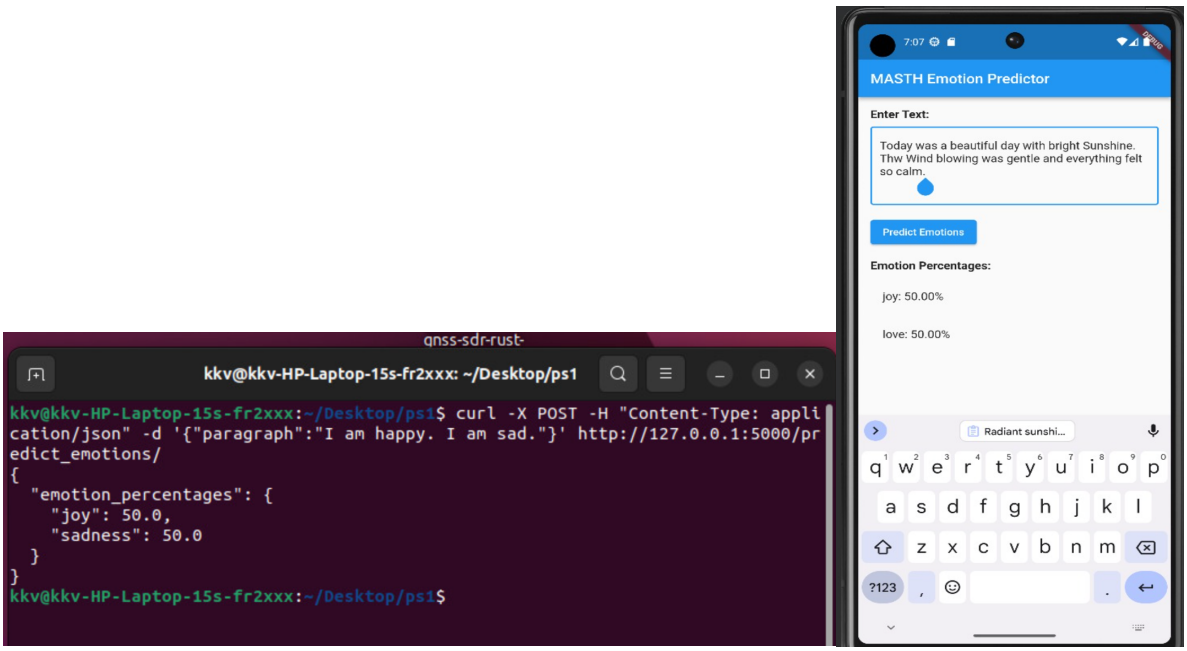
To seamlessly integrate the trained emotion detection models into the Masth app, APIs (Application Programming Interfaces) were developed. The APIs served as interfaces that facilitated communication between the app and the emotion detection models, allowing the app to leverage the models' capabilities for analysing text journal entries and providing feedback on the expressed emotions.

Flask was used to develop the respective API's, and the models were pickled using the pickle library of python and then these pickle files were directly called in the API's. for the LSTM implementation the tokenizer and label encoder were also pickled and loaded in the API.

Issues face with API development mainly included versioning issues and compatibility issues between mac os, windows and linux.

Before integrating the API's with the MASTH app they were tested with a mock app developed using android studio. The API's integration with API's of other projects was also tested on the mock app, before being sent for final integration into the MASTH app.

Input to the API is a paragraph in json format, the route of the API for detecting emotions is /predict_emotions . The response will be in json format containing the emotions detecte and their Percentages. The images below show the API being run locally as well as being executed in a mock app.



USE CASES

The AI and Deep Learning models in this project find diverse applications:

Mental Health Support App: Integrated emotion detection aids teenagers with personalized feedback, promoting emotional expression and self-awareness.

Social Media Sentiment Analysis: Identifies customer opinions, trends, and brand reputation for data-driven decisions and customer engagement.

Customer Feedback Analysis: Gauges satisfaction and preferences, enhancing customer experience and product improvements.

Emotionally Intelligent Chatbots: Offers empathetic responses, improving user interactions and satisfaction.

Automated Content Moderation: Identifies harmful content, ensuring a safer online environment.
Benefits and Impact:

Enhanced Emotional Well-being: Users gain insights into emotional patterns, fostering emotional intelligence and mental well-being.

Improved Customer Understanding: Businesses tailor products and services to customer needs.

Time and Cost Savings: Automated sentiment analysis saves time and costs.

Personalized Recommendations: Enhances user satisfaction and loyalty.

Real-time Intervention: Crucial support for users experiencing emotional distress.

Enhanced User Experience: More engaging and personalized interactions.

Social and Market Insights: Valuable data for marketing strategies and competitive analysis.

In summary, these models have broad applications, significantly impacting mental health support, customer insights, and user experiences.

LIMITATIONS and FUTURE WORK

The successful integration of the emotion detection system into the Masth app opens up a range of exciting future possibilities for enhancing mental health support for teenagers. Here are some potential future perspectives to consider:

Collaboration and Partnerships: Establishing collaborations with mental health professionals, researchers, and organizations to leverage their expertise in refining the emotion detection system. Collaborative efforts can enhance the system's accuracy, promote ethical considerations, and ensure alignment with best practices in mental health support.

Privacy and Ethical Considerations: Continuously addressing privacy concerns and ensuring the ethical use of user data. Adhering to strict data protection policies, maintaining transparency, and obtaining informed consent are crucial for building trust and maintaining user engagement.

There are several avenues for future work to enhance the emotion detection system within the Masth app:

Data Expansion: Expanding the dataset used for training the models is crucial to improve their generalization capabilities. A larger and more diverse dataset that encompasses a wide range of emotions and real-life scenarios will enhance the models' accuracy and reliability.

Transfer Learning: Leveraging transfer learning techniques, such as pre-training models like BERT on large text corpora, can greatly enhance the models' understanding of emotions in text. Fine-tuning these pre-trained models on the available dataset can improve the models' performance and capture intricate emotional patterns.

Multi-modal Integration: Exploring the integration of multi-modal data sources, such as images, voice recordings, or wearable device data, to provide a richer and more holistic understanding of users' emotions. Combining text journal entries with other modalities can enhance the accuracy and depth of emotional analysis.

CONCLUSION

The integration of an emotion detection system into the Masth app represents a significant step towards empowering teenagers with personalized mental health support. Through the implementation of various machine learning models, including Logistic Regression, Random Forest, and a three-layer bidirectional LSTM, the project has demonstrated promising results in classifying emotions from text journal entries.

The developed APIs, built using Flask, have successfully facilitated the seamless communication between the app and the emotion detection models. Users of the Masth app can now receive prompt and personalized feedback based on their expressed emotions, promoting self-awareness and facilitating emotional expression.

However, it is important to acknowledge the limitations of the models when applied to real-life diary entries. Factors such as subjectivity, context, and limited training data may occasionally result in faulty predictions. Future work should focus on addressing these limitations and refining the models' performance on real-life scenarios.

Appendices

The code for the models and the API's can be found on this Github Repository. The Emotions dataset for NLP and Emotions in Text datasets on Kaggle have been used to train the models. The GloVe vectors can be found on kaggle.

Breakdown of Logistic Regression and Random Forest models accuracies :

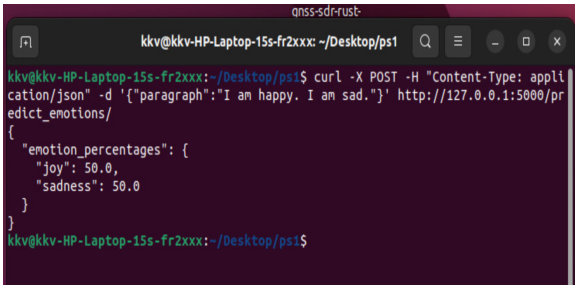
	precision	recall	f1-score	support
anger	0.93	0.88	0.91	583
fear	0.90	0.86	0.88	488
joy	0.89	0.98	0.93	1400
love	0.94	0.72	0.82	299
sadness	0.92	0.96	0.94	1204
surprise	0.95	0.61	0.74	173
accuracy			0.91	4147
macro avg	0.92	0.84	0.87	4147
weighted avg	0.91	0.91	0.91	4147

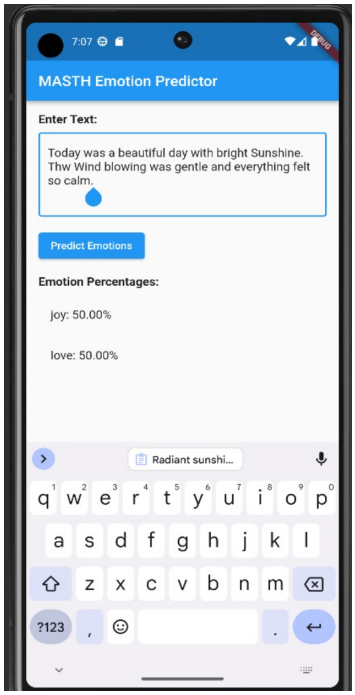
	precision	recall	f1-score	support
anger	0.94	0.97	0.96	583
fear	0.95	0.96	0.95	488
joy	0.98	0.97	0.98	1400
love	0.94	0.96	0.95	299
sadness	0.97	0.98	0.98	1204
surprise	0.96	0.90	0.93	173
accuracy			0.97	4147
macro avg	0.96	0.96	0.96	4147
weighted avg	0.97	0.97	0.97	4147

The breakdown of LSTM model's accuracy :

0	0.98	0.95	0.96	583
1	0.92	0.96	0.94	488
2	0.97	0.99	0.98	1400
3	0.97	0.90	0.93	299
4	0.97	0.99	0.98	1204
5	0.93	0.83	0.88	173
accuracy			0.97	4147
macro avg	0.96	0.94	0.95	4147
weighted avg	0.97	0.97	0.97	4147

API run locally and in mock app.





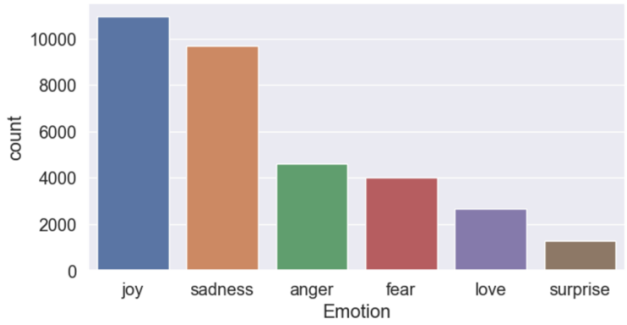
This is a sample view of the training dataset.

```
In [3]:
#print first 5 rows
df_train.head()
```

Out[3]:

	Text	Emotion
0	i didnt feel humiliated	sadness
1	i can go from feeling so hopeless to so damned...	sadness
2	im grabbing a minute to post i feel greedy wrong	anger
3	i am ever feeling nostalgic about the fireplac...	love
4	i am feeling grouchy	anger

The distribution of the text into the 6 emotion categories is as follows :



An example of duplicate values :

```
In [15]:
df_train[df_train['Text'].duplicated() == True]
```

Out[15]:

	Text	Emotion
5067	i feel on the verge of tears from weariness i ...	joy
6133	i still feel a craving for sweet food	love
6563	i tend to stop breathing when i m feeling stre...	anger
7623	i was intensely conscious of how much cash i h...	sadness
7685	im still not sure why reilly feels the need to...	surprise
8246	i am not amazing or great at photography but i...	love

```
In [16]:
df_train[df_train['Text'] == df_train.iloc[5067]['Text']]
```

Out[16]:

	Text	Emotion
1501	i feel on the verge of tears from weariness i ...	love
5067	i feel on the verge of tears from weariness i ...	joy

Code to remove duplicate entries:

```
In [17]:
index = df_train[df_train['Text'].duplicated() == True].index
df_train.drop(index, axis = 0, inplace = True)
df_train.reset_index(inplace=True, drop = True)
```

```
In [18]:
df_train[df_train['Text'].duplicated() == True]
```

Out[18]:

	Text	Emotion