

Avichai mazin: 211780267

Orel mazin: 213435902

## Emotion detection classification

Emotion detection from text is a key task in NLP with applications in fields like mental health, customer service, and social media analysis. Recognizing emotions in short texts, such as tweets, is especially challenging due to informal language and limited context.

Our work relates to efforts like the [GoEmotions](#) dataset, which highlights the complexity of capturing subtle emotional cues in text.

### Our Research Question:

*Which modeling approach best captures emotion in short-form social media text?*

This project focuses on **emotion classification** in short texts from Twitter. Our goal is to compare different modeling approaches and determine which one best captures emotional content in brief, informal text.

We evaluate three categories of models:

- **Classical machine learning models:** Logistic Regression, K-Nearest Neighbors (KNN), Random Forest, Support Vector Machines (SVM), XGBoost, and AdaBoost.
- **Deep learning:** Convolutional Neural Network (CNN).
- **Pre-trained language models:** fine-tuned BERT (added in the final section).

### Dataset Description

We used a publicly available [dataset](#) from kaggle of English-language tweets from twitter labeled with six basic emotions:

sadness (0), joy (1), love (2), anger (3), fear (4), surprise (5).

The dataset includes 416,809 samples, each consisting of a short tweet and its corresponding emotion label. All labels are manually annotated and verified through crowd-sourcing. Each data point includes the following fields:

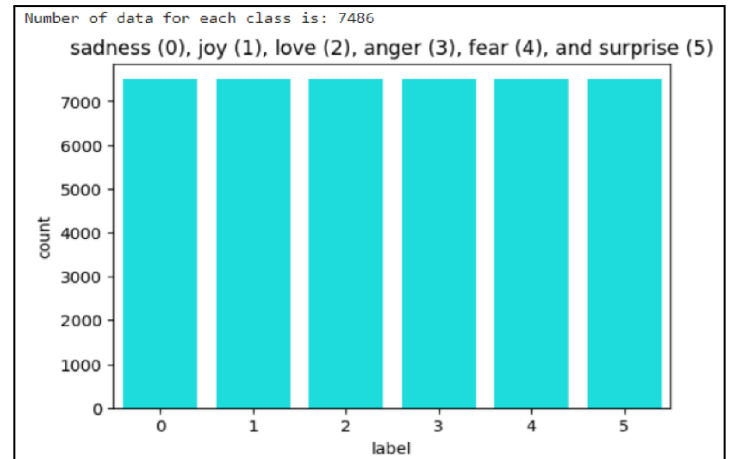
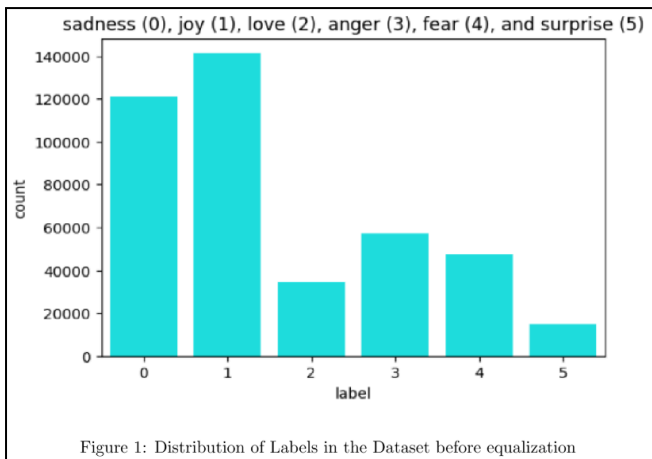
- **text:** the tweet content (string)
- **label:** the emotion class (integer 0–5)

There are no missing values in the dataset.

### Class Distribution:

A clear class imbalance firstly exists: **joy** and **sadness** are overrepresented, while **surprise** and **love** are rare, so we changed it:

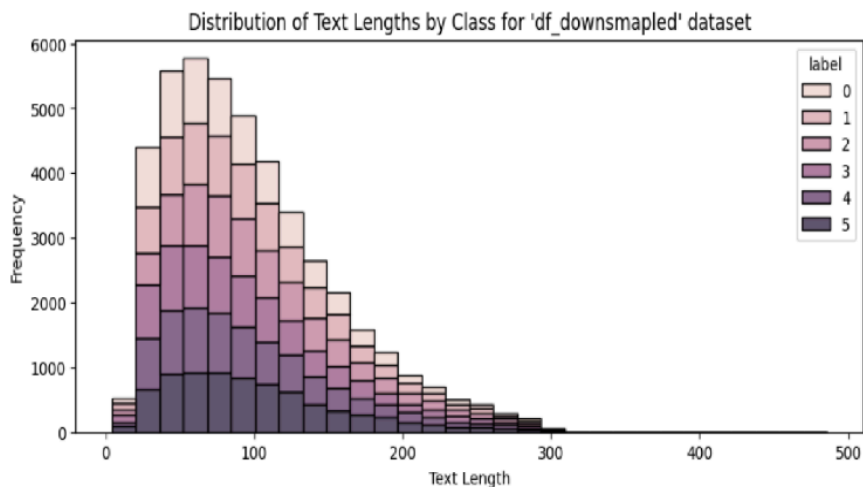
## Number of samples per class (before and after downsampling)



To draw meaningful and reliable conclusions from the models performance, it was essential to deeply understand the structure and characteristics of the dataset.

Therefore, a comprehensive Exploratory Data Analysis (EDA) was conducted prior to modeling, allowing for informed choices regarding preprocessing, model selection, and evaluation.

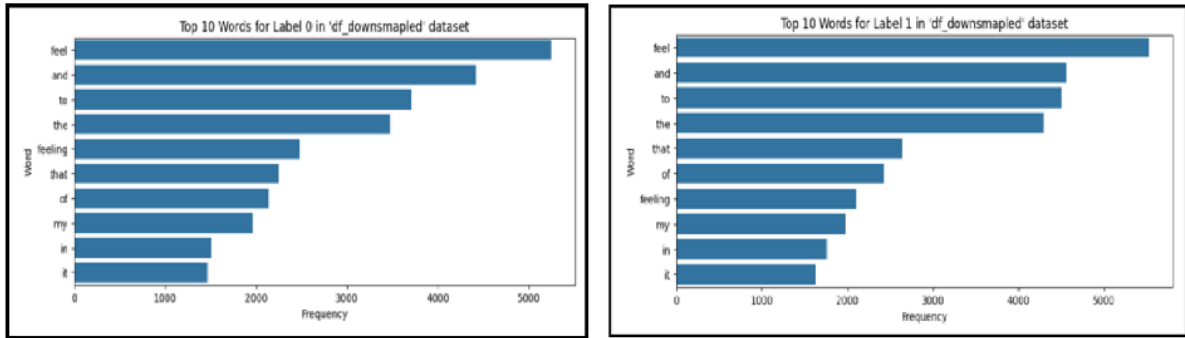
## EDA (Exploratory Data Analysis)



**Figure 1:** Tweet length distributions by emotion class. Most tweets are short (20–120 words), and all classes show a similar pattern.



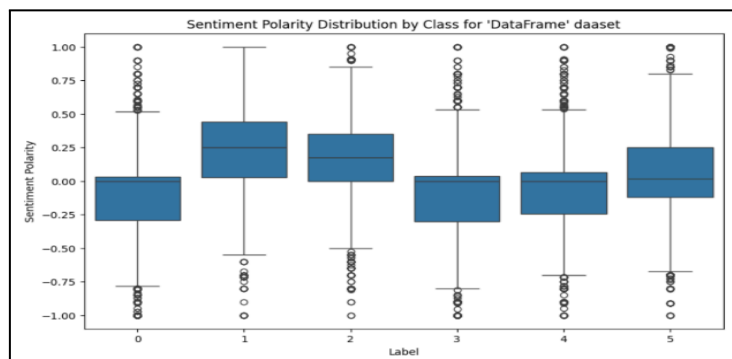
**Figure 2:** Most frequent words in each emotion class. Words like *feel* really *feeling* appear a lot, meaning people often say how they feel in tweets.



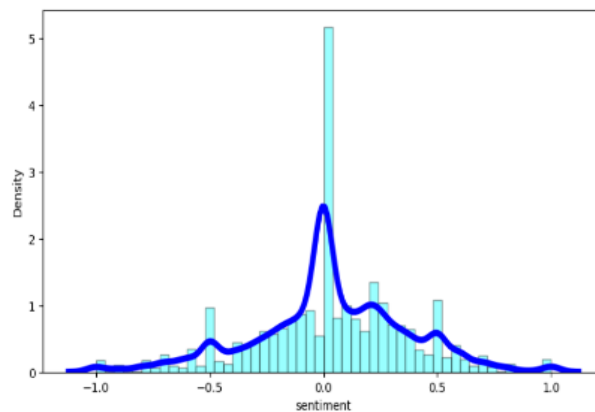
**Figure 3:** Bar charts showing the 10 most frequent words in each class. Many of these words (e.g., *feel*, *to*, *the*) appear in all classes and are not emotion-specific.

```
custom_stopwords = {"i", "in", "like", "feel", "feeling", "my", "the", "to", "still",
                    "for", "know", "littl", "think", "time", "thing", "would", "go",
                    "really", "feel", "am", "so", "get", "one", "to", "and", "at", "can",
                    "day", "way", "make", "me", "want", "could", "would", "tri", "u", "href", "http", "www", "com", "https"
}
```

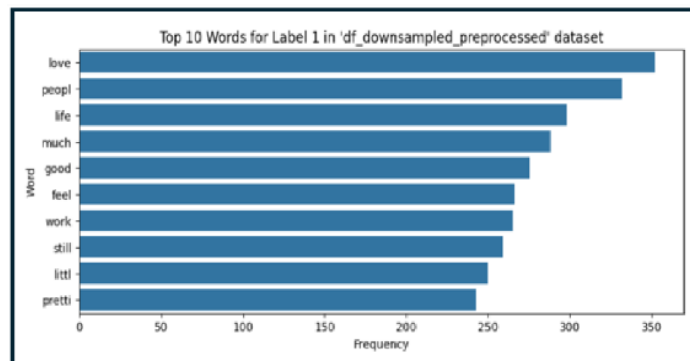
**Figure 4:** A list of common words that were removed because they don't help the model detect emotions.



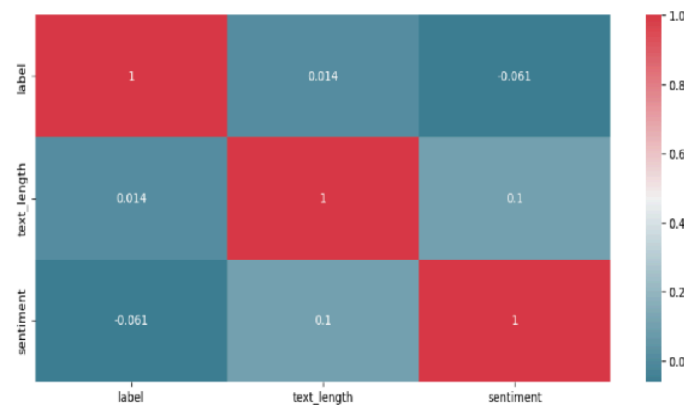
**Figure 5:** Boxplot showing the sentiment polarity for each emotion class. Positive emotions (e.g., joy, love) have higher median values, while negative emotions (e.g., sadness, anger, fear) have lower values. The dots show extreme sentiment scores (outliers).



**Figure 6:** Sentiment values are mostly close to zero, meaning tweets are generally neutral. The histogram shows how often each sentiment score appears, and the blue line shows how they are spread out.



**Figure 7:** After cleaning the data, the most frequent words in each class are more related to the emotion—like *love* in joy.



**Figure 8:** Correlation heatmap showing weak relationships between label, text length, and sentiment. None of the variables strongly predict the others.

## Algorithms used in the project

**Models, Tuned Parameters and Embedding Types**  
(Parameters selected via Grid Search, Random Search or validation-based tuning)

Model	Best Parameters (After Tuning)	Embedding Type
Dummy	Predicts most frequent or random class	-
KNN	22 neighbors, weighted by distance	GloVe / CountVectorizer
SVM	C=1.3, linear kernel, degree=4	GloVe / CountVectorizer
Random Forest	119 trees, depth=15, min split=5	GloVe / CountVectorizer
XGBoost	95 trees, depth=8, learning rate=0.05	GloVe / CountVectorizer
AdaBoost	160 trees, learning rate=0.3	GloVe / CountVectorizer
CNN	Embedding + Conv1D + MaxPooling + Dense	Self-trained

## Accuracy Comparison Between Models

In our experiments with classical machine learning models, we evaluated two different text embedding methods: **GloVe** (pre-trained word embeddings) and **CountVectorizer** (frequency-based embedding). These two approaches were chosen after reviewing common embedding techniques in the literature, as they were found to be particularly suited for short, informal text like tweets.

We observed that models trained with **CountVectorizer** consistently outperformed those using GloVe. This is likely because CountVectorizer better captures word frequency patterns in domain-specific or informal language, such as slang, abbreviations, and emojis commonly found in tweets.

While GloVe provides rich semantic relationships from large general corpora, it may not reflect the unique vocabulary used on social media platforms. Our results suggest that simple frequency-based methods can be more effective when the data distribution differs significantly from the GloVe training corpus.

It is worth noting that other embedding methods such as **Word2Vec**, or **TF-IDF**, were considered but not tested due to scope limitations. These alternatives may be explored in future work to further optimize performance and compare their ability to handle noisy, short-form text.

```
The train Accuracy for knn model is: 0.971529558276745
The test Accuracy for knn model is: 0.41763134461264473

The train Accuracy for linearSvc model is: 0.447122342201937
The test Accuracy for linearSvc model is: 0.43544078361531613

The train Accuracy for xgb model is: 0.8235834353779361
The test Accuracy for xgb model is: 0.4718388245770258

The train Accuracy for abc model is: 0.39173995324501837
The test Accuracy for abc model is: 0.3705476402493321

The train Accuracy for randomForest model is: 0.847573193810531
The test Accuracy for randomForest model is: 0.44367764915405167
```

Figure 13: GloVe embedding

```
The train Accuracy for knn model is: 0.9907445691930918
The test Accuracy for knn model is: 0.7525788497217069

The train Accuracy for linearSvc model is: 0.9429407461594733
The test Accuracy for linearSvc model is: 0.8784415584415585

The train Accuracy for xgb model is: 0.8622499284373907
The test Accuracy for xgb model is: 0.8485343228200372

The train Accuracy for abc model is: 0.3169110397251996
The test Accuracy for abc model is: 0.3044155844155844

The train Accuracy for randomForest model is: 0.8519449126936166
The test Accuracy for randomForest model is: 0.8473469387755102
```

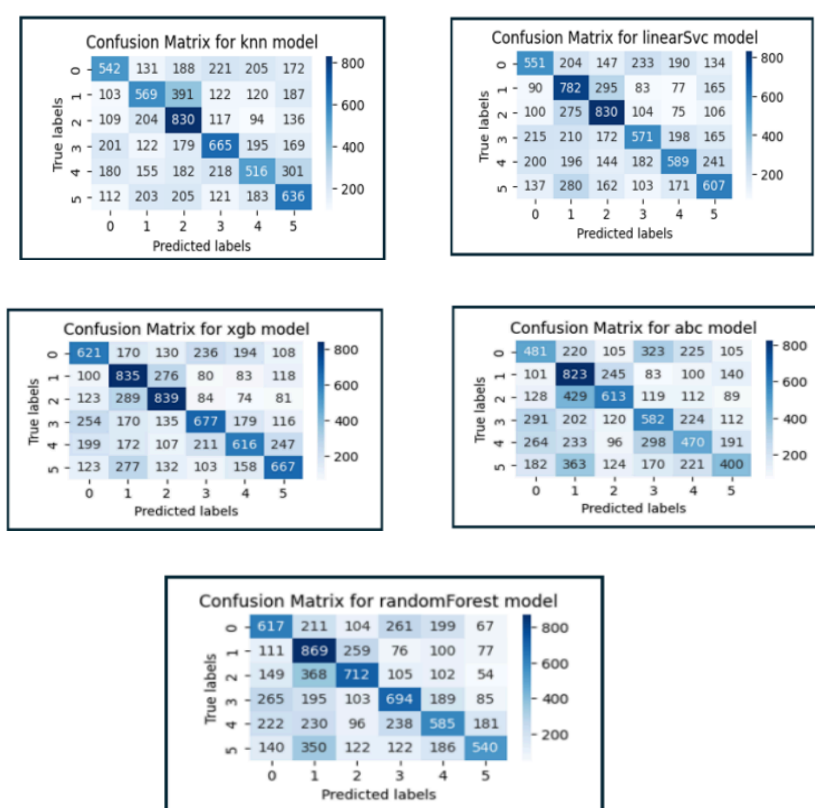
Figure 14: CountVectorizer embedding

As shown above, models trained with CountVectorizer embedding consistently achieved higher test accuracy compared to GloVe, highlighting the impact of embedding choice on model performance.

```
1123/1123 - 41s - 37ms/step - accuracy: 0.9876 - loss: 0.0235 - val_accuracy: 0.9068 - val_loss: 0.5898
281/281 ----- 4s 13ms/step - accuracy: 0.9107 - loss: 0.5376
Test Accuracy: 0.9068343639373779
```

The CNN model achieved the highest test accuracy (0.9068), demonstrating the advantage of deep learning with learned embeddings over traditional models with static embeddings.

## Confusion Matrices for Classical Models



**Figure 16:** presents the confusion matrices for the classical machine learning models.

Models like **LinearSVC** and **XGBoost** show strong diagonal patterns, indicating relatively high classification accuracy across most emotion classes. In contrast, models like **AdaBoost** display more frequent off-diagonal errors, particularly confusing **anger (3)** with **fear (4)** and **sadness (0)** with **love (2)**.

These confusions highlight difficulties in distinguishing between emotionally similar or linguistically overlapping expressions.

## Fine-Tuning BERT for Emotion Classification

To leverage recent advancements in Natural Language Processing, we performed fine-tuning on **BERT**, a powerful pre-trained transformer model, widely recognized for its superior language understanding capabilities. We randomly selected **3,000 tweets** from the original dataset and split them into **80% training** and **20% testing** subsets. Each emotion was encoded as an integer (0–5) for classification purposes. Tweets were tokenized using BERT's built-in tokenizer with a maximum length of **64 tokens**, suitable for short-form texts.

The fine-tuning process involved training the model for **3 epochs** (complete training cycles), using a batch size of **4 tweets per step** due to computational constraints, and employing standard fine tuning hyperparameters (learning rate and weight decay) recommended by HuggingFace guidelines.

The fine-tuned BERT model achieved an impressive accuracy of **89.33%** on the test dataset. close to our best-performing CNN model (**90.68%**) and significantly higher than all classical machine learning methods tested.

This result clearly demonstrates that fine-tuning transformer-based models, even briefly and on limited datasets, can significantly improve emotion classification performance, effectively capturing subtle emotional cues within tweets.

fine tuned BERT Accuracy: 89.33 %

## Summary of Main Findings

In this research, we compared some approaches for emotion detection in short-form Twitter texts, including classical machine learning models, Convolutional Neural Networks (CNN), and a transformer-based model (fine tuned BERT).

The key findings are:

- **Deep learning approaches** (CNN and BERT) significantly outperformed classical machine learning methods, demonstrating superior ability in capturing nuanced emotional expressions within informal, short texts.
- Among classical models, **LinearSVC (SVM) and XGBoost** showed relatively strong results, while models like **AdaBoost** struggled considerably.
- Using learned embeddings (CNN, BERT) proved more effective compared to static embeddings (e.g., GloVe, CountVectorizer).

The highest accuracy achieved was **90.68%** by the CNN model, followed closely by fine-tuned BERT at **89.33%**. Both deep learning models showed promising results, highlighting the benefit of leveraging advanced NLP techniques.

## Insights from Error Analysis

The confusion matrices revealed important insights into the classification errors:

- Models frequently confused emotions with linguistic overlap or similar emotional nuances, particularly:
  - **Fear and Anger** often showed confusion due to overlapping vocabulary and expressions.
  - **Sadness and Love** were occasionally misclassified, likely due to subtle contextual usage in short informal text.

Identifying these specific areas of confusion allows future research to target improvements, either by increasing the amount of training data for these classes or by applying specialized preprocessing and linguistic context analysis techniques.

## Practical Implications Of This Project

Emotion detection from tweets has significant practical implications, including:

- **Mental Health Monitoring:** Automatic identification of emotional distress in social media, potentially aiding in early intervention.
- **Customer Service Automation:** Enhanced chatbots capable of responding with greater emotional sensitivity, leading to improved customer experience.
- **Social Media Analytics:** Better understanding of public sentiment towards events, brands, or topics, useful for targeted marketing and public opinion research.

Our results, demonstrating high accuracy and clear potential for real-world application, provide a foundation for deploying emotion detection systems in these domains.

