Text Mining Project

Final Report

Emotion Recognition in Chat Messages

Submitted by :

Kunal Dulloo (24020845017)
Gauri Nagre (24020845018)
Nikita Jain (24020845019)

Introduction

Social media platforms like Reddit capture raw, unfiltered human emotions in text. Mining this data provides an opportunity to study how people express themselves and to build applications that respond with empathy. Our project focuses on **emotion recognition from Reddit comments**, culminating in a **chatbot front end** that detects emotions and generates context-aware replies.

We began by scraping Reddit using **PRAW**, collecting comments across diverse subreddits. After cleaning the dataset with **pandas, NumPy, and regex** - removing duplicates, filling missing values, and standardizing formats - we explored its structure through **Exploratory Data Analysis (EDA)**. Visualizations such as the **distribution of comment lengths**, **bar charts of top subreddits and authors**, and **before - and - after plots of missing values** revealed key patterns in the data.

Emotion Analysis: Using **transformer-based NLP models** (SamLowe/GoEmotions & Crimius/DistilRoBERTa) to classify each comment into one of 28 fine-grained emotional categories.

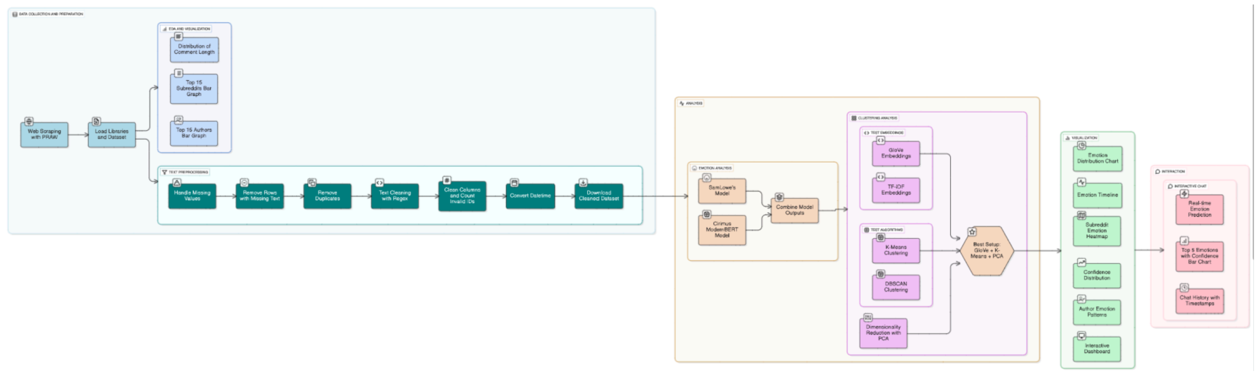
Clustering & Confidence Filtering: Grouping comments by emotional similarity using **unsupervised machine learning**, with a confidence threshold to avoid misclassification noise.

Visualization: Creating interactive dashboards to explore emotion trends, subreddit-level distributions, author behavior, and model confidence.

Interactive Chat: Allowing users to type custom text and see real-time emotion predictions, simulating an **emotion-aware chatbot** experience.

This end-to-end system can be used by **community managers**, **social scientists**, and **mental health researchers** to monitor community well-being, detect toxicity, and study

Methodology:



Data Collection

Dataset sourced from **Reddit** using **PRAW (Python Reddit API Wrapper)** – allows structured access to posts, comments, and metadata.

Reddit is chosen for **diverse discussions** and **open emotional expression**.
Comments scraped from **multiple subreddits** to capture variation in tone, topic, and emotion.

Each record included: **text, author ID, subreddit name, comment length, and timestamp**.

Preprocessing steps:

- Removed rows with missing text.
- Filled missing authors with **“Unknown”** and subreddits with **“unspecified.”**
- Removed duplicates and invalid entries.

Resulted in a **clean, reliable dataset** for further analysis.

Using **PRAW (Python Reddit API Wrapper)** with Reddit’s API, you can collect emotion-based chat data in a structured way. After creating an app in Reddit preferences, you get the required specifications: **client_id, client_secret, and user_agent**. With these, you connect to Reddit via PRAW, scrape posts from emotion-related subreddits (like *r/happy*, *r/depression*, *r/anger*), or filter by emotion keywords (e.g., *happy, sad, angry*). The collected posts (titles, bodies, comments) can then be stored in CSV/JSON format, forming a labeled dataset for **emotion recognition tasks**. This approach is more stable and structured than raw web scraping since it uses Reddit’s official API.

Exploratory Data Analysis (EDA)

After cleaning, we moved into **Exploratory Data Analysis (EDA)** to understand the structure of the dataset, identify trends, and highlight challenges specific to emotion recognition on Reddit. The goal of this phase was not only to validate preprocessing but also to surface insights that would guide embedding choices and model selection.

We began with a **distribution of comment lengths (characters and words)**. This visualization showed that most Reddit comments are relatively short, with the average around 13 words, but the distribution had a long tail reaching up to 300 words. This insight was important because shorter texts tend to be harder for models to classify into emotions due to limited context. **Boxplots of document lengths** further emphasized outliers and highlighted variance across the dataset.

Next, we examined **subreddit-level activity**. A **bar chart of the top 15 subreddits** revealed which communities contributed the largest number of comments. This analysis helped confirm dataset diversity while also surfacing potential thematic biases—for instance, some subreddits might naturally lean toward specific emotions (e.g., r/relationships toward sadness or anger). Similarly, a **bar chart of the top 15 authors** showed which users were most active, flagging risks of overrepresentation.

To ensure data quality improvements, we compared **before-and-after visualizations**. Missing values, duplicate entries, and invalid rows were plotted as stacked bar graphs to demonstrate reductions post-cleaning. A second distribution plot of comment lengths (after preprocessing) confirmed that noise reduction hadn’t distorted the overall structure of the data.

Temporal dynamics were also explored through **line graphs of word burstiness over time**. By plotting weekly raw word counts, we identified periods where certain terms spiked, hinting at emotional or topical surges tied to real-world events. This step highlighted how emotion-related words don’t just exist in isolation but fluctuate with time.

These findings are connected directly to embedding preparation, since understanding which words dominate and how they distribute in text helps ensure embeddings capture both common usage and rare but meaningful signals.

Together, these EDA steps painted a clear picture of the dataset: diverse, slightly skewed toward shorter comments, temporally dynamic, and rich in both noise and emotion-related tokens. This foundation ensured the subsequent modeling pipeline was built on a well-understood dataset rather than a black box.

Text Preprocessing

Before modeling, the raw Reddit data required systematic preprocessing to remove inconsistencies and prepare it for analysis. We first addressed **missing values**, dropping rows with no text, filling missing author names with `"Unknown"`, and substituting absent subreddit labels with `"unspecified"`. **Duplicate rows** were identified and removed to avoid bias from repeated entries.

Next, we applied **text cleaning** techniques. Regular expressions (regex) were used to strip unwanted characters, punctuation, and extra spaces, ensuring the dataset retained only meaningful text. We also standardized column formats, such as converting timestamps into consistent datetime objects and validating IDs. Finally, we saved a **cleaned version of the dataset**, which was later used for embedding and modeling.

This preprocessing stage ensured that noise was minimized, data quality was consistent, and the dataset was robust enough for embedding models and clustering.

Emotion Analysis

At the heart of our pipeline lies **emotion classification**, powered by **transformer-based NLP models**.
We utilized two state-of-the-art models:

- **SamLowe/roberta-base-go_emotions** – trained specifically on Google's GoEmotions dataset, which maps text into 28 fine-grained emotions like *joy, anger, sadness, curiosity,* etc.

- **Crimius/j-hartmann-distilroberta-base** – a smaller, efficient model providing competitive performance, used as a benchmark and for model comparison.

For each Reddit comment, the model outputs a probability score for all 28 emotions.
We selected the emotion with the highest probability and **filtered the results by a confidence threshold (0.5)**.
This approach minimizes noisy predictions — comments with low confidence are marked as **neutral**, which improves the overall interpretability of results.

Clustering & Confidence Filtering

Once each comment was labeled with an emotion, we performed **unsupervised clustering** to group comments that were emotionally and semantically similar.
Two clustering algorithms were implemented:

- **K-Means:** Produces a predefined number of clusters (k), finding centroids that minimize within-cluster variance.
- **DBSCAN:** A density-based clustering method that can discover arbitrary-shaped clusters and label outliers as noise.

We experimented with two text representation techniques:

- **TF-IDF:** Captured context-specific word importance across Reddit comments.
- **GloVe Embeddings:** Provided semantic vector representations of words based on global co-occurrence statistics.

After testing multiple combinations, we found **K-Means with GloVe** gave the most meaningful, well-separated clusters.

We also applied **PCA** for dimensionality reduction (50 components for clustering, 2 components for visualization), making the clusters easier to interpret.

Visualization

We designed an **interactive analytics dashboard** to allow deep exploration of emotional patterns.
Key visual components include:

- **Emotion Distribution:** Pie and bar charts showing how emotions are distributed across the dataset.
- **Emotion Timeline:** A line chart visualizing how emotional trends evolve over time, useful for detecting spikes in anger, sadness, or joy.
- **Subreddit Emotion Heatmap:** Compares which emotions dominate across the most active subreddits.
- **Confidence Distribution:** Visualizes how confident the model is across different emotions, helping identify categories that may need further model tuning.
- **Author Emotion Patterns:** Heatmaps showing the emotional behavior of top authors, providing insights into individual posting tendencies.

These visualizations make it possible to detect **patterns, anomalies, and emotional shifts** across communities.

Interactive Chat

To demonstrate practical applications, we built a **real-time interactive chat interface**.
Users can type any message and instantly receive:

- A list of the **top 5 predicted emotions** with confidence percentages.
- A **horizontal bar chart** ranking all emotions detected.
- A timestamped **chat history**, allowing users to revisit previous entries and track emotional tone over multiple interactions.

This feature turns the project into an **emotionally intelligent chatbot**, capable of offering personalized feedback, mood tracking, or moderation support in real time.

Key Insights

1. **Confidence Filtering Improves Reliability**
   Applying a threshold of 0.5 significantly reduced false positives, leading to a more trustworthy emotion dataset. Comments below the threshold were treated as neutral, which helped maintain data quality.

2. **K-Means + GloVe Outperforms Alternatives**
   GloVe embeddings produced sharper separation for short-form Reddit comments. DBSCAN was less suitable due to variable comment density, often labeling too many points as noise.

3. **Emotion Trends Reveal Community Behavior**
   Timeline visualizations showed clear spikes in emotions like anger and joy, indicating that our system could be used for event detection or monitoring community sentiment shifts.

4. **Subreddit-Level Differences Are Meaningful**
   Some subreddits showed dominance of curiosity and admiration, while others leaned toward disapproval and sadness. This can inform moderators about the emotional tone of their communities.

5. **Interactive Chat Has Practical Use Cases**
   The chatbot can be integrated into community tools for toxic content detection, user well-being monitoring, or even personal mood tracking applications.

6. **Potential for Deployment & Scaling**
   The pipeline can be extended to larger datasets, deployed as a web app, or integrated into moderation tools, making it relevant for industry use cases beyond research.