Generative Pretrained Transformers for Emotion Detection in a Code-Switching Setting

Andrew Nedilko
Workhuman
agnedil@gmail.com

Abstract

This paper describes the approach that we utilized to participate in the shared task for multi-label and multi-class emotion classification organized as part of WASSA 2023 at ACL 2023. The objective was to build models that can predict 11 classes of emotions, or the lack thereof (neutral class) based on code-mixed Roman Urdu and English SMS text messages. We participated in Track 2 of this task - multi-class emotion classification (MCEC). We used generative pretrained transformers, namely ChatGPT because it has a commercially available full-scale API, for the emotion detection task by leveraging the prompt engineering and zero-shot / few-shot learning methodologies based on multiple experiments on the dev set. Although this was the first time we used a GPT model for the purpose, this approach allowed us to beat our own baseline character-based XGBClassifier, as well as the baseline model trained by the organizers (bert-base-multilingual-cased). We ranked 4th and achieved the macro F1 score of 0.7038 and the accuracy of 0.7313 on the blind test set.

1 Introduction: Emotion Detection

Emotion detection using machine learning (ML) models presents significant challenges due to several factors. Firstly, emotions are subjective - the way they are expressed can vary greatly between different people and cultures.

Secondly, emotions can depend on the context. The topic and the participants of the conversation, the cultural and social context can affect the way emotions are understood. But models that interpret the context are quite a challenging problem in ML.

Thirdly, the language to express emotions can be complex and varied. There are few standards for labeling emotions in text. This variability poses difficulties in developing effective ML models for emotion detection.

Finally, high-quality labeled data for creating ML models in emotion detection is scarce. The labeling process is time-consuming and subjective, making it not easy to receive large volumes of reliable training data. This further complicates the development of effective ML models for emotion detection and is even more true for such low-resource languages as Urdu and for such subjective domains as highly colloquial SMS text messages.

Due to broad usage of artificial intelligence (AI) systems, e.g. automated phone systems, online chatbots as the first step of customer support systems, emotion detection is becoming more important. But it is still a challenging task for a machine because even humans can disagree about emotion interpretation. This makes it harder to transfer the human knowledge to machines.

2 Related Work

People often find it difficult to fathom all the subtleties associated with emotions, and therefore historically there has been only limited research done for the purpose of emotion detection and classification. But recently, emotion detection and empathy prediction have really gained popularity in various shared tasks and challenges. However, we must acknowledge that the majority of emotion research has been done in a monolingual setting. Such methods can hardly be efficient for code switching which is common in social media when two different languages can be used interchangeably in the same message.

As described by Ameer et al. (2022) and Ilyas et al. (2023) - Urdu is a South Asian language spoken by over 300 mln. speakers. Traditionally, it is written in Perso-Arabic script, but Roman Urdu (RU) is more popular for informal settings, for example on social media platforms like Facebook, Twitter, YouTube etc. RU is the language of the Internet.

Ameer et al. (2022) presented a large corpus for the multi-label emotion classification: 11,914 code-mixed SMS messages (Roman Urdu - English) -

Proceedings of the 13th Workshop on Computational Approaches to Subjectivity, Sentiment, & Social Media Analysis, pages 616–620

which serves as the dataset for the current shared task. Every message was annotated manually for the presence of 12 emotions: anger, anticipation, disgust, fear, joy, love, optimism, pessimism, sadness, surprise, trust, and neutral (no emotion). The best performing ML methods were classical machine learning models trained on word unigrams with a combination of OVR multi-label and SVC single-label ML algorithms.

Ilyas et al. (2023) generated a new code-mixed Urdu - English emotion corpus from multiple sources. They developed and applied their own Annotation Guidelines at two levels: neutral vs. emotional sentences and the choice of correct emotion labels. A CNN architecture with Glove embeddings outperformed all other ML models at the first level, while RandomForest was the best model for the second level.

Akhter et al. (2020) proposed the first offensive Urdu dataset with user-generated content from social media. Their ML experiments showed that character-based trigrams outperformed other word and character n-grams with such ML models as LogitBoost and SimpleLogistic.

As a general approach, Wang et al. (2015) utilized a term-document bipartite graph to capture the bilingual and sentimental code-switching information and proposed a label propagation-based approach to learn and predict in such a graph.

3 Dataset and Task

This work is a result of our participation in **Track MCEC: Multi-Class Emotion Classification**. Given a code-mixed SMS message, the task was to classify it into one of the above 12 categories. The dataset for this task was proposed by Ameer et al. (2022).

The training set consists of approx. 9.5k examples (see Figure 1), while there are approx. 1190 examples in each of the development and test sets (Figure 2 and Figure 3, respectively). To get a more reasonable level of bias variance trade-off in ML models, the dataset had to be deduplicated. The training set contained 2k duplicates, the dev set and test set – 40 duplicates each. Besides, 0.5k data points from the training set leaked into the development set and about the same number [of somewhat different training data points] leaked into the test set (approx. half the size of each subset). The dev and test sets could not be deduplicated or reduced in size in order to be able to report the correct metrics. Therefore, the leaked data points were removed from the training set.

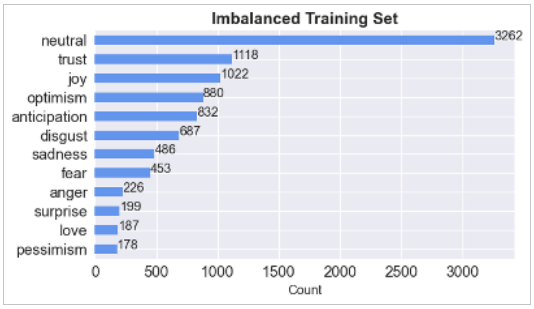

Figure 1: Distribution of labels in the training set.

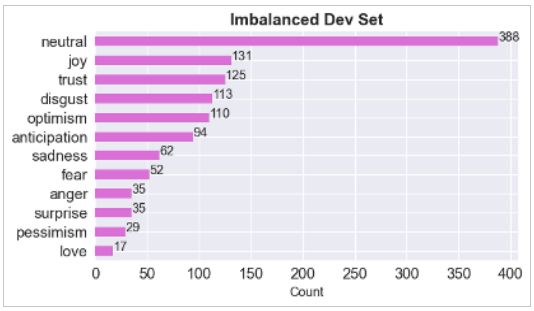

Figure 2: Distribution of labels in the dev set.

After deduplication and removal of the overlap with the dev set and test set, the training set became approx. two thirds of its original size (6.1k examples). All three subsets are imbalanced in a somewhat similar way with slight variations; the majority category is neutral, and the least represented ones are pessimism and love.

4 System Description

4.1 Baseline Models

Our baseline model was XGBClassifier with character ngram counts as features. The ngram range was (1,5). We did limited hyperparameter tuning and cross-validation (the main focus of this study was a different model). The initial macro F1 score

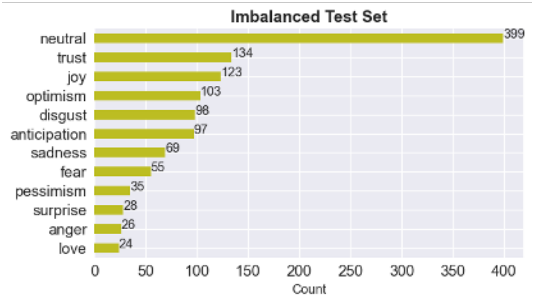

Figure 3: Distribution of labels in the test set.

was below 0.6, while the best one was 0.68 with the accuracy of 0.705.

4.2 GPT and Iterative Prompt Engineering

Large transformer models and their ensembles (Desai et al., 2022) demonstrated good results on sequence classification, but it is still hard to achieve a high macro F1 score when detecting emotions.

Given the impressive recent advances in autoregressive models with the generative pretrained transformer (GPT) architecture, we decided to use GPT models with the zero-shot or few-shot approaches to capture the human emotions in a more precise manner.

We used ChatGPT for this purpose because it has a commercially available full-scale API, and it was trained on multiple languages and understands them inherently under the hood. Several dozens of prompts were engineered to run full-scale experiments on the dev set. The training set was used only for few-shot learning.

First, ChatGPT was asked to translate texts into English and then do the classification. But the final winning prompt did not mention the translation and asked the model to do the classification directly. It was a few-shot approach where, for each data point in the dev set, we selected the top 100 most similar texts from the training set as examples. We used the cosine similarity on OpenAI embeddings as the similarity measure between dev set and training set texts.

This was the first time we used ChatGPT for emotion detection on a code-mixed dataset with such a colloquial vocabulary as SMS text messages. Nevertheless, it outperformed our baseline classifier and the baseline model trained by the organizers of the shared task (bert-base-multilingual-cased). We ranked 4th and achieved the macro F1 score of 0.7038 and the accuracy of 0.7313 on the final test set.

5 Analysis of Results

The initial idea was that LLMs like ChatGPT are perfect for the task at hand because they are known for being able to translate from one language to another and they can do zero-shot text classification, especially with such self-explanatory labels as joy, fear, anger, surprise, etc.

Our initial approach was a zero-shot multiclass emotion classification with a prompt asking the LLM to translate and classify each given text from the dev set into one of the emotion categories. We then calculated the macro F1 score as if it was a usual ML classification task. Extracting emotion categories from the LLM's responses was not always straightforward, as the LLM tended to include extraneous words. We tried to iteratively improve the prompts to achieve better F1 scores.

The quality of the response from ChatGPT greatly depends on the prompt. For example, the first step was to do the translation, and when we asked ChatGPT to translate code-mixed texts into English, it did a good job silently. But when we decided to improve the results and asked ChatGPT: "Act as a smart Roman Urdu to English translator and do your best to translate the text below into English", unexpectedly the LLM started to complain saying that the quality of the Urdu text was not very good, and that it couldn't translate the text well. Since we had to batch process about a thousand of such responses in the dev set, filtering the cases when the English translation was provided vs. the cases when there was a complaint was not a trivial task.

The macro F1 score for the initial zero-shot classification results was below 0.5. This could be due to the fact that there were 12 labels and some of the emotions were really hard to extract from the available text without the knowledge of the full context which was not always provided explicitly.

Next, we tried a zero-shot binary text classification (neutral vs. emotional texts). ChatGPT was biased towards the neutral category, but when we tried to use this prompt: "Act as a careful and accurate zero-shot text classifier and classify the text neutral or emotional . . . ", the LLM started to complain again saying it couldn't produce an accurate classification because of the colloquial nature of the text or for other reasons. It seemed like ChatGPT had a problem of excessive perfectionism when one asked it to be very good at what it does!

The binary classifier's macro F1 score was around 0.5. Therefore, next we decided to do the few-shot multiclass classification. ChatGPT has a limited context window size of 4096 tokens while the total number of tokens in the training set is over 110k. At first, we randomly split the training set into chunks of 100 texts each because this allowed to keep the total number of tokens in the concatenated examples under 4k. We tried to keep the proportional distribution of labels in each chunk in line with the overall distribution. Then, we iter-

atively used these chunks with all the data points from the dev set: one chunk of examples per one dev set data point.

Afterwards, we tried a smarter approach which allowed us to get our best macro F1 score on the test set of 0.7038 and the accuracy of 0.7313. First, we embedded every SMS text message using the OpenAI embeddings. Then, using the cosine similarity, for each example in the dev set (or in the test set for the final submission) we selected 100 most similar examples from the training set and used those as few-shot examples to teach ChatGPT about existing emotion labels. See Table 1 below for a brief summary of results.

Classifier	**Macro F1**	**Accuracy**
Baseline XGBClassifier	0.68	0.705
Baseline BERT-base	0.7014	0.7298
Best few-shot ChatGPT system (most similar examples)	0.7038	0.7313

Table 1: Summary of Emotion Classification Results

Other approaches included sending the second prompt asking the LLM if it was sure of its previous response. There were quite a few cases when ChatGPT changed its response which helped with the zero-shot classification, but not with the few-shot effort.

We also tried concatenating all examples and their categories into one huge prompt or having each individual example in a single prompt using the chat-like framework offered by the ChatGPT API. This did not have any significant impact on the results.

6 Conclusions

Overall, we should say that this task is quite challenging because:

- the data consists of code-mixed text messages - two languages mixed together;
- Urdu is a low-resource language; during training ChatGPT saw significantly less Urdu than English - it must be considerably less efficient at Urdu;
- the emotion detection task is challenging for machines in general;
- there are 12 emotion categories which is a lot;
- even for a human, the SMS text messages often don't have enough context to understand the type of emotion.

Nevertheless, our approach helped us beat both baseline models. Based on a subjective assessment, the time spent for prompt engineering, as opposed to the wait time when the code was just running, was less than the time we would normally spend on ML model selection, hyperparameter tuning, and cross-validation to avoid overfitting when dealing with classical and deep learning ML models.

Based on our limited experiments in the web UI, we anticipate that using GPT-4 will help achieve even better results once the GPT-4 API becomes available to the general public. It was not available at the time we ran our experiments for this task.

Limitations

In addition to the data-based limitations listed in Section 6, we faced the following technology-based limitations:

- The results are greatly dependent on the prompt design. It is not uncommon to spend a lot of efforts on coming up with the right prompt. Each use case may need a different prompt.
- Overall, ChatGPT provides a stable output especially if one asks for a specific output format. But there is still an element of volatility when one or few responses contain extraneous text, or the categories are outside of the predefined list. This is due to the conversational nature of the model, and such cases had to be processed as exceptions.
- ChatGPT "remembers" the past conversations, but this memory is limited to the context window size which is only 4096 tokens. This makes it challenging to work with large datasets which have to be split into pieces to be processed independently.
- One has to remember that one must pay for the use of the ChatGPT API - very long prompts used multiple times or too many examples for few-shot learning may be discouraged for cost savings purposes.

References

Muhammad Pervez Akhter, Zheng Jiangbin, Irfan Raza Naqvi, Mohammed Abdelmajeed, and Muhammad Tariq Sadiq. 2020. Automatic detection of offensive language for urdu and roman urdu. *IEEE Access*, 8:91213–91226.

Iqra Ameer, Grigori Sidorov, Helena Gomez-Adorno, and Rao Muhammad Adeel Nawab. 2022. Multi-label emotion classification on code-mixed text: Data and methods. *IEEE Access*, 10:8779–8789.

Shaily Desai, Atharva Kshirsagar, Aditi Sidnerlikar, Nikhil Khodake, and Manisha Marathe. 2022. Leveraging emotion-specific features to improve transformer performance for emotion classification. In *Proceedings of the 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis*, pages 245–249, Dublin, Ireland. Association for Computational Linguistics.

Abdullah Ilyas, Khurram Shahzad, and Muhammad Kamran Malik. 2023. Emotion detection in code-mixed roman urdu - english text. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 22(2).

Zhongqing Wang, Sophia Lee, Shoushan Li, and Guodong Zhou. 2015. Emotion detection in code-switching texts via bilingual and sentimental information. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 763–768, Beijing, China. Association for Computational Linguistics.