Report by Inga Grigoryan (2110098) and Armen Grigoryan (2109932)

Emojional intelligence: Meme Generation Model

Express your inner chaos in PNG

Abstract

Technology's integration in everyday life has substantially reshaped the way communication and entertainment are perceived by humans. These aspects of life have evolved from in-person verbal exchanges to digital formats – texts, stickers, GIFs, and memes: the powerful tools to help us convey emotion, humor and nuance. To that end, this project aims to detect the emotional tone of user-provided text, and generate a corresponding meme that reflects that tone. A logistic regression classifier is trained on TF-IDF features, extracted from an emotion labelled database(1), to recognize six core emotions: joy, sadness, anger, fear, love, and surprise. Once the emotion and the corresponding fit scores are determined, the program selects an appropriate meme template, along with the caption, delivering the generated image with the help of memegen.link API(2).

1. Introduction

The goal of Emojional Intelligence is twofold:

1. Detect the emotional tone of the sentence provided by the user, thereby assigning it an appropriate emotion category,
2. Generate a corresponding meme that visually and textually expresses the detected emotion, providing an entertaining and share-worthy image.

To accomplish this, the project is divided into 3 core components:

- *Model trainer*: this part pre-processes the text, using **TFI-ID Vectorizer** from the **scikit-learn** library to train the model.
- *Meme Generator*: the actual meme generation part! The tool used in this section is memegen.link as a meme API. The pre-existing meme images are taken from here and matched with one of the texts we have provided for each of the six emotions.
- *Web application*: the integration layer for all components. The back end is implemented using Python's Flask microframework, while the front end is built with HTML, CSS, and JavaScript, to provide a simple GUI to display the results.

2. Method

2.1 Model Trainer

This section aims to perform the actual training process for the model. In order to achieve this, a number of steps have been implemented.

2.1.1 The dataset. First and foremost, an appropriate **dataset,** which can be found on Kaggle(1), was chosen for training the model; it is noteworthy that additional components have been added to the preexisting data, in order to improve the effectiveness and correctness of the model. Several emotions were severely underrepresented in the dataset, resulting in inaccurate extraction of emotions. The dataset includes the following components:

- train.txt (approximately 19000 entries),
- test.txt,
- val.txt,

the elements of which are all represented in the <sentence;label> format, where label is one of the six possible emotions: **joy, sadness, anger, fear, love, and surprise**. Each file is read into a two-column table of sentence and label, after which the training and validation files are combined to form a larger set. The test file remains untouched until the final performance reporting, to ensure unbiased estimation.

2.1.2 The Model

Following the loading procedures, we now convert the raw sentences into **features** suitable for learning. Pipelining from scikitlearn is used to achieve the goal of employing TF-IDF vectorizer to tokenize the sentences, which are then fed to the multinomial logistic regression classifier.

In order to avoid misspellings, punctuation issues, repeated characters, and nonsensical words, TF-IDF is configured in the following way:

- the maximum feature count is set to 10000, which reduces dimension, and the chances of overgeneralizations,
- logarithmic TF scaling is applied to term frequency, so that the extremely common terms do not overshadow the others,

- terms that appear in less than 2 documents are considered irrelevant and are discarded,
- finally, N-gram range is set to (1,3) in order to allow for short expressions, such as don't like, very happy, etc., to be captured.

Despite the fact that TF-IDF has its limitations, such as lack of context awareness, vocabulary coverage (e.g. slang), and sarcasm detection, it remains an appropriate choice for the purpose of the project.

Afterwards, the arguments of the logistic regression classifier are configured the following way:

- max_iter is set to 1000, giving the optimizer the room for up to a thousand passes over the training set,
- then, the class_weight is set to balanced in order to avoid misclassifications of more rare emotions, thereby providing balance and preventing the majority classes from overshadowing the others.

As a result, we are left with a model that is expressive, since it can respond to longer phrases, rather than just single words, and one that is fair as it is capable of recognizing underrepresented, minority emotions. The best performing model is then found based on the maximized macro-F1 value, and its performance is evaluated on the initially untouched testing set.

Finally, all of this is stored in a separate file for further use. This specific structure was chosen in order to keep the flow of the project as clean as possible, provide a systematic approach and make further developments easier to implement.

2.2 Meme Generator

At this step, our pre-trained model is loaded. This is where the magic happens! The user input is received, one of the six emotions is predicted, along with their associated probabilities. Then, a random meme is selected and bound with one of the provided texts from the pool of available captions. The services of memegen.link API are used to generate the final product.

The main limitation of this approach is the possibility of mismatches while generating the memes. Since the choice of the captions and the pictures is randomized, some pairings may not make a lot of sense. However, there are a few measures that have been implemented to ensure the memes are as relevant as possible. First, both the captions and the images are organized according to the relevant emotions, so that no matter what pair is generated, we still end up with a somewhat applicable result. Second, the user is given a range of memes, rather than a single one. This way, they can pick and choose the one they like most, and share it with their friends!

2.3 Web Application (GUI)

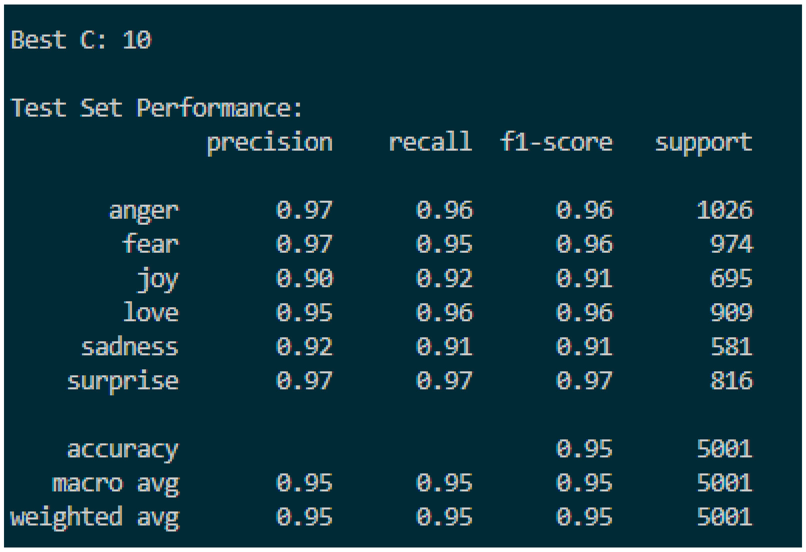
The GUI was developed with HTML, CSS and JavaScript. It is a simple, user-friendly interface, where all one needs to do is share their thoughts, click the "Generate" button, and enjoy the results. The output is not limited to just a single picture; instead, the program generates 5 different memes, allowing the user to pick their favorite.

3. Results

The analysis of the results, just like the definition of the aims for the project, has 2 layers:

- What can be concluded regarding the efficiency of the model?
- What can be said regarding the final generated product?

First and foremost, study of the results from the model reveals the pattern observed in Figure 1.1.



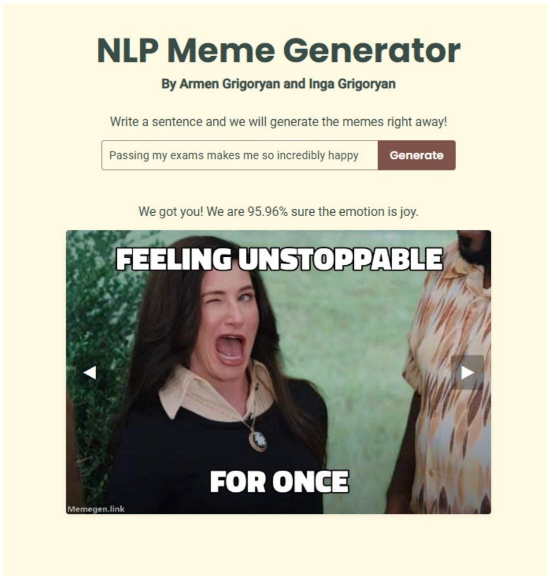
```
Best C: 10

Test Set Performance:
              precision    recall  f1-score   support

       anger       0.97      0.96      0.96      1026
        fear       0.97      0.95      0.96       974
         joy       0.90      0.92      0.91       695
        love       0.95      0.96      0.96       909
     sadness       0.92      0.91      0.91       581
    surprise       0.97      0.97      0.97       816

    accuracy                           0.95      5001
   macro avg       0.95      0.95      0.95      5001
weighted avg       0.95      0.95      0.95      5001
```

Figure 1.1

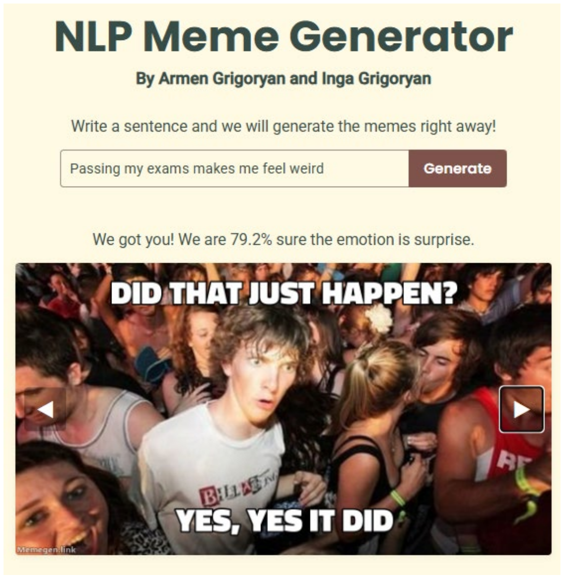
The main takeaways are:

1. The best value for C is 10. This means that a moderately strong regularization ensures optimality, hinting at the fact that the model benefits from fitting the margins tighter than the default (C=1), but at the same time makes sure that it does not overfit.
2. The overall accuracy is 0.95. This means that out of the 5001 examples, only approximately 250 are mis-classified. At the same time, the numerical result for F1 for both macro and weighted average is found to be the exact same, revealing that minority-class emotions are not overpowered by the majority ones.
3. Fear, anger, and surprise are the clear winners when it comes to precision, with a precision score of 0.97, closely followed by love, indicating that there are very few false positives when it comes to these emotions. On the other hand, 10% of the time a sentence is classified as "joy", when it should in fact be something else.
4. F1 = 0.91 for joy and sadness signal the only notable weaknesses, however these indicators are still quite strong.

Then, looking at the actual generated memes, we can confidently say that the program performs very well in some scenarios, even ones including long and complex sentences, while it is found lacking in some simpler, yet ambiguous situations. Figures 2.1 and 2.2 show good examples of a prompts that work quite well.


NLP Meme Generator
By Armen Grigoryan and Inga Grigoryan
Write a sentence and we will generate the memes right away!
Passing my exams makes me so incredibly happy
Generate
FEELING UNSTOPPABLE
FOR ONCE
Memegen.link

We got you! We are 95.96% sure the emotion is joy.


NLP Meme Generator
By Armen Grigoryan and Inga Grigoryan
Write a sentence and we will generate the memes right away!
Passing my exams makes me feel weird
Generate
DID THAT JUST HAPPEN?
YES, YES IT DID
Memegen.link

We got you! We are 79.2% sure the emotion is surprise.

Figure 2. 1 very accurate tone detection

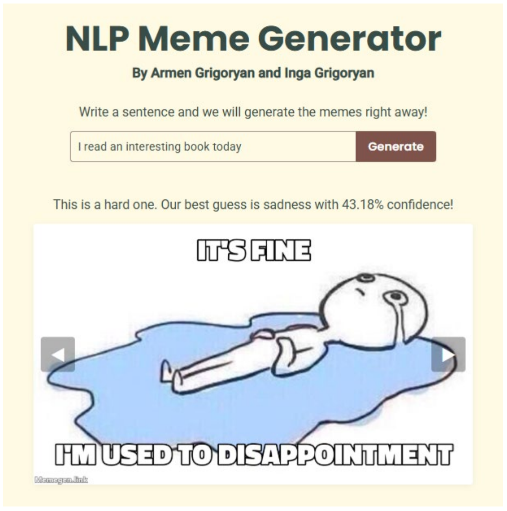
Figure 2.2 ambiguous yet accurate detection

It is worth mentioning that the prompt in Figure 2.2 could have been interpreted in a number of ways, since it is not a trivial task to understand what exactly the word "weird" represents in this context.

Nonetheless, the model manages to find a solution that seems the most reasonable out of all the possibilities.

On the other hand, Figure 2.3 shows a scenario with a less accurate result.

Figure 2.3 inaccurate result


NLP Meme Generator
By Armen Grigoryan and Inga Grigoryan
Write a sentence and we will generate the memes right away!
I read an interesting book today
Generate
IT'S FINE
I'M USED TO DISAPPOINTMENT
Memegen.link

This is a hard one. Our best guess is sadness with 43.18% confidence!

The prompt "I read an interesting book today" does not particularly have any emotional undertones that the model can recognize. "Interesting" can have the meaning of scary, fun, entertaining, enlightening, and thousands of other things! The model is not familiar with these concepts, which is why the result might prove to be somewhat inaccurate for some users.

Overall, the generated pictures can be said to accomplish the goal of the project quite well. The ability to pick and choose the best option as described in the previous section, in turn, makes the program even more enjoyable for the user, since it is not as restrictive, and the chances of satisfying the user's needs are 5 times higher this way.

4. Future improvements

Although the model performs quite well considering the tools that have been implemented in the scope of the project, a number of things can still be improved in the future. As highlighted before, it is important to consider the fact that TF-IDF cannot exactly deal with **sarcastic** tones or recognize **irony**. Transformer based models, such as **BERT**, could be used to tackle the issue, which would make the model much smarter in tone recognition.

Second, additional functionalities could be added to match the emotions to the **real time** facial expressions of the user. **VisageSDK**(3) is a very suitable software for this purpose, as it provides tools to not only capture the emotions of the user through a live camera, but also their gender and age, which can significantly impact the types of expected outputs. For example, younger generations would enjoy memes and quotes from their favorite TV shows, while the older generations might enjoy old school dad jokes much more!

Finally, a feature that can prove to be extremely relevant would be addicting a **“feedback station”**. Here, the user would be able to assess the accuracy and relevance of the provided memes, thereby feeding new, real time data to the model. This can substantially improve performance, since each user would be able to tailor the program to their specific needs and wants.

References

1. Praveen. “Emotions Dataset for NLP.” *Kaggle*, 16 Apr. 2020, www.kaggle.com/datasets/praveengovi/emotions-dataset-for-nlp?resource=download.
2. Browning , Jace. “Custom Template.” *Memegen*, memecomplete.com/custom/?standard=true., 2025.
3. Technologies, Visage. “Visage: SDKTM - Face Tracking, Analysis & Recognition.” *Visage Technologies*, 12 Aug. 2022, visagetechnologies.com/visage-sdk/.