Artificial Intelligence Fundamentals

Project of emotion classifier based on text and image data

Weronika Piotrowska, Marcin Wojnarowski, Mikołaj Stańczyk, Patryk Prusak

Abstract. Implementation of emotion detection/classification based on text and images separately using statistical and machine learning models. Together with a discussion and comparison to existing solutions, results are presented.

KEY WORDS

1. Emotion Detection 2. Machine Learning 3. Artificial Intelligence

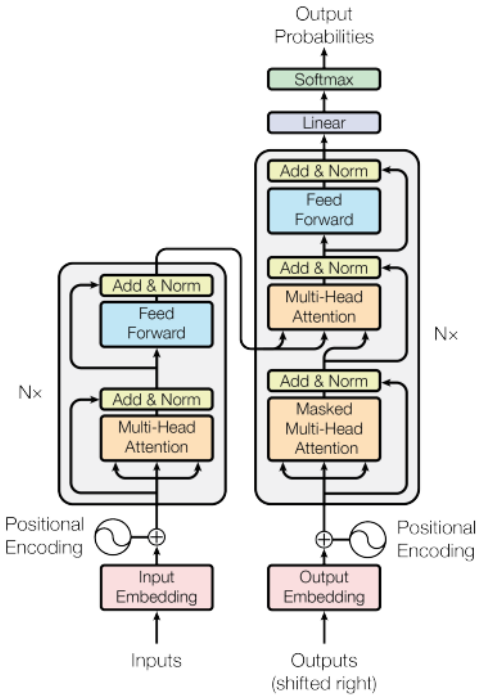

Figure 1: The Transformer - model architecture.

Contents

1. Introduction

There are many approaches to automatic emotion recognition. Even though a significant progress has been made in the recent years, human emotions still remain complex, ambiguous and thus incomprehensible for the machines. The goal of creating a neural network capable of recognizing human emotions based on speech, text or facial expression is still a challenging task. Our approach to this problem will be not only to use photographic images, but also short text, such as Tweets.

2. Problem Analysis

Text based emotion detection remains a complicated task due to its strongly reliance on Natural Language Processing (NLP) which is known to be hard to generalize. Human speech (or in this case writing) might seem structured and therefore easy to study, but that unfortunately is far from the truth. We humans rely on intuition which is notoriously hard to reason about for machines trying to abstract these notions. NLP has many approaches to it, but two stand out:

1. Probabilistic statistical models – these base their reasoning purely only the previously analyzed data and provide prediction based on the most probable outcome. These generalize poorly, but can provide very solid results for a contained goal.
2. Deep learning models – with the recent boom in deep learning, it of course has influenced NLP as well. Popular models such as BERT or GPT-3 belong to the largest deep learning models ever which shows the great complexity of the task at hand. These generalize much better, but are much more computationally expensive.[1]

The problem of computer-assisted emotion recognition based on images has been moderately studied, resulting in many attempts, some of them with decent results. This fact presents us with many approaches we could use and hopefully improve. The basic process of finding an emotion from a picture goes as follows:

Fig. 1. Example process of image emotion detection

Despite it's popularity, this area has a big potential for new inventions and applications, which we are happy to explore in this project.

3. Applications

3.1. Image Emotion Recognition—It is commonly thought that each of us is rather skilled at recognizing each others emotions, however, that is often not the case. Even trained individuals, like therapists, find it difficult to decipher patient's mood and their conclusions are often way off the mark. And so emotion recognition software could be a great asset to professionals willing to better understand their patients and improve the effectiveness of therapy. Such product has the potential to gain on popularity in the coming years because the

area of mental health undergoes a small revolution, in a sense that it is being digitalized.

3.2. Text Emotion Recognition—Text emotion detection has many applications including automatic moderation. Large companies might want to filter out comments which talk about the company using negative sentiments, or filter in those with positive sentiments. Other than filtering, it can give a very quick feedback regarding a matter. For example *The Body Shop* uses a sentiment analysis tool *Brandwatch* to gather the general reception of a campaign.*

If both image recognition and text recognition will produce satisfying results, we can combine the two. For example, one can base the user's sentiment not only on the Tweet they wrote, but also on the picture attached. Furthermore, it can be broaden to using videos as the input data, compiling both images and text. However, this solution would require additional algorithms and computing power, in order to extract image frames from the video and text from speech data.

4. Existing Solutions

4.1. GoEmotions—Emotion detection from text has been tackled multiple times before. One notable project is the GoEmotions by Google Research. It created a multi-classification model based on the BERT model for reddit comments. As the authors have described, getting clean, representative data was a challenge.[1] Reddit has a known demographic bias leaning towards young male users, which skews the data towards a toxic, offensive language. This is aligned with problems which we observed, that is, a high quality representative NLP dataset is hard to obtain. GoEmotions focused on 28 different emotions (Fig. 3) and assigned to a given text a probable combination.

Positive		Negative		Ambiguous
admiration 👋	joy 😃	anger 😡	grief 😢	confusion 😕
amusement 😂	love ❤️	annoyance 😒	nervousness 😬	curiosity 🤔
approval 👍	optimism 🤞	disappointment	remorse 😔	realization 💡
caring 🤗	pride 😌	disapproval 👎	sadness 😞	surprise 😲
desire 😍	relief 😅	disgust 🤮		
excitement 🤩		embarrassment 😳		
gratitude 🙏		fear 😨		

Fig. 2. GoEmotions 28 emotion classes

4.2. Twitter-roBERTa-base for Emotion Recognition—This solution, based on roBERTa model‡ using TweetEval benchmark§ distinguishes between four emotions: joy, optimism, anger and sadness. As the output, it gives the user probability with which the model predicted

* https://www.brandwatch.com/case-studies/the-body-shop/view/?utm_source=google&utm_medium=cpc&utm_campaign=sentiment_analysis

‡ https://doi.org/10.48550/arxiv.1907.11692

§ https://doi.org/10.48550/arxiv.2010.12421

certain emotions.[†]

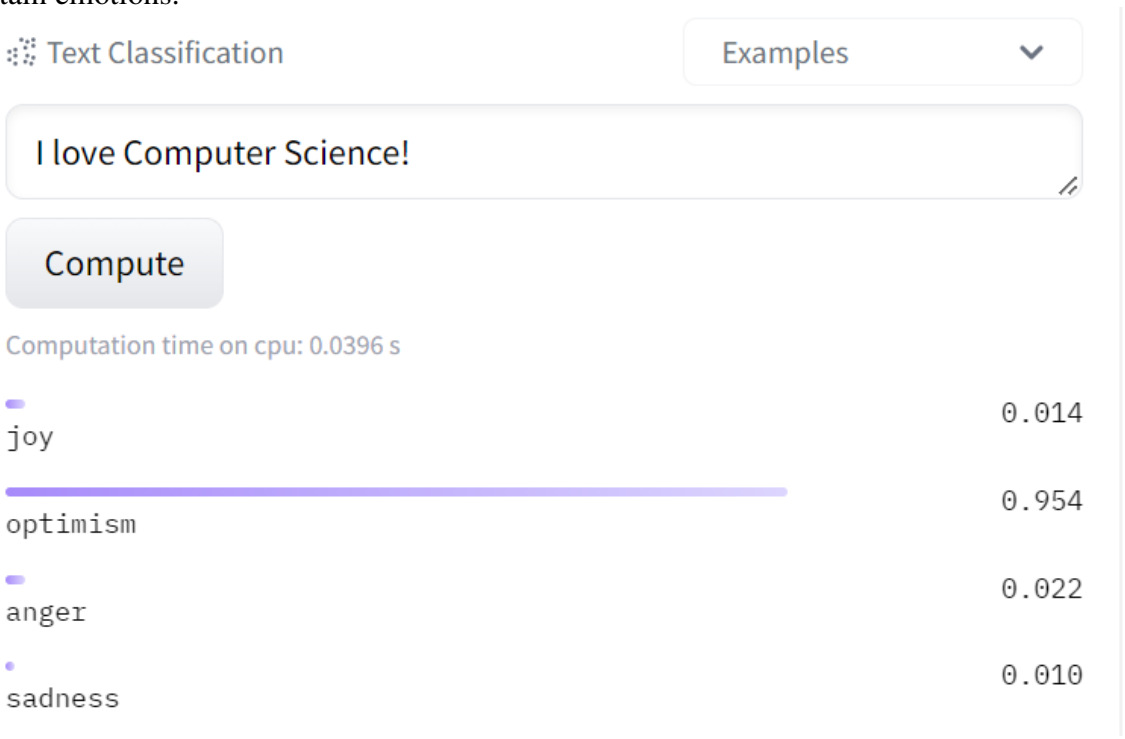

Fig. 3. Sample output from the model

4.3. Multi-class Emotion Classification for Short Texts—The approach presented in this solution uses LSTM and CNN models, classifying int five classes: neutral, happy, sad, anger, hate. The authors achieved 62.29% accuracy.[†]

4.4. Meta-learning—The authors have used CMU MULTI-PIE dataset which contains images with different head-poses, occlusions and illumination levels. The used method is called ERMOPI(Emotion Recognition using Meta-learning across Occlusion, Pose and Illumination) that detects emotion from faces using meta-learning approach and works well with partial occlusions, different head poses and illumination levels. The key benefit of this approach is the decreased need for training samples compared to existing solutions to emotion recognition and at the same time, very reliable results. This approach achieved 90% accuracy for CMU Multi-PIE images and 68% for AffectNet images.[‡]

4.5. The Face Emotion Recognizer—This is an open-source Python library built and maintained by Justin Shenk. It uses a convolutional neural network with weights contained in the HDF5 file. This project offers the possibility to use MTCNN (multi cascade convolutional network) to detect faces or the default OpenCV Haarcascade classifier. It detects emotions in six categories: fear, neutral, happy, sad, anger and disgust.[*]

4.6. Deep-Emotion—This paper presents an deep learning approach based on attentional convolutional networks in order to tackle more difficult images such as datasets with partial

[†] https://huggingface.co/cardiffnlp/twitter-roberta-base-emotion
[†] https://tlkh.github.io/text-emotion-classification/
[‡] https://www.sciencedirect.com/science/article/pii/S1319157821001452
[*] https://towardsdatascience.com/the-ultimate-guide-to-emotion-recognition-from-facial-expressions-using-python-64e58d4324ff

faces. The model is able to focus on crucial parts of the face to detect emotions. It achieves significant improvement over previous models in datasets such as FER-2013 or FERG. This method also shows that different emotions are sensitive to different parts of human face.[†]

5. Chosen Solution

5.1. Datasets—For the emotion recognition based on images, we have chosen the FER-2013 dataset[†] (Fig. 4). Facial Expression Recognition 2013 contains 35,887 facial gray-scale images (pre-split into 80/20 train/test set) of different expressions with size restricted to 48×48. Main labels assigned to pictures can be divided into 7 categories:

1. Angry
2. Disgust
3. Fear
4. Happy
5. Sad
6. Surprise
7. Neutral

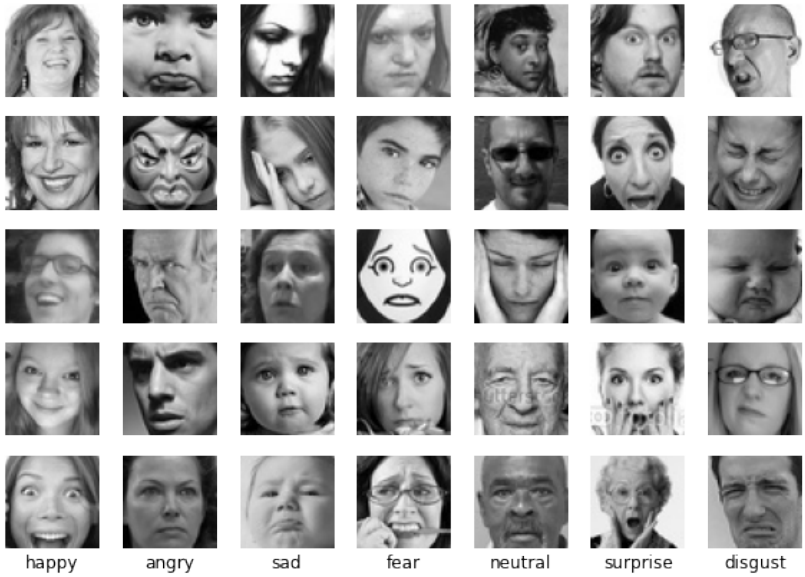

Fig. 4. A sample of the FER2013 dataset grouped by their respective emotions

We can quickly notice a few things. Firstly, the faces seem nicely centered though even if they weren't CNNs inherently are translate-invariant so it would not be a problem. Secondly, the training and testing set does not seem to differ substantially.

On the other hand, many issues can be identified:

1. Many of the pictures are stock pictures where the emotion is grossly exaggerated. This may cause the model to fail to learn the subtleties of how an emotion is expressed.
2. Different scales/rotations. Kernels in a CNN are not scale/rotation-invariant and will fail to generalize for different sizes/rotations of studied objects (lips, eyebrows, etc).

[†] https://mdpi-res.com/d_attachment/sensors/sensors-21-03046/article_deploy/sensors-21-03046.pdf?version=1619511102

[†] https://www.kaggle.com/msambare/fer2013

Layers such as maxpooling are said to be helping here [‡]

3. Different lightning conditions.
4. Mixture of real pictures with computer drawn.
5. All previous arguments are technical limitations which can be overcome. However, one last important problem which can be noticed is that some of these emotion labels are very debatable. Me (as a fellow human) would fail to classify some of those to the associated label.

With all of that in mind, and our expectations being managed we proceeded to prepare our model.

Each label has about 4,000 samples each besides *disgust* that has about 600 images (Fig. 5).

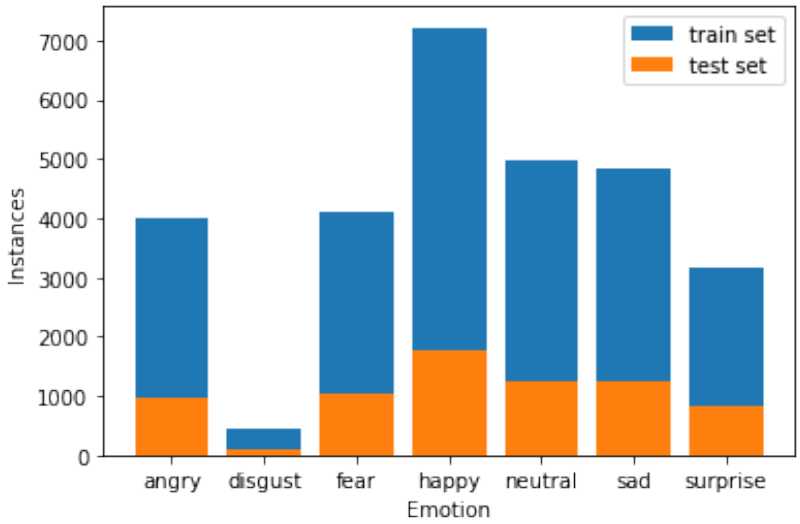

Fig. 5. Distribution of classes in the FER-2013 dataset.

Similarly, we have chosen a text dataset to portray similar emotions. We are using *Predict emotion from textual data : Multi-class text classification* published on Kaggle.[§] The dataset contains 40,000 short text messages scrapped from Twitter. Each of them were labeled by human employees, distinguishing the following emotions as seen on Fig. 6.

[‡] Deep Learning. Ian Goodfellow and Yoshua Bengio and Aaron Courville – https://www.deeplearningbook.org

[§] https://www.kaggle.com/datasets/pashupatigupta/emotion-detection-from-text

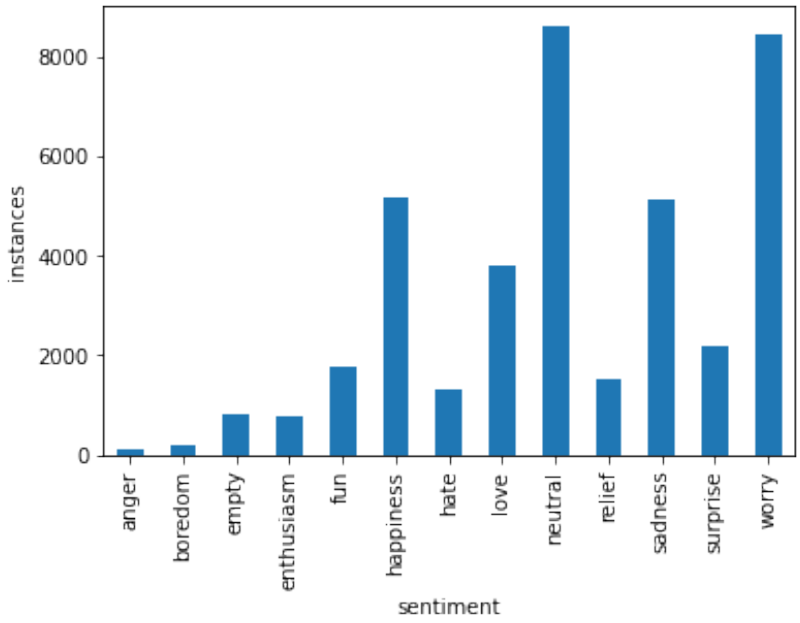

Fig. 6. Amount of instances per class

One can immediately notice that the proposed dataset is very imbalanced - this, we have decided to merge similar classes (e.g. happiness and fun) and drop classes that are not useful. The final processed dataset we will be working on looks as follows:

1. Happiness
2. Love
3. Neutral
4. Sadness
5. Worry

We have chosen

5.2. Libraries—To recognise emotions from images following libraries will be used:

- NumPy - for vectorized operations on arrays
- keras - software library providing Python interface for artificial neural networks
- tensorflow - Google's machine learning framework. Both the python and JS versions will be used

To recognise emotions from text following libraries will be used:

- NumPy - for low level vectorized arrays
- pandas - used for data frames manipulation and analysis
- nltk - library which implements models for text analysis

5.3. Models—For image emotion detection we converged onto a deep but classical Convolutional Neural Network with strong regularization. As seen on Fig. 7 the final model consists of interleaved convolutional layers with max pooling layers. Every block contains a batch normalizer to keep the computations numerically stable as well as a 25% dropout layer for regularization. Occasionally an L2 regularizer is applied to some layers to further ensure regularization. Finally, all layers use the rectified linear unit as an activation function to introduce non-linearity, this particular choice is from the simple fact that it seems to be

the industry standard. After spatial feature extraction with convolutional layers is done, the output is flattened and passed to a 3-layered dense fully-connected network which outputs the final class prediction as a softmax probability distribution. The goal function to minimize will be the categorical cross entropy function, and the optimizer used is the stateful Adam optimizer.

As an additional regularizer, training images are randomly rotated, scaled, and flipped horizontally to introduce noise.

For the text classification, we will use nltk's pre-trained Naive Bayes Classifier, which is based on naive Bayes algorithm. The algorithm makes 'naive' assumption that all labels are independent, so the following equation holds:

$$P(label|features) = \frac{P(label) \cdot P(f_1|label) \cdot \ldots \cdot P(f_n|label)}{P(features)}$$

The algorithm computes probability for each label and returns the label that fits the given test the best.

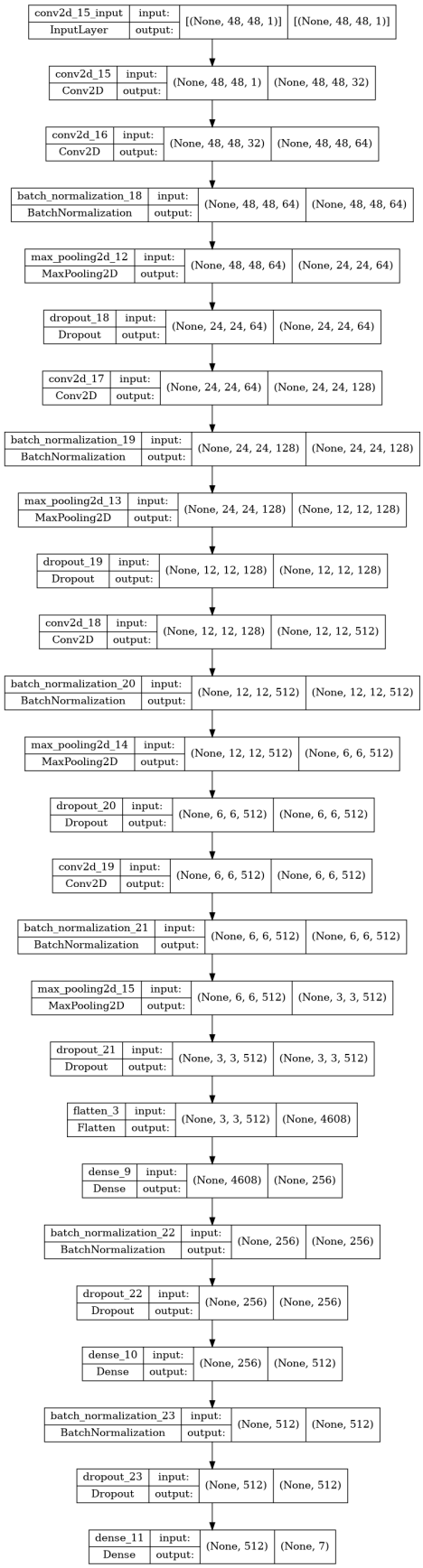

Fig. 7. Visual representation of the final image emotion detection model

6. Results

The 4.5 million parameter image model was trained for about 3 hours in Google Colab using their free GPU resources. A total of 200 epochs were completed with constant validation set checks. Final results are presented bellow

- Train set – loss: 0.6216, accuracy: 80.03%
- Test set – loss: 1.0202, accuracy: 66.98%

Notice, that a result of 67% for emotion recognition is not a bad result: after all, emotions are ambiguous and often hard to describe with just a single term. Perhaps multi-classification would prove to be better for such a task. Additionally, emotions are far from being linearly different. That is, sadness and neutrality do not differ greatly when expressed facially. On the other hand, happiness and angriness are much more distinct. Classes have different distances between each other.

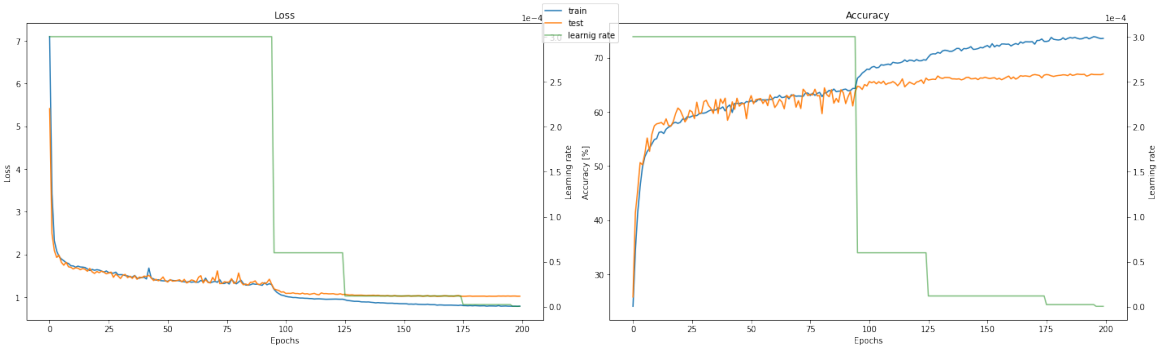

Fig. 8. Training history for the train and validation set. Two measures: loss (left), accuracy (right).

In Fig. 10 we can immediately notice a few interesting things: the model learns to classify images effectively within just a few initial epochs. Afterwards the improvement slows down dramatically. After the first learning rate decrease we can notice a sudden improvement in both sets. While initially the loss does not diverge in a significant matter between the sets, at around epoch 125 (when the second learning rate drop is applied) we can notice that the test set (here used as a validation set) no longer reports improvements and the model is clearly starting to overfit the dataset.

The learning rate reductions clearly show the model reaching a more stable performance. Ideally, training should be stopped either right after the first LR drop, or right before the second one.

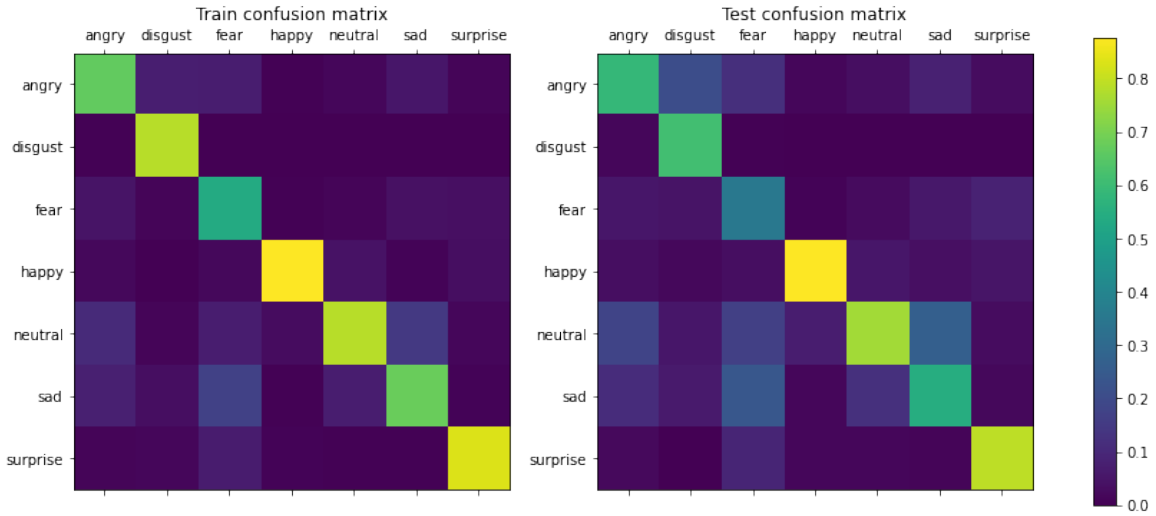

Fig. 9. Confusion matrix for the final image model. Separately for the training set (left) and testing set (right).

Finally, the confusion matrix is shown (Fig. 9). X-axis depicts the label and the y-axis plots the distribution of the model's predictions. As expected the brightest parts are on the diagonal (true positives). Train's and test's confusion matrices exhibit a similar pattern, which enforces the belief that the model managed to not overfit the train set. The brightest class seems to be 'happy', which was predicted correctly 87% of the time in the test set and 94% in the train set. The small amount of examples of disgust is visible here: when y=disgust, x is only bright for disgust. Occasional brighter parts outside of the diagonal can be observed, these can be probably attributed to ambiguous pictures; the biggest confusion seems to be classifying fear as sadness, and classifying sadness as neutrality.

For the text emotion detection, the results were not satisfactory. The model obtained 39% accuracy, which was very low. The model found the most informative words, that is, words that strongly indicate the class, to be as follows:

```
Most Informative Features
           contains(sad) = True           sadnes : happin =     31.1 : 1.0
       contains(mothers) = True             love : sadnes =     23.1 : 1.0
         contains(hurts) = True            worry : love   =     19.9 : 1.0
        contains(stupid) = True           sadnes : love   =     19.8 : 1.0
         contains(moms) = True             love : sadnes =     19.4 : 1.0
          contains(use) = True           neutra : love   =     18.6 : 1.0
         contains(suck) = True           sadnes : happin =     18.4 : 1.0
        contains(sucks) = True           sadnes : happin =     18.2 : 1.0
         contains(hate) = True           sadnes : happin =     18.2 : 1.0
         contains(star) = True           happin : sadnes =     17.5 : 1.0
```

Fig. 10. The most informative features (words) given by the model.

6.1. Demo—Using the trained model a demo app was prepared which recognizes emotions in real time (60 times per second) from a camera feed. The app was written in TypeScript using the JS interface of Tensorflow called Tensorflow.js. There, the exported keras model was imported and warmed up in the WebGL environment. The webcam is started and each frame is streamed through the recognition pipeline. First, using the blazeface model[2] a face is detected (or not) and the bounding box is returned. If a face was found, the image is cropped to the face and normalized to the expected input of our model (apply grayscale filter, resize, and reshape). Once done, the face image is fed to our model and a prediction is made and finally displayed (Fig. 11). In this case the label was determined based off the argmax of the final softmax layer, there was no threshold.

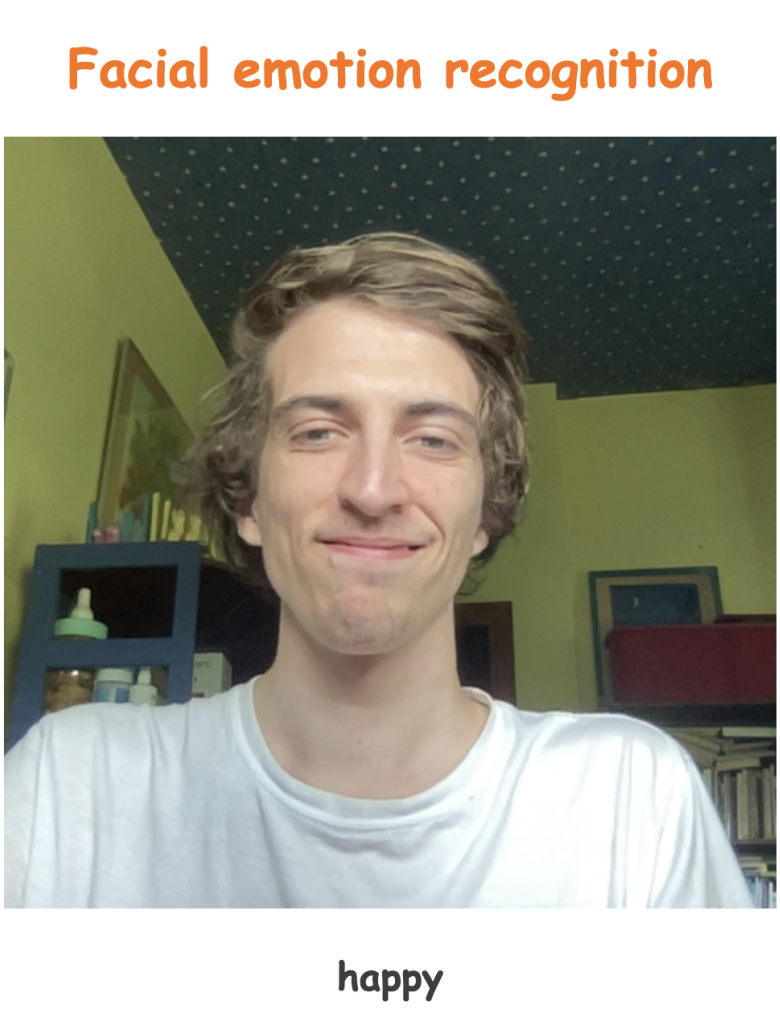

Fig. 11. Demo app deployed to be available online

Additionally, the app can run in debug mode where the normalized image is shown together with the full prediction distribution (Fig. 12).

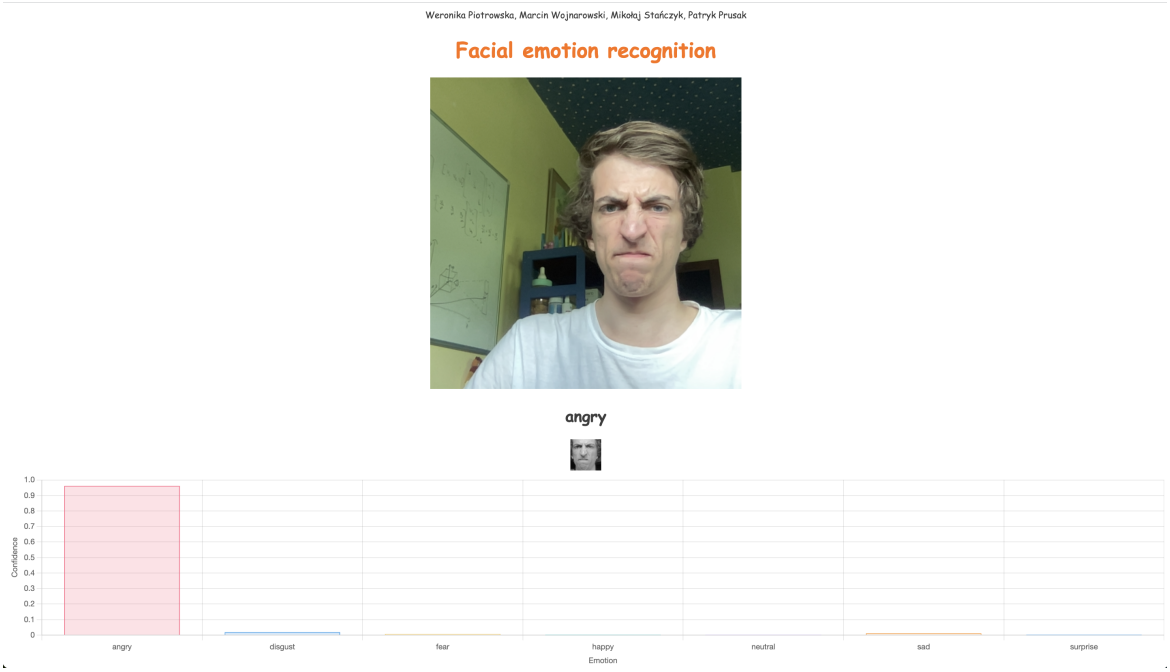

Fig. 12. Demo app showing the normalized input image and the output distribution

7. Conclusion

The results obtained from the image emotion recognition are very satisfactory. Despite the reported accuracy of only 67%, in practice the model yielded very accurate predictions. In the prepared demo, the model fared well and was able to predict emotions very often. Drawbacks of our solution include a poor dataset (emotions are exaggerated) which causes it to fail to recognize subtle hints of emotions, and a not-so-sophisticated model (deep, but basic).

However, the results obtained from text emotion recognition were not satisfactory. The model obtained 39% accuracy on the test set. We suspect the failure was because of wrongly labeled dataset, since some samples were labeled by two labels. Also, while analyzed the data, we found that the labels were sometimes not accurate for the given sample.

Due to that, we decided to not include text emotion recognition model in our GUI.

8. Attachments

1. text_src.zip – Source code of the text emotion detection notebook
2. image_src.zip – Source code of the image emotion detection notebook
3. demo_src.zip – Source code of the demo for real-time webcam emotion detection
4. https://github.com/shilangyu/AIF-emotion-detection – repository with all the whole work

Notes and References

[1] Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., Ravi, S. "GoEmotions: A Dataset of Fine-Grained Emotions." (2020) doi:10.48550/ARXIV.2005.00547 URL https://arxiv.org/abs/2005.00547.

[2] Bazarevsky, V., Kartynnik, Y., Vakunov, A., Raveendran, K., Grundmann, M. "BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs." (2019) doi:10.48550/ARXIV.1907.05047 URL https://arxiv.org/abs/1907.05047.