Università Sapienza
AML, First Semestre 2020

# Facial Emotion Recognition an Unbalanced Task

Isikli Dilara
id 1891292

Jambaladinni Pooja
id 1911273

Katirci Vedat
id 1908990

Russo Dario
id 1714011

January 8, 2021

**Abstract–Voice is not the only tool exploit in the communication between people, as Dr. Albert Mehabrian claims, people interacts each other also with other means and the facial expressions represents the 55% of them. Being able to recognise at least the basic expressions can be helpful in many fields of research and piratical applications, such as psychology and marketing. The aim of this paper is to show how detect 7 of them *(angry, disgust, fear, happy, neutral, sad, surprise)* by deep learning, trying to overcome the main issues related to the most spread used database for this task (FER).**

# 1 Introduction

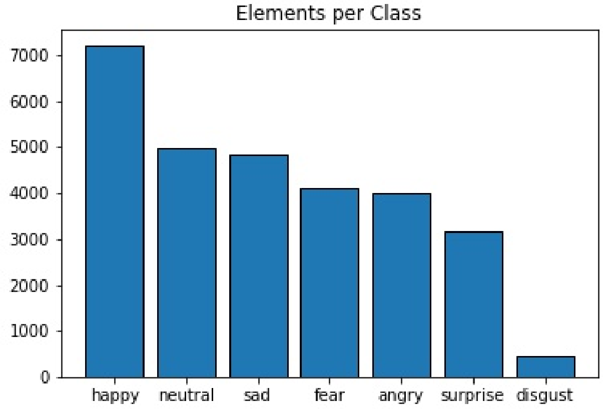
As one of the main tool to communicate each other, facial emotions recognition is wide and open research field. Since first completion came up on Kaggle in 2013 [2], the ability to detect them has grew up to a reach 98.9% accuracy **under controlled conditions**. However, same task under **natural conditions** is still an open task, due to issues such as illumination, pose and occlusion [4]. Nonetheless, these issues, the state of the art model has reached 75.42% on FER database, combining different models by voting [3][11], beating the human ability, which has an accuracy of about 65%.
The **size** (28709) of the FER database is one of the main reason that prevents deep networks to achieve a higher accuracy, it is hard to train a too deep network on it. Moreover, classes in this data-set are **highly unbalanced**. Our aim in this paper is to show how is it possible to partially overcome these kind of problems.

# 2 Related Work

Being a so interesting and useful filed, many authors study it, bringing a huge amount of technique and solutions. A first classification can be done between static and dynamic approach, the former is **based on images** and it is the one we choose, the latter concerns video frames. [12] The winner of the Kaggle competition is Yichuan Tang and achieved a score of 71.2%, thanks mainly to the use of the **SVM Loss** [10]. Nowadays, the best scores are achieved by combining more models together by **voting**, the best models at the moment are the ones developed by Pramerdorfer and Kampel [8] and Zang [11], with an accuracy in turns of 75.2% and 75.1%. The most of the authors reprocess the images in order to crop the face and find landmarks [8][12][11], some of them also align the face [5][12]. It also interesting to see how **data augmentation** plays an important role in this challenge, the most of the authors to achieve a higher a accuracy use pictures from other database such as CS+, AffectNet, JAFFE and MMI [12]. Moreover, other data augmentations such as translation, rotation and flips are widely used. The approach that leads to best results in this field is the one made by Porcu, Floris and Atzori which increase the number of pictures creating new ones, thanks to a **GAN** [7], reaching an accuracy of 83.3%. For what concerns the type of networks used, the CNN is the most common, but also other architecture are used such as DBN, DAE, RNN and so on.[12]

# 3 Datasets

For our task we only use FER Dataset (made by 35887 gray-scale images normalized to 48x48 pixels and 7 classes) because it the more challenging due to its structure and its issues. This databse is strongly **unbalanced** and have a small number of pictures (compared to the other ones). Furthermore, it is highly probable that n**not all labels are assigned correctly**, which means that it is impossible to reach 1 as accuracy. **FER+** [1] try to solve the "misslabled" images, giving for each picture a set of possible labels. At the moment, the most promising database is the AffectNet [6], which contains more than 1,000,000 facial images retrieved from the web (this fixes the first two issues).

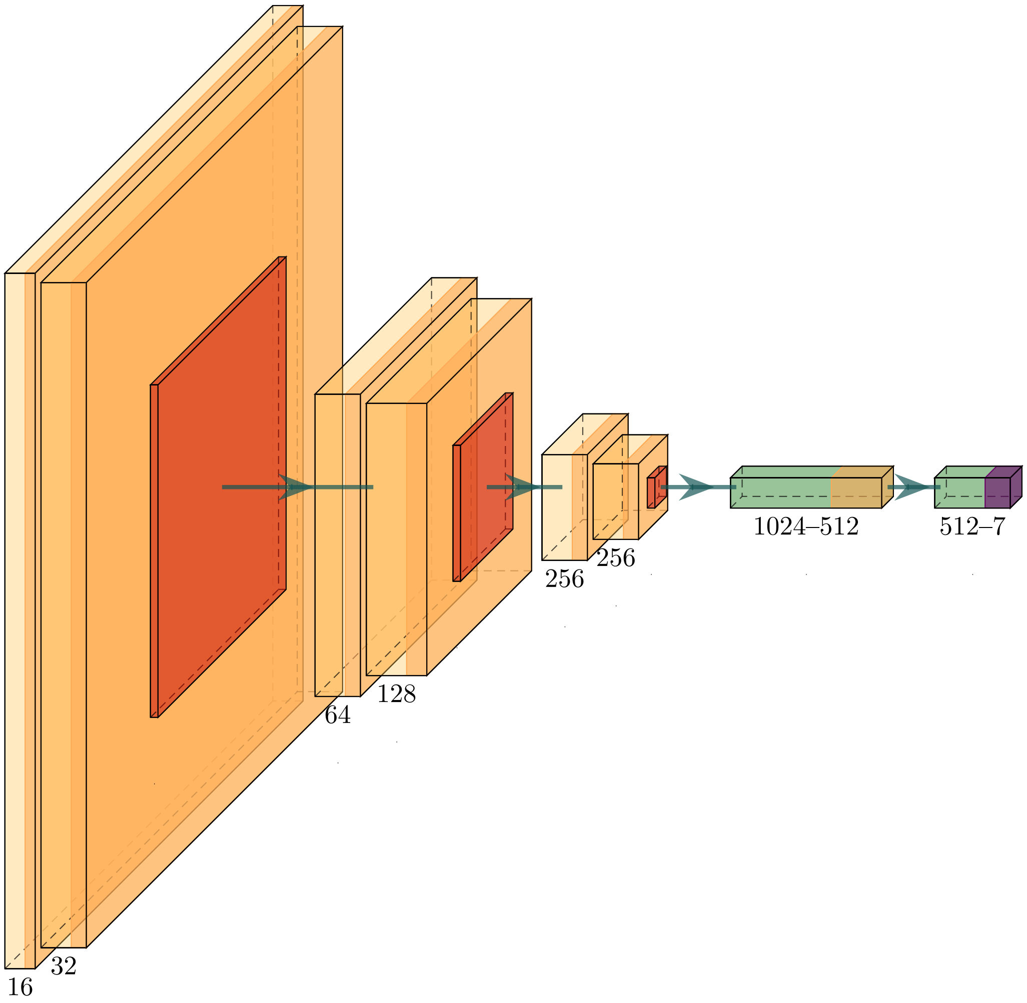


# 4 Proposed method

After reading many papers about the subject, we thought about creating our network, which would have been our base line, for our task of improving accuracy working on the database.

## 4.1 The Model

As Pramerdorfer and Kampel says [8] the networks that perform better in this task are often the smaller [8][9]. According to this reasoning, our Net is made by only **6 Convectional Layers**, all of them with a **kernel size of 3**, they in turn have the following number of filters (16, 32, 64, 128, 256, 256), and textbf2 FC layers, the first has an input size of 1024 and output size 512 and the other one has an output size equal to the number of the classes (7). As activation function we use the ReLU for all layers, but the last, in which we apply the SoftMax. Two different kind of **dropouts** are applied, for the Convectional layers the percentage is set to 20% and it is put with an 'off/on' strategy, namely one layer not and one layer yes. With the same strategy we apply a **MaxPool** with kernel 2 and stride 2. For the dropout of the FC we use a probability of 50% and we put it only on the first FC layer. After each CNN layer is applied the **Batch Normalization**, before the activation function.
We use **a Cross Entropy Loss**, **Adam** as optimizer and two learning rate scheduler, one that reduces learning rate exponentially each epoch and another that reduces by a factor of 0.1 the learning rate every time it gets stack for more than 5 epoch. This setup will be hold also for other models, the only thing that will change may be the loss.



Parameter optimization has been done with Optuna library, but it ends up that the most common known parameters work well for all our models: learning rate (0.001), learning rate decay (0.9) and weight decay(0.0005).
This model starts to have a fair performances around 57% of accuracy, but it is the one among all the ones we are going to test with the lower level of recall, around 53.74%. On the other hand, this the best model in terms of **classification homogeneity**, the accuracy for each class is between 40% and 50%, but the classes with the higher number of pictures reach up to 70% of accuracy.

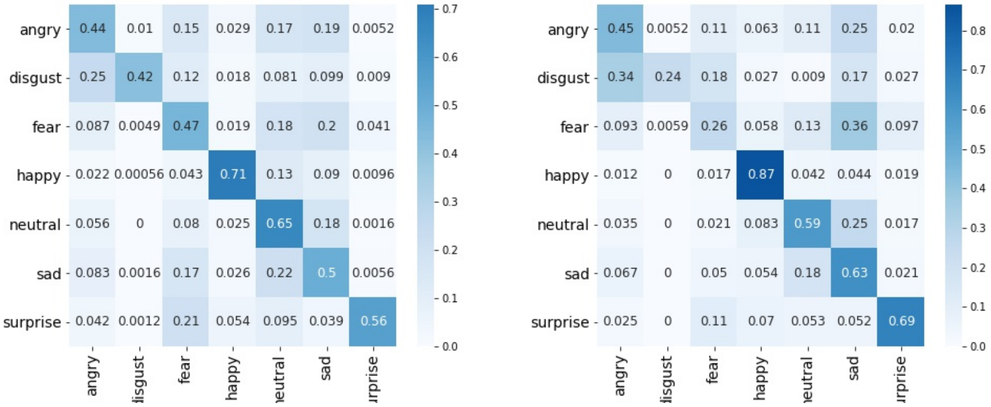
| Precision | Recall | F1 | Accuracy |
|---|---|---|---|
| 0.6149611568 | 0.5373467523 | 0.5612054064 | 0.572304263 |

## 4.2 Balancing Classes

The main issue with a so unbalanced dataset is that the network risks to learn very well only the most common labels and miss classify other ones or in the worst scenario, it classifies the most of the pictures in the most common classes. To avoid this situation we modify the Data Loader, in pytorch, to produce batches with the same amount of picture for each class. The weights for each class are choose by dividing the total number of picture for the ones in the given class:

$$w_j = \frac{N}{\sum_i^N I(c_i=j)}$$

To further improve the model, instead of using the Cross Entropy loss, we are going to opt for the **negative log likelihood loss**, which performs quiet well with unbalanced datasets.



(a) base line confusion matrix

(b) balanced batch confusion matrix

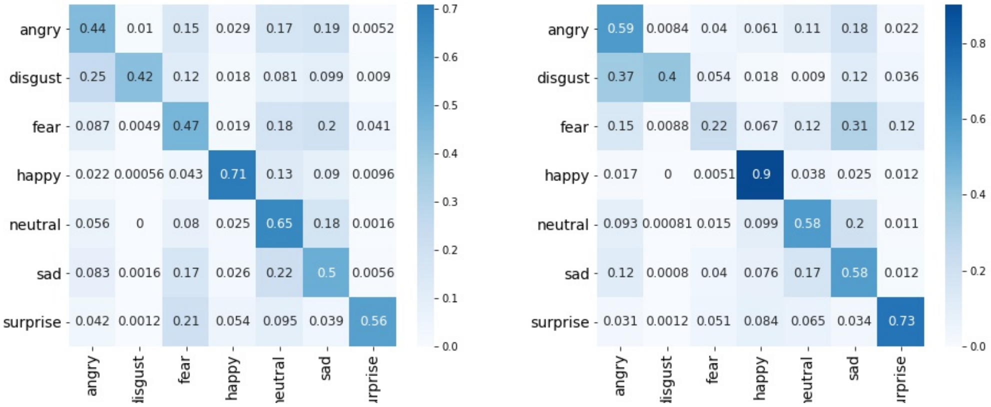
It seems that we obtained the opposite result: even if we improved the overall accuracy, the model predicts worse the classes with fewer elements.

| Model | Precision | Recall | F1 | Accuracy |
|---|---|---|---|---|
| base line | 0.6149611568 | 0.5373467523 | 0.5612054064 | 0.572304263 |
| balanced batch | 0.6106891815 | 0.5315175727 | 0.5467941792 | 0.6044859292 |

## 4.3 Data Augmentation

After balancing, we have seen that the model improves its accuracy, but it lowers its ability to predict well classes with fewer elements. To try to deal with this issue, we also implemented data augmentation. We use three simple transformations that seem to work very well: horizontal flip, rotation (-20,20) and translation (10, 10) [7]. For this task, we use again Cross Entropy Loss and no balanced batch.

| Model | Precision | Recall | F1 | Accuracy |
|---|---|---|---|---|
| base line | 0.6149611568 | 0.5373467523 | 0.5612054064 | 0.572304263 |
| augmented data | 0.6231450634 | 0.5708364125 | 0.57813852 | 0.6235720256 |



(a) base line confusion matrix

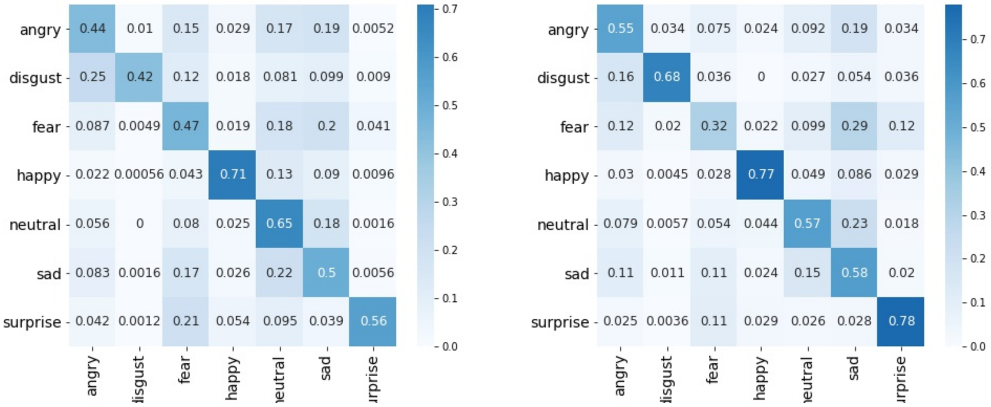
(b) augmented data confusion matrix

It is clear from the results, that even if we increase the accuracy again, it is due to the fact that the most copulated classes are detected well. Instead, the other ones still have the same problem (miss classification). In addition, with more data, the model seems more incline to confuse some emotions. We can say that the model with augmented data is better also in terms of the **ROC analysis**, studied as average behaviour of the all components.

## 4.4 Combine the results

The combination of these two technique (without the use of negative likelyhood log loss) seems to give the best results:

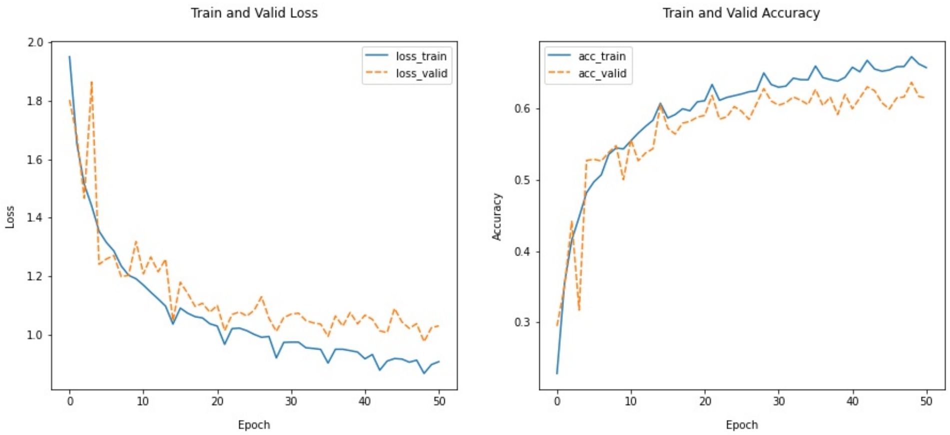
- **miss-classification** is reduced, but at the same time there is an increasing of the accuracy of the most populated classes.



(a) base line confusion matrix

(b) augmented and balanced data confusion matrix

- **model learning** ability of learning are optimized.



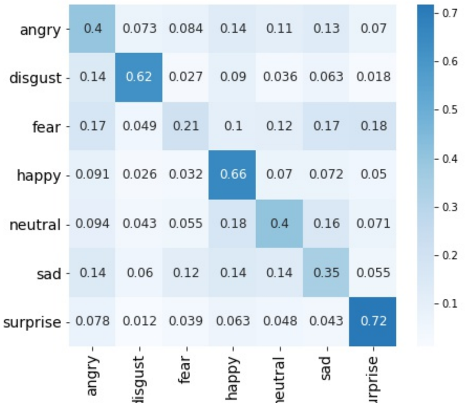
.jpg

- **accuracy level** decreases about 1% (maybe training with more epochs or adjusting hyper parameters will solve this). But in terms of **F1-score**, this model is the best with a score of 59%.

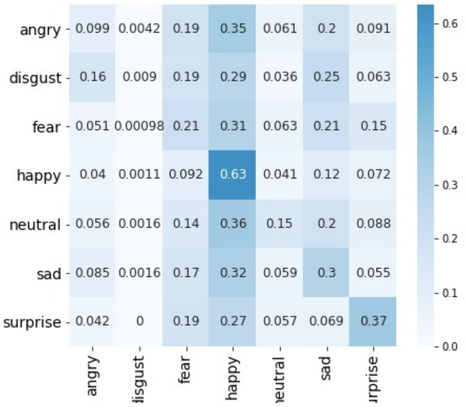
| Model | Precision | Recall | F1 | Accuracy |
|---|---|---|---|---|
| base line | 0.6149611568 | 0.5373467523 | 0.5612054064 | 0.572304263 |
| augmented data | 0.6231450634 | 0.5708364125 | 0.57813852 | 0.6235720256 |
| balanced batch | 0.6106891815 | 0.5315175727 | 0.5467941792 | 0.6044859292 |
| balancing augmented | 0.5846289605 | 0.6087149768 | 0.5901687004 | 0.6100585121 |

## 4.5 Transfer Learning

As we said before, models with too many parameters don't work well on FER dataset, for this reason, we choose the smallest version of two famous architecture: VGG11 and ResNet18. We redesigned a little bit the architecture of the former, replacing the last two pool layers before the classifiers with an aaverage pool with kernel 2 and stride 1. Then, the original classification layer has been removed and replaced with three fully connected layers, with input and output size of: FC1 (2048, 1024), FC2 (1024, 512), FC3 (512, 7). Between the first and the second FC layers there is a dropout of 30%.
For the training (only of the new layers) we use the standard strategy with the Cross Entropy, no balancing and with augmentations.
The results of the VGG11 are almost 10% lower than our baseline, but it seems to be better in terms of miss-classification. Instead, the RES18 is too complex for our model with an accuracy of 32%, moreover it classifies the most of pictures in few categories. Anyway, if take a look at the loss and accuracy curve, it is clear that it stops to learn quiet soon and then the curves are wavy, if we probably spent more time on hyper parameters tuning, this last net could perform well.



(a) VGG11 confusion matrix



(b) RES18 confusion matrix

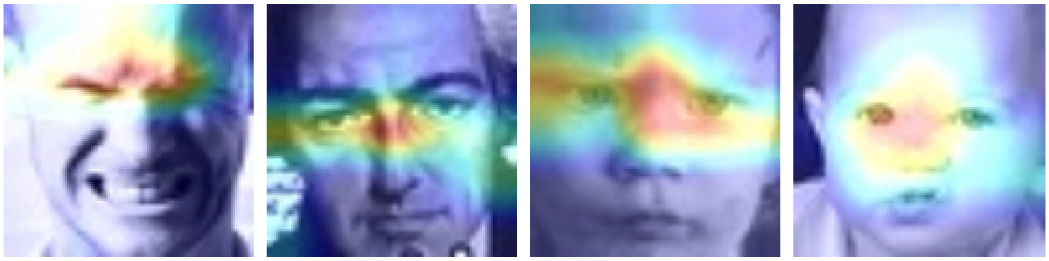
| Model | Precision | Recall | F1 | Accuracy |
|---|---|---|---|---|
| base line | 0.6149611568 | 0.5373467523 | 0.5612054064 | 0.572304263 |
| VGG11 | 0.4182989676 | 0.4794978768 | 0.426692578 | 0.4682362775 |
| RES18 | 0.2697199369 | 0.2544557002 | 0.2440763625 | 0.3220952912 |

## 5 Interpretation

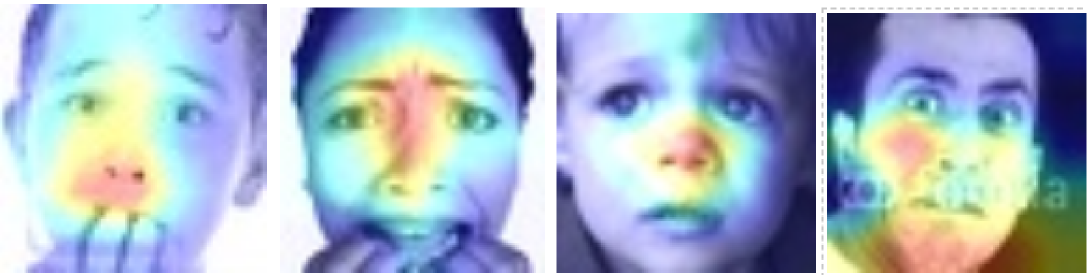
For interpret-ability of our model, we decided to use Grad-CAM method. This method calculates gradient with respect to predicted class and backpropagate it for the last convolutional layers activation function. By using this gradient, we are creating heat-map and then we are resizing this heat-map at the size of the images and applying them on the images to see which part of the image is more sensitive to the small changes.The result that we get by using Grad-CAM are below;

1. As we have observed, the model most of the time focusing some specific area for each emotion. Moreover, we found that eyes are crucial to detecting the emotions because our model most of the cases affected by area around eyes. Below you can see our general interpretation for some classes:

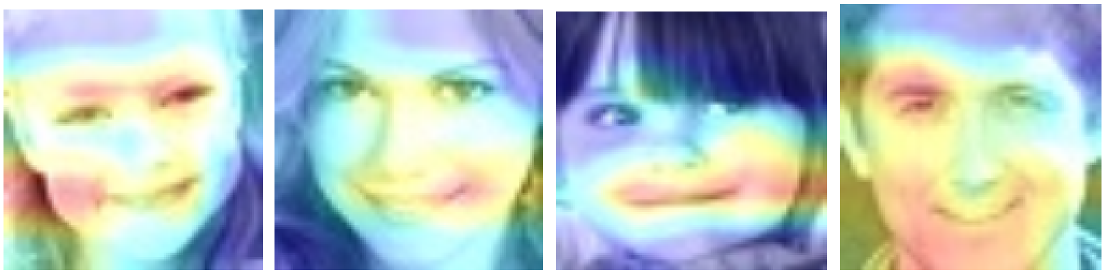
    - Anger: Most of the cases this emotion detected by the upper part of the face.



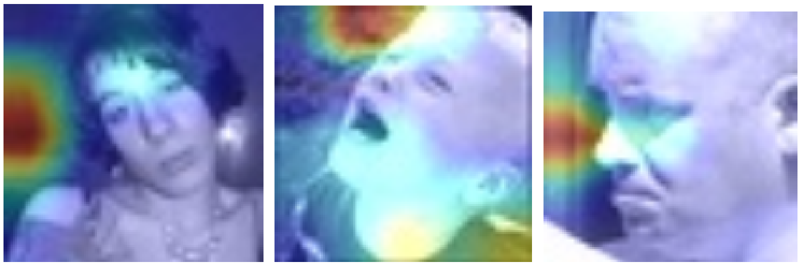
    - Fear: Most of the cases this emotion detected by the around nose and lower part of the face.



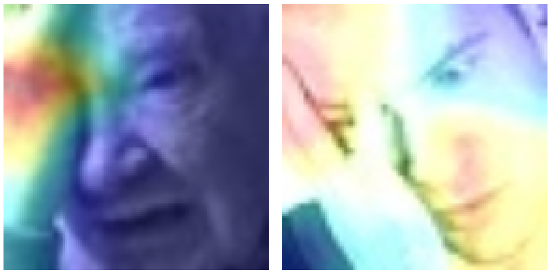
    - Happiness: Most of the cases this emotion detected by the around mouth and lower part of the face.



2. It is realized that, sometimes model is deciding emotions by focusing irrelevant area of the images and this case is valid for both false and true prediction. We believe that this case is misleading in terms of the calculation accuracy because model predicting the true class by looking background of the images. We have realized that this is happening most of the time when faces are not localized in a good way.



3. For the occlusion some cases, model able to predict correct class for emotions but it is hard to interpret the situation. As an example, old lady put his hand in front of her eye. Maybe model was able to learn this gesture (most of the time when people put their hand in this way, they are sad) or model is misled by the features that are extracted from her hand.



## 6 Conclusion

In this paper we have seen how an unbalanced dataset can badly influence the process of learning of a network. However, with the right tools it is possible to overcome this issue, at least in part. In our specific case a good combination between data augmentation and batch elements balancing, brings an improvement 0f 4% in the network and an high quality model. On one side, with data augmentation it is possible to increase the overall accuracy, thanks to the most populated classes. On the other side, the balancing of the batches (same number of elements for each class) implies a classifier more balanced on all the possible labels, even if it reduces a little bit the accuracy. With our final model, we can increase even the F1-score of 3%, thing that doesn't happen when we use these two tools separately.

## 7 Future Steps

The main problem in this dataset still remains the unbalanced classes distribution, to overcome it, we suggest to increases the observations of the classes with fewer elements, thnaks to Generative Adversarial Network, in such a way that at the end of the process all classes have the same number of elements. In addition, it is also possible to opt for a voting, in our case combining our network with the VGG11, which has the ability of reduces the miss-classified labels.

To improve the performance, it is also recommended to add some prepossessed steps, like face and landmarks recognition, to cut and align the face. This prevent to the fails that we have our network, for some picture in which there is also a background. For what concerns occlusion, instead, it is recommended to explore dipper the random erasing, which in our case doesn't really bring to a better result.

# References

[1] Fer+. URL https://github.com/microsoft/FERPlus.

[2] Fer-2013. URL https://www.kaggle.com/msambare/fer2013.

[3] M.-I. Georgescu, R. T. Ionescu, and M. Popescu. Local learning with deep and handcrafted features for facial expression recognition. URL https://arxiv.org/pdf/1804.10892.pdf.

[4] A. Khanzad, C. Bai, and T. F. C. Facial expression recognition with deep learning. 2020. URL http://cs230.stanford.edu/projects_winter_2020/reports/32610274.pdf.

[5] B.-K. Kim, S.-Y. Dong, J. Roh, G. Kim, and S.-Y. Lee. Fusing aligned and non-aligned face information for automatic affect recognition in the wild: A deep learning approach. URL https://openaccess.thecvf.com/content_cvpr_2016_workshops/w28/papers/Kim_Fusing_Aligned_and_CVPR_2016_paper.pdf.

[6] A. Mollahosseini, S. Member, B. Hasani, and M. H. Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. URL https://arxiv.org/pdf/1708.03985.pdf.

[7] S. Porcu, A. Floris, and L. Atzori. Evaluation of data augmentation techniques for facial expression recognition systems. 2020.

[8] C. Pramerdorfer and M. Kampel. Facial expression recognition using convolutional neural networks: State of the art. URL https://arxiv.org/pdf/1612.02903.pdf.

[9] shangeth. Facial-emotion-recognition-pytorch-onnx. URL https://github.com/shangeth/Facial-Emotion-Recognition-PyTorch-ONNX.

[10] Y. Tang. Deep learning using support vector machines. 2013. URL http://www.cs.toronto.edu/~tang/papers/dlsvm.pdf.

[11] Z. Yu and C. Zhang. Image based static facial expression recognition with multiple deep network learning. . URL https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/icmi2015_ChaZhang.pdf.

[12] Z. Yu and C. Zhang. Deep facial expression recognition: A survey. *Shan Li and Weihong Deng*, . URL https://arxiv.org/pdf/1804.08348.pdf.