# Investigating the problem domain of FER in the wild

Shabnam Khodadadi
Queen Mary University of London
London, E1 4NS
s.khodadadi@se21.qmul.ac.uk

## 1. Introduction

In this article, we are investigating the problem domain of Expression Recognition (FER) in the wild (uncontrolled environment). In the previous essay we have described the problem and the related works in this area Also the preprocessing steps that we have done to prepare the dataset for training. In the following we will provide the information of the models chosen for experiments the progress and the results. The contents devided to four main parts, Main Machine Learning Model, Baseline Machine Learning Model, Ablation Study and Production Data results.

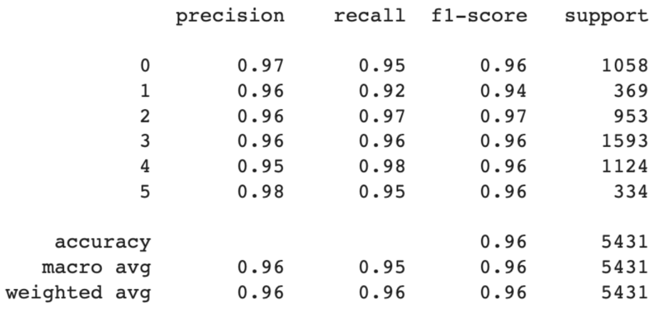
## 2. Task 1: Main Machine Learning Model

To solve the Facial Expression Recognition(FER) in the wild we choose VGG16 as a base structure of our main model to train, following [2] which use this network as a baseline of expression classification challenge for Aff-Wild2 dataset in second ABAW2 Competition [4]. We use pretrained model and transfer learning approach because we have limited training samples although complex deep networks need large amount of training data, also to decrease the computation time due to our available limited resources. VGG16 [6] pretrained on imagenet dataset to predict 1000 classes, However in our problem we have the 6 classes to predict so we have to adjust the output layer to make it suitable for our own problem. Therefore, we loaded the model without a top, so model's fully-connected layers and the output layer, are not loaded, allowing us to add new layers to be trained. Overall, our architecture consists of five blocks of convolutional layers with pre-trained weights and max pooling layer, following a flatten layer to prepare the CNN output features for next three fully connected layers with 4096 units and relu activation function. Finally we use the dense layer with softmax activation function for classification, this function takes as input a vector of real numbers, and give the output vector of probabilities of each class, the output of layer is set to six which is equal to our class numbers. We trained the network with the train set of our preprocessed images(Aligned, normalised, resized) and use the validation and test set of normalized but non-alighened 32*32 images for evaluation. As mentioned before, the CNN layers weights are pretrained and only the fully connected layers will trained with our data. At first, we set the input of VGG16 network to (96,96,3) as we have 96*96 pixel RGB(3 channel) images but it takes six hours to train with 10 epoch with our available resources, so we resized our images to 32*32 and train the network with the input of (32,32,3) with 10 epoch and batch size of 256 also the learning rate of 10-4 . Adam is used as the optimiser and for loss function we used sparse categorical cross entropy [2] [4]. The result of our model for train and validation data is: train-loss: 2.6268e-04 - train-accuracy: 1.0000 - val-loss: 0.1841 - val-accuracy: 0.9685 and the accuracy of test set is 0.97. The value of train accuracy show that our model faced the problem of overfitting because of limited training samples and complex network, so we decided to decrease the FC layers from three to two also use dropout layer with rate of 25% after each fully connected layer to reduce the overfitting, consequently the result changed to train-loss: 0.0300 - train-accuracy: 0.9913 - val-loss: 0.1407 - val-accuracy: 0.9645 and the test accuracy of 0.960780680179596. As our problem is multi-class problem and we have an unbalanced dataset, the accuracy is not enough to evaluate the model. Therefore we calculate the f1-score, precision and recall of the predicted labels for each class. Figure 1 illustrates the results 1. Our model has the f-score of 96% and the precision and recall of near 1 which show its good performance on test set.

## 3. Task 2: Baseline Machine Learning Model

Another alternative model is trained to solve the studied problem. We choose MobileNetV2 as it is a light-weight deep neural network would be suitable for our small dataset. This model also used as a baseline for FER in first ABAW competition [3]. We used the pretrained model and add the two trainable fully connected layer with 128 and 64 units respectively and relu activation function and on top of that using a softmax layer with six output. Training done under the same condition of our main model with the same hyper-

Figure 1. result of VGG16 on test set

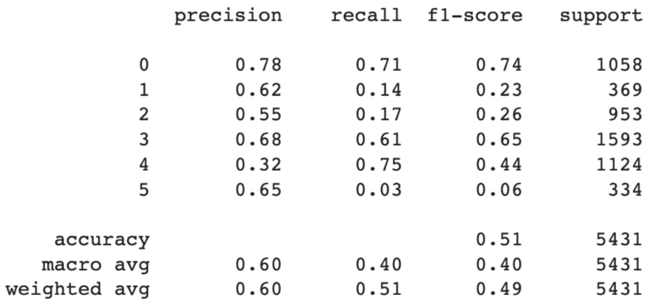


```
              precision    recall  f1-score   support

           0       0.97      0.95      0.96      1058
           1       0.96      0.92      0.94       369
           2       0.96      0.97      0.97       953
           3       0.96      0.96      0.96      1593
           4       0.95      0.98      0.96      1124
           5       0.98      0.95      0.96       334

    accuracy                           0.96      5431
   macro avg       0.96      0.95      0.96      5431
weighted avg       0.96      0.96      0.96      5431
```

Figure 2. result of MobileNetV2 on test set



```
              precision    recall  f1-score   support

           0       0.78      0.71      0.74      1058
           1       0.62      0.14      0.23       369
           2       0.55      0.17      0.26       953
           3       0.68      0.61      0.65      1593
           4       0.32      0.75      0.44      1124
           5       0.65      0.03      0.06       334

    accuracy                           0.51      5431
   macro avg       0.60      0.40      0.40      5431
weighted avg       0.60      0.51      0.49      5431
```

parameters, optimiser and loss functions. We provide the input size of (32,32) to MobileNetV2 but it doesnt have the proper weights for this image size so the default weights for images of size 224 are loaded instead, we mentioned before that we don't have enough resources to process large images. The result of training MobileNetV2 model with our train,validation sets are, train-loss: 1.0741, train-accuracy: 0.2244, val-loss: 1.1371, val-accuracy: 0.5397 and the accuracy for the test data is 0.5409. For our problem with 6 class and cross entropy loss function, the loss value in first epoch should be around 1.7 (ln(1/6)) but in the results we can see that even after 100 epoch the loss value remained the same, show that the learning process is not good, this problem may because of insufficient pretrained weights that we mentioned before. Figure 2 exibits the other metrics result 2, the model has a general f1-score of 51%. Precision have the average value of 50% for all classes except HAPPINESS with value of 32%, It means that the model predict some sample of other classes as HAPPINESS But the difference is not considerable. Recalls for classes illustrate that the model is not good for FEAR, SURPRISE, DISGUST. The difference between our main model results and this model is that the main model perform equally for all classes but baseline model is not in addition to the point that main model has the highest accuracy and f1-score in train,validation and test sets. We will provide the comprehension analyse of results in task4 section.

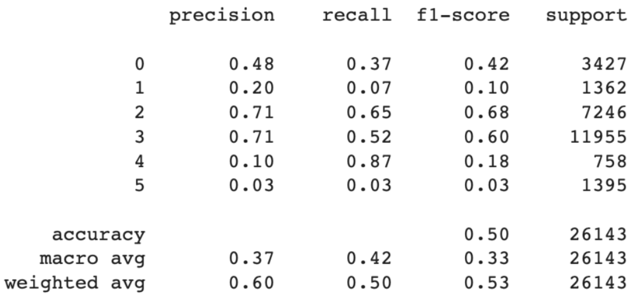
## 4. Task 3: Ablation Study

We perform an ablation study on our main model VGG16 as it has a better performance with the test set in comparison to MobileNetV2. Model trained with different value of batch size, learning rate, epochs and units used in fully connected layers. Figure 5 5 show some of the best results and differences. The VGG16 model with 4096 units of fully connected layers, batch size of 256, learning rate of 0.0001 and epochs equal to 100 has the best performance so we will use this model and hyperparameters to predict the production dataset.

## 5. Task 4: Production Data

Main model of VGG16 based is selected to predict the production dataset classes. Based on [4], the baseline accuracy for the task of FER in the wild for Aff-Wild2 dataset is 46% and the F1-score is 26%. Our result is loss 1.750, accuracy 0.496 , Figure 3 illustrates the results 3 Analyzing the f1-score, precision and recall of all classes, Precision shows that from all the samples that model predict for a specific class, how many of them truly belong to that class. HAPPINESS and DISGUST has the precision of approximately 7% which is very low in comparison to other classes with 60% value. It can be inference that that the model predict many samples as HAPPINESS and DISGUST though their not truly belong to these classes. This problem happened because of the large inter-class similarities. Expressions from different classes may only exhibit some minor differences. For example, SURPRISE and HAPPINESS has the similar mouth action, people open their mouth in both of these expressions, these similarities between classes cause the model to predict similar classes as one class. Figure 4 4 exhibit the confusion matrix which confirm this assumption, 6% of samples with SURPRISE class wrongly predict as HAPPINESS, also 11% of SADNESS predict as HAPPINESS. The reason that model predict most of the classes as HAPPINESS is not an unbalanced data, as we have more SADNESS samples in our dataset than HAPPINESS. This happened because it is more easier to recognise HAPPINESS emotion with focus on only the mouth but for SURPRISE and SADNESS we should consider the other area of face like eyes and areas between eyes, in other word we should consider global approaches rather than local approaches [7]. Analysing the recall value, recall is the ratio of correctly predicted samples in each class. HAPPINESS has the high recall value of 92% shows that the model perfectly predict this class, However poor performance in FEAR and DISGUST. This weakness maybe due to the Small intraclass similarities. Expressions from the same class might look very differently depending on a person's ethnicity, gender, age, and cultural background. With respect to f1-score values, the model perform good in SURPRISE and SADNESS However poorly in FEAR and DISGUST it may result of limited data in these classes, In details most of FER in the wild dataset are unbalanced, due to the different possibility of collecting various expressions, for example it is more easier to collect HAPPINESS samples in comparison to rare emotion like DISGUST. Fix the problem of unbalanced dataset will improve the model performance, we have done this with both data augmentation and batch balancer on the fly for small size of images but couldn't apply it to 32*32 images due to memory limitation. For general improvement of FER data related problems, we can use the meta data of videos or images for example the scene information can give us prior probability information of happening of specific emotion [1]. Moreover, they are model related improvement exists. Firstly, in our experiments we resize the images to 32*32 although their original size is 128*128. Use the original size of image for training will improve the performance of model in production data as they are more features to learn for the model which increase its generality [3] [4]. Secondly, we use transfer learning and cross-dataset approach, use the pretrained model weights trained on imagenet dataset that is different from our dataset in the case such as illumination variation, occlusions, pose and label annotations. With having the enough resources, training the model from scratch will result to model performance improvement as it learn and test with same dataset attributes [5]. Finally, the approach that used for splitting the dataset to train and test will affect the performance, we split the data based on classes, take into consideration the point that both train and test data should contain sample of all type of classes.

Figure 3. VGG16 results on production data

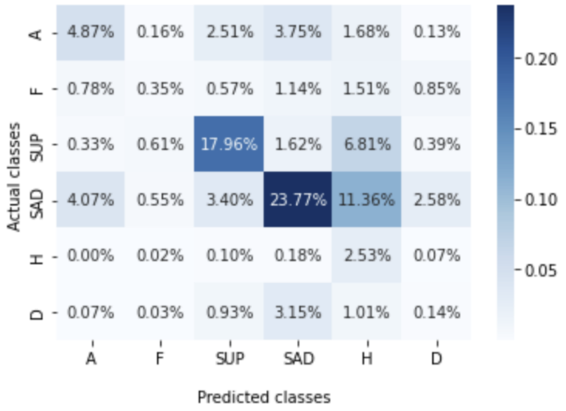


```
              precision    recall  f1-score   support

           0       0.48      0.37      0.42      3427
           1       0.20      0.07      0.10      1362
           2       0.71      0.65      0.68      7246
           3       0.71      0.52      0.60     11955
           4       0.10      0.87      0.18       758
           5       0.03      0.03      0.03      1395

    accuracy                           0.50     26143
   macro avg       0.37      0.42      0.33     26143
weighted avg       0.60      0.50      0.53     26143
```

Figure 4. confusion matrix production data



## References

[1] Damien Dupre, Nicole Andelic, Gawain Morrison, and Gary McKeown. Assessment of automatic facial expressions recognition" in the wild": a time-series analysis using gamm and sizer methods. In *20th International Conference on Human-Computer Interaction*, 2018. 3

[2] Dimitrios Kollias. Abaw: Valence-arousal estimation, expression recognition, action unit detection & multi-task learning challenges. *arXiv preprint arXiv:2202.10659*, 2022. 1

[3] D Kollias, A Schulc, E Hajiyev, and S Zafeiriou. Analysing affective behavior in the first abaw 2020 competition. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)(FG)*, pages 794–800. 1, 3

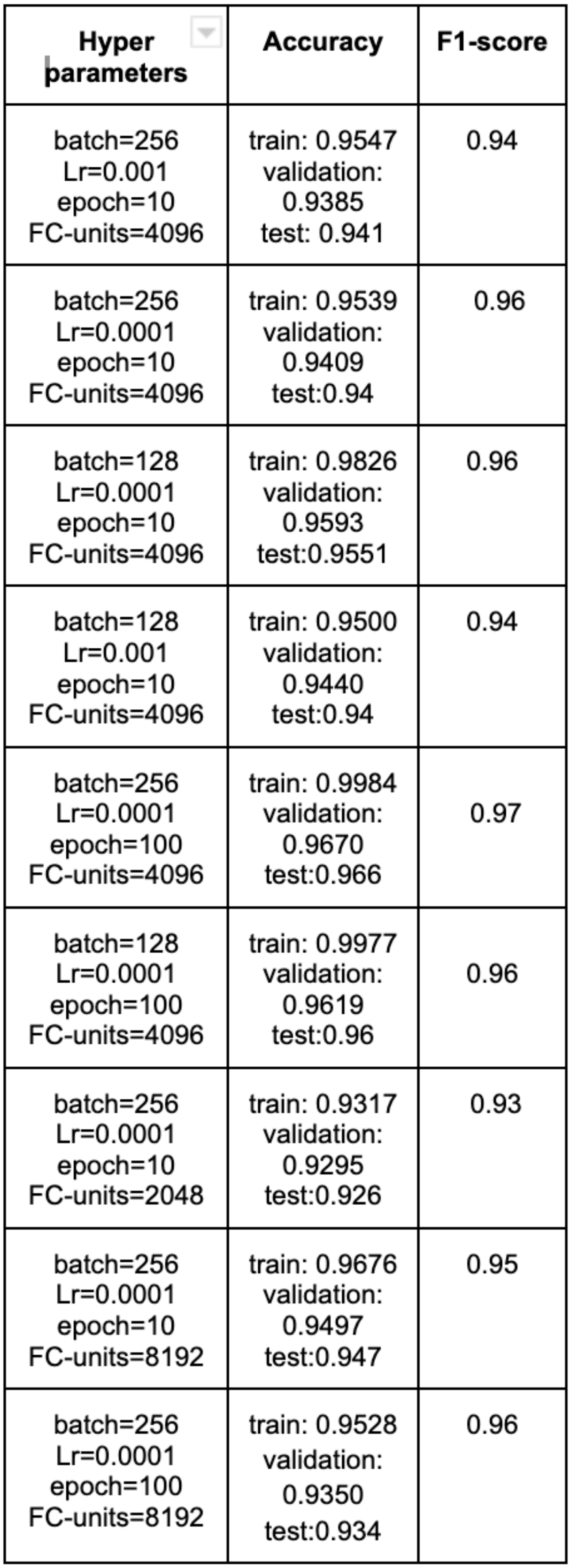
[4] Dimitrios Kollias and Stefanos Zafeiriou. Analysing affective behavior in the second abaw2 competition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3652–3660, 2021. 1, 2, 3

[5] Shan Li and Weihong Deng. Deep facial expression recognition: A survey. *IEEE transactions on affective computing*, 2020. 3

[6] Hussam Qassim, Abhishek Verma, and David Feinzimer. Compressed residual-vgg16 cnn model for big data places image recognition. In *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, pages 169–175. IEEE, 2018. 1

[7] Fanglei Xue, Qiangchang Wang, and Guodong Guo. Transfer: Learning relation-aware facial expression representations with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3601–3610, 2021. 2

Figure 5. ablation study results VGG16



| Hyper parameters | Accuracy | F1-score |
|---|---|---|
| batch=256<br>Lr=0.001<br>epoch=10<br>FC-units=4096 | train: 0.9547<br>validation:<br>0.9385<br>test: 0.941 | 0.94 |
| batch=256<br>Lr=0.0001<br>epoch=10<br>FC-units=4096 | train: 0.9539<br>validation:<br>0.9409<br>test:0.94 | 0.96 |
| batch=128<br>Lr=0.0001<br>epoch=10<br>FC-units=4096 | train: 0.9826<br>validation:<br>0.9593<br>test:0.9551 | 0.96 |
| batch=128<br>Lr=0.001<br>epoch=10<br>FC-units=4096 | train: 0.9500<br>validation:<br>0.9440<br>test:0.94 | 0.94 |
| batch=256<br>Lr=0.0001<br>epoch=100<br>FC-units=4096 | train: 0.9984<br>validation:<br>0.9670<br>test:0.966 | 0.97 |
| batch=128<br>Lr=0.0001<br>epoch=100<br>FC-units=4096 | train: 0.9977<br>validation:<br>0.9619<br>test:0.96 | 0.96 |
| batch=256<br>Lr=0.0001<br>epoch=10<br>FC-units=2048 | train: 0.9317<br>validation:<br>0.9295<br>test:0.926 | 0.93 |
| batch=256<br>Lr=0.0001<br>epoch=10<br>FC-units=8192 | train: 0.9676<br>validation:<br>0.9497<br>test:0.947 | 0.95 |
| batch=256<br>Lr=0.0001<br>epoch=100<br>FC-units=8192 | train: 0.9528<br>validation:<br>0.9350<br>test:0.934 | 0.96 |