Facial Expression Recognition with Machine Learning

Duarte Rodrigues[1]
up201705420

Emanuel Ricardo Brioso[1]
up201708998

Esmeralda Cruz[1]
up202003602

ABSTRACT The ability to automatically recognize facial expressions provides a wide range of applications, from the diagnosis of some neurological disorders to providing information about the emotion of a target audience towards a marketing message, brand, or product for companies, as well as many others. Consequently, there has been a lot of research in this field, by applying deep learning approaches, including convolutional neural networks (CNNs) for feature extraction, which has proved to be a success. In this paper, two architectures of CNNs were applied, a custom CNN and ResNet-50, to identify the key seven human emotions: *anger, disgust, fear, happiness, sadness, surprise* and *neutrality*. The influence of transfer learning was also tested, by initializing the models with the weights of ImageNet database. After training the models, their results of accuracy and loss were analyzed, compared with each other, as well with some results found in literature. For the ResNet-50 model, the accuracy obtained for initialization with random weights and ImageNet weights was 0.6294 and 0.6372 and for loss it was 0.9717 and 0.9715, respectively. For the custom CNN model, the accuracy and loss were 0.6492 and 0.9407, respectively.

INDEX TERMS Facial Expressions, Deep Learning, Convolutional Neural Networks, ResNet-50

I. INTRODUCTION

Being able to recognize and understand facial expressions is a very important part of nonverbal communication, which consists of two-thirds of human interactions. This research field was introduced by Charles Darwin in his book *“The Expression of the Emotions in Man and Animals”* and has since then received significant attention, especially during the past few years [1].

Due to the important role of facial expressions in communication, automatic Facial Expression Recognition (FER) has been widely studied, as a result of its wide range of applications, such as crowd analytics, clinical diagnosis, biometrics and many others. An automatic FER system has normally three main stages: (i) pre-processing, specifically to enhance face detection, (ii) feature extraction and (iii) expression recognition. While pre-processing is fundamental for background removal and face alignment, feature extraction is correlated to accuracy of expression analysis and recognition [2].

Facial Expression Recognition systems can be categorized into two main groups: geometric-based and appearance-based methods. The geometric-based methods determine the location of facial components (e.g., mouth, nose, eyebrows, eyes) and extract geometric features from them, measuring distances, curvatures and

[1] Faculdade de Engenharia da Universidade do Porto

other properties. The appearance-based methods start from the assumption that facial expressions result in changes in local texture [2].

Most FER systems focus on recognizing six basic emotions, considered to be universal among cultures: *anger*, *disgust*, *fear*, *happiness*, *sadness* and *surprise*. Still some systems can also include the expressions *neutral* and *contempt* [3].

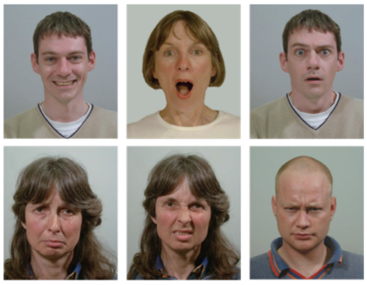


FIGURE 1 - Facial expressions for the six basic emotions. Top row, left to right: happiness, surprise, fear. Bottom row, left to right: sadness, disgust, anger.

Recognizing these emotions under realistic conditions is a challenge, due to variations in head pose, illumination, expression intensity and others.

However, Convolution Neural Networks (CNNs) may be the solution to overcome these problems. In several recent works on FER, CNNs have been successfully used for face detection, feature extraction and recognition, exhibiting a significant performance improvement compared to other approaches. Differences in terms of CNN architecture, pre-processing, training and validation result in different accuracies among various works, as seen in [3].

The aim of this paper is to analyze the different results obtained in accuracy and loss, using different CNN architectures and transfer learning.

II. SOFTWARE ARCHITECTURE

A. Dataset

The dataset used was FER2013, created by Pierre-Luc Carrier and Aaron Courville. This publicly available dataset consists of 35,685 face images. The FER2013 was used in a Kaggle challenge where it was divided into training, validation and test sets with 28,709, 3,589 and 3,589 samples, respectively [3].

To reduce the background noise and for the program to focus on the facial landmarks, all images have a resolution of 48 by 48 pixels. Additionally, all images are grayscale.

Since images are categorized based on the emotion shown in the facial expression (*anger, disgust, fear, happiness, sadness, surprise* and *neutral*), expression labels are provided for all images. Faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. In this dataset, the human accuracy is around 65.5% [4].

The dataset reflects realistic conditions, varying significantly in terms of person age, face pose, expression intensity, illumination and other factors, which is beneficial but also presents a challenge (Fig. 2) [3].

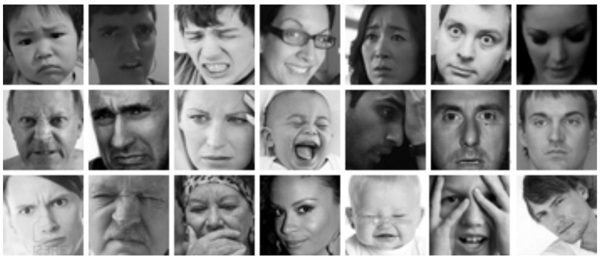


FIGURE 2 - Example images from the FER2013 dataset, illustrating variabilities that occur under realistic conditions [4]. Images in the same column represent identical expressions: anger, disgust, fear, happiness, sadness, surprise and neutral (from left to right).

B. Pre-process and Data Augmentation

Drastic differences in illumination such as histogram distribution, shadows, and so on, can cause noise in the training process and decrease accuracy [9].

As in [3], to mitigate the influence of this illumination, the intensities were centered by subtracting the average pixel intensity of the dataset and dividing by its standard deviation.

Numerous works [3] used data augmentation in FER2013 for training the model. In this work, the techniques focused on were horizontal flipping, zooming the image, and image rotation to augment the data.

C. Convolutional Neural Networks (CNNs)

i. ResNet-50

Residual Network, or ResNet, is a CNN first introduced in 2015 by Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun in their paper “Deep Residual Learning for Image Recognition”.

As the authors of this paper mention, in a multi-layer deep neural network, the training accuracy drops as the number of layers increase, phenomenon known as “vanishing gradients”. In a ResNet architecture, to avoid vanishing gradients, a “shortcut” or a “skip connection” helps to build deeper neural networks by allowing the gradient to be directly backpropagated to earlier layers. Typical ResNet models are implemented with double or triple layer skips that contain nonlinearities (ReLU) and batch normalization in between [5,6].

ResNet uses two main types of blocks, identity and convolution. The identity block is the standard block used and corresponds to the case where the input and output activation have the same dimensions. On the other hand, the convolution block is when the input and output activation dimensions are different from each other [7].

There are different versions of ResNet, including ResNet-18, ResNet-34, ResNet-50, and more. The numbers denote layers, although the architecture remains the same. Hence, ResNet-50 is 50 layers deep [8].

The ResNet-50 model is composed of 5 stages, each with a convolution and an identity block. Each convolution block and identity block have 3 convolution layers each. This version of ResNet has over 23 million trainable parameters [7].

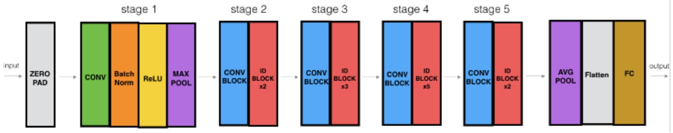


FIGURE 3 - Stages of the ResNet-50

To avoid putting the image input shape as (224, 224, 3), it was chosen to not include a fully-connected layer at the output end of the architecture by setting “include_top = False”, but rather to separately add a custom fully-connected layer, Global Average Pooling and Dense Output Layer, to the ResNet-50 model. Therefore, it was possible to specify the input shape as wanted (48, 48, 3).

Using the ResNet-50 model (based on the Keras Applications library) two experiments were done: one using randomized weights to train (weights = None) and another using weights pretrained on the ImageNet dataset (weights = ‘imagenet’), testing transfer learning.

ii. Custom CNN

Besides the ResNet-50, a custom network with 8 convolutional layers was used, based on the FER2013 Kaggle competition participants. In the following images, it is presented the used architecture as blocks:

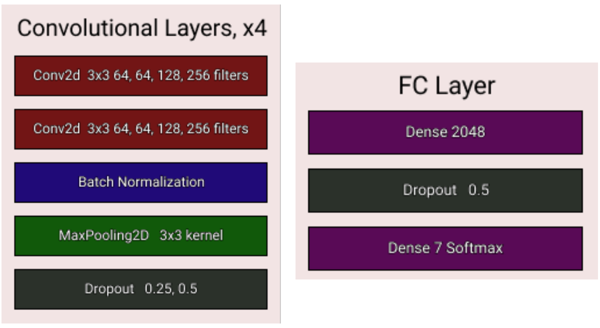


FIGURE 4 – Architecture of the custom CNN

The convolutional layers start with two convolutions, followed by a batch normalization.

The batch normalization is a method commonly used to make artificial neural networks faster and more stable through the normalization of the input layer.

The max pooling layers reduce the dimensions of the input image by combining the outputs of a grid of pixels at one layer into a single neuron in the next layer. In this case, it chooses the highest intensity pixel in one cluster of 3x3 and passes it on to the next layer, and therefore, it is called max-pooling.

The dropout layer is responsible for randomly turning some neurons of the network off, in order to add some noise to the training process. By doing that, the situation, where some neurons have a lot more responsibility than others, is mitigated [10].

Finally, the dense layer, that determines which type of activation function will be used to obtain the final outputs of the model.

As for the loss function, it was used the cross-entropy loss function. This is widely used in classification problems, because it leads to faster training and better generalization [11].

III. RESULTS AND DISCUSSION

As previously mentioned, it was implemented 2 types of CNN architectures. The first, a tailor-made 8 layered CNN, designed to be a simpler model to train, with less parameters but without losing accuracy; and a ResNet-50, a deeper network with a higher computation power. All the models were trained in 150 epochs, which revealed to overfit in some cases, as will be discussed ahead. The ResNet-50 took approximately 1h30 min to run, using Google Colab GPU support.

As also mentioned, the influence of transfer learning was also studied, in order to check the differences in accuracy when the model was based in previous trained weights or not.

Starting with the Keras ResNet-50 model, with random initial weights, after compiling the model and applying the dataset of testing, the results show an accuracy of 0.6294 with a loss of 0.9717, growing as it be can observed in Figure 5.

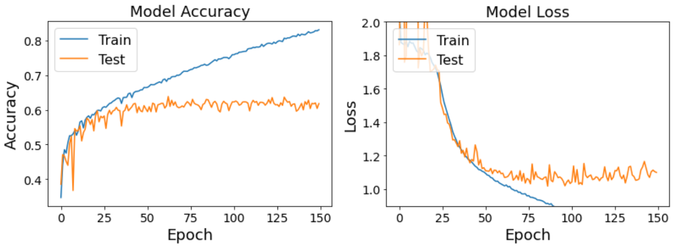


FIGURE 5 – Plot of the evolution of the Accuracy (left) and Loss (right) comparing the train and validation in ResNet-50 without transfer learning.

Around 55 epochs, the validation and train curves diverge. This means that from this point on, probably the model started to overfit to the training data. However, this number of epochs was chosen in order to be comparable to the smaller CNN that took a bit longer to compute, in order to get the same accuracy.

When comparing this model architecture with the transfer learning version, where it was used the

ImageNet weights as basis. This database is a model created to identify random daily objects, and it is highly usable as a launch point for object classification. With this, the evolution and final results of the model are slightly better, showing an accuracy of 0.6372 and loss of 0.9715.

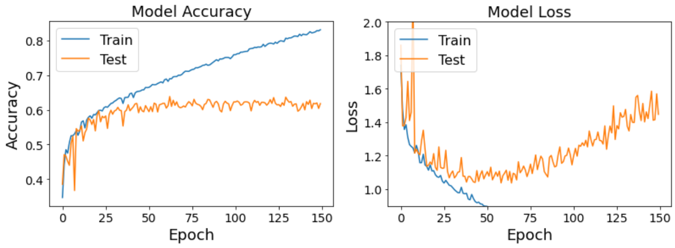
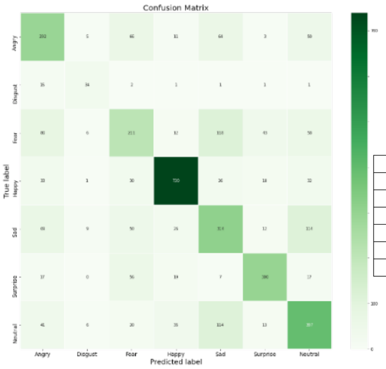


FIGURE 6 – Plot of the evolution of the Accuracy (left) and Loss (right) comparing the train and validation in ResNet-50 with transfer learning (ImageNet).

As the number of epochs was not changed, the tendency for overfitting after the 55 epochs still exists, since the validation loss starts to rise. Nevertheless, if the number of epochs is reduced to 55, on the training phase, ending the overfit, the overall accuracy in the test dataset, does not change significantly.

A final good result measurement is through the confusion matrix where, from the test set, each emotion has the comparison between the correct and incorrect predictions. This model could not, for example, differentiate very well *fear* from *surprise*, however the *happiness* emotion was very well achieved and identified. However, each emotion does not have the same number of images and it would seem that *disgust* was poorly identified. But, by normalizing the rate of correct predictions per the number of pictures of each emotion, a more realistic visual score appears.

With the normalized results, for the *disgust* emotion, it shows a rate of ~62% correct predictions (Fig. 7).



	ResNet-50
	% of Correct Predictions per Emotion
Angry	59.47%
Disgust	61.82%
Fear	39.96%
Happy	84.07%
Sad	52.86%
Surprise	72.12%
Neutral	63.42%

Color Code	
0-20%	🟩
20-40%	🟩
40-60%	🟩
60-80%	🟩
80-100%	🟩

FIGURE 7 – In the left the confusion matrix of the ImageNet transfer learning ResNet-50 model. In the right, a confusion vector, normalizing the # of correct predictions per emotion.

Moving on to the custom CNN, this was not trained based on the ImageNet weights, since the layers wouldn't match, so it is not viable to compare this model to the one with transfer learning. However, besides the 8 layered architecture and the 150 epochs needed to train, the results became really promising with an accuracy of 0.6492 and loss 0.9407.

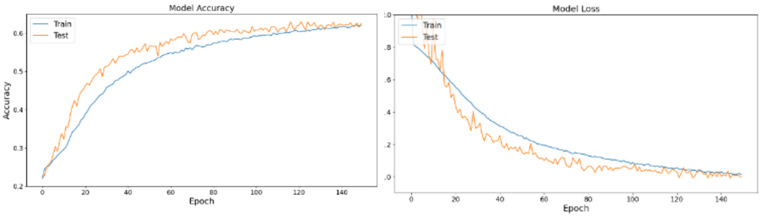
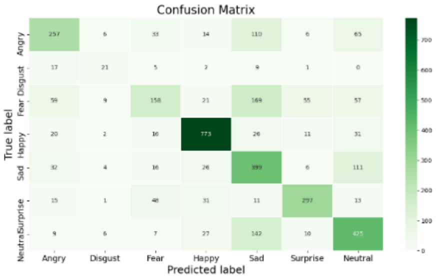


FIGURE 8 – Plot of the evolution of the Accuracy (left) and Loss (right) comparing the train and validation in 8 layered CNN.

Since the CNN took advantage of all the training steps, the test efficiency ended up being slightly better, as seen in the confusion matrix and vector in Figure 9.



	8 L. CNN
	% of Correct Predictions per Emotion
Angry	52.34%
Disgust	38.18%
Fear	29.92%
Happy	87.94%
Sad	67.17%
Surprise	71.39%
Neutral	67.89%

FIGURE 9 – In the left the confusion matrix of the 8 L. CNN model. In the right, a confusion vector, normalizing the # of correct predictions per emotion.

Here in this model, even if the overall accuracy is higher, due to the numerous correct *happy* predictions, the emotion classification is not as perfect as in the ResNet-50. Visually, the darker cells of the matrix are not only focused in the diagonal, being the most clear case the *fear* emotion, where it is most misclassified as *sadness*.

Finally, with these results some comparisons with the literature can be made. For instance, in [3], another 8-layer CNN was examined. Even though the architecture is not the same, is interesting to see how CNN with the same depth can have different results. In [3], the architecture was the type PCCPCCPCFFF and the final values end up being 9% less accurate in this project. Regarding the ResNet-50 approaches, in [8], this method is also applied to FER2013 getting an accuracy of 65.1%, slightly higher than our models. This could be due to the different pre-process of the images as well as other transfer learning databases used to start the models.

IV. CONCLUSION

In conclusion, in this project, it was developed and tested 2 model architectures, one based on the Keras library, the ResNet-50 and another implemented specifically to this dataset. Besides that, the influence of transfer learning, using the ImageNet, was also compared. All in all, the accuracy of all the models experimented rounds up to 65% in the test set, making the best predictions in the *happy, sad* and *surprise* emotions. As a future work, a different pre-process of the images could be realized and the extraction of facial features. Transfer learning from other databases could also be tested, like VGG-Face.

V. REFERENCES

[1] Ekman, P. (1999). Facial expressions. Handbook of cognition and emotion

[2] Ferreira, P. M., Marques, F., Cardoso, J. S., & Rebelo, A. (2018). Physiological inspired deep neural networks for emotion recognition.

[3] Pramerdorfer, C., & Kampel, M. (2016). Facial expression recognition using convolutional neural networks: state of the art. arXiv preprint arXiv:1612.02903.

[4] Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, et al (2015). Challenges in representation learning: A report on three machine learning contests.

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition.

[6] Alom, M. Z., Taha, T. M., Yakopcic, et al (2018). The history began from alexnet: A comprehensive survey on deep learning approaches.

[7] Rezende, E., Ruppert, G., et al, 2017 Malicious software classification using transfer learning of resnet-50 deep neural network.

[8] Savoiu, A., & Wong, J. (2017). Recognizing facial expressions using deep learning.

[9] Nilanjan Dey, (2019), Uneven illumination correction of digital images: A survey of the state-of-the-art

[10] Srivastava, Nitish & Hinton, et al. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

[11] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Berlin: SpringerVerlag, page:235