Journal of Theoretical and Applied Information Technology

15th December 2023. Vol.101. No 23
© 2023 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DETECTION OF FACIAL MICRO-EXPRESSIONS USING CNN

F. TADEO SÁNCHEZ GARCÍA [1] , B. LÓPEZ LIN [2], RODOLFO ROMERO-HERRERA [3]

[1]IPN, ESCOM, Department of Computer science and engineering, Ciudad de México

[2]IPN. ESCOM, Department of Computer science and engineering, Ciudad de México

[2]IPN. ESCOM, Department of Postgraduate, Ciudad de México

E-mail: [1] blopezl1902@alumno.ipn.mx, [2]fsanchezg1602@alumno.ipn.mx, [3]rromeroh@ipn.mx

ABSTRACT

Brief, involuntary micro-facial expressions represent a window into a person's hidden or repressed emotions. The ability to analyze and detect them can have a significant impact in various fields that require an understanding of human behavior. However, the process of detecting micro-expressions poses significant challenges, such as the implementation of the detection method or the generation of extensive and quality data. This article develops a Machine Learning model with a convolutional neural network; It is compared with other existing models of micro expressions in the prediction of one of 7 human emotions, to recognize micro expressions and predict one of 7 emotions.

Keywords: *Deep Learning, Human Emotions, Micro-Expressions, Convolutional Neural Networks.*

1. INTRODUCTION

Facial micro expressions are brief, involuntary expressions that reveal a person's hidden or repressed emotions[1]. The analysis and detection of these micro-expressions have various applications in the analysis of human behavior [1] [2]. It gives us clues to determine if a person is lying; Improves communication and mutual collaboration by knowing the emotions of the participants; assists in the analysis of suspicious behavior or attempted deception; and, works as a means of support to complement evaluation techniques, such as clinical interviews.

Convolutional neural networks were used from a Kaggle Data Set [3], which does not affect the training of the model due to its size and quality. Therefore, we will seek to predict the 7 emotions. The variability of faces can affect the quality, as there may not be enough faces for the model. Since micro expressions occur in fractions of a second, several samples are needed; In addition, the possible quality of the camera and the conditions where the program is executed are added, for example, the computational power required to process the data. In addition to the above, an erroneous prediction will be a problem if it is used in a real context, as it could generate false positives or false negatives.

Then the objective is to generate a Deep Learning model capable of detecting characteristics of micro facial expressions, of one of the 7 human emotions that are revealed at the time of carrying out the detection process.

The objective must meet an accuracy greater than 80%, during training and testing, since the system will be used in security situations, so processing time is also critical. It should also be considered that biometrics were not used, but micro-expression recognition using CNN neural networks, so the system is independent of the position of the face and the separation of the camera.

2. RELATED WORKS

In “Toward Bridging Micro Expressions From Different Domains” [6]; The authors propose an effective method consisting of auxiliary set selection model (ASSM) and transductive transfer regression model (TTRM). The method uses transfer learning to reduce the gap between the source and target domains and improve the performance of micro-expression recognition. A new approach that combines two types of cameras and effective facial points to identify human emotions is presented in [7]. The approach uses a deep neural network and a random forest regression algorithm to classify emotions. P. Flotho, et al [8] propose a new strategy to magnify motion in videos with subtle and imperceptible facial movements. The method integrates facial landmark information and an approach to decompose facial movements in an unsupervised manner using sparse PCA. In addition, a method based on deep learning is used that is retrained with micro-expressions. In [9] the authors present a general framework for creating original and aesthetic images from textual or visual input. They use a generator network that learns to synthesize high-quality images and a discriminator network that evaluates the plausibility and creativity of the generated images [10]. Examines the time-frequency characteristics of an EEG associated with the recognition of facial expressions with different durations. It shows that there are significant differences between expressions with a duration of less than 200 ms and expressions with a duration greater than 200 ms; the article suggests that the recognition of micro-expressions is based on different mechanisms. P. Flotho, C. Heiß, G. Steidl, and D. J. Strauss [11] propose a method for the identification of micro-expressions based on the use of different color spaces and machine learning techniques, evaluating its performance on two public data sets. The article "Micro-expression recognition with small sample size by transferring long-term convolutional neural network" [12]: presents a method for the recognition of micro-expressions with small sample size, using a long-term convolutional neural network that is trained with macro expression data and transferred to micro-expression data. Y.-H. Oh, J. See, A. C. L. Ngo, R. C. -w. Phan, and V. M. Baskaran [13]: offer a comprehensive review of the analysis of facial micro-expressions, covering theoretical aspects, existing databases, proposed methods, and future challenges [14]. Summarizes the results of an international competition that sought to promote the progress of methodologies for the recognition and detection of micro-expressions. It is also feasible to employ a method that uses Principal Component Analysis (PCA) to reduce the data size and obtain more relevant patterns from facial images and then use SVM to classify the micro-expressions into four emotions [15]. S. P. Teja Reddy, S. Teja Karri, S. R. Dubey, and S. Mukherjee present the article "Micro-expression recognition using 3D spatiotemporal features" [16], where they show a new method that uses a bidirectional recurrent neural network (BiRNN) to capture the temporal information from micro expressions, and a convolutional neural network (CNN) to extract spatial information from facial images. Finally, [17] shows two methods for detecting micro-expressions based on the local temporal pattern (LTP) and the local binary pattern (LBP). The authors compare the results of the two methods on the SAMM and CAS (ME)2 databases and propose a performance evaluation metric.

The related works demonstrate the scientific interest in solving microexpression recognition problems. Although they do not focus on the recognition of affective states in microexpressions by evaluating different models, but on the techniques that allow recognizing the existence of faces with expressions. Due to the nature of future applications of the system, recognition of affective facial expressions is important to take vital security measures that address emotion recognition in microexpressions. The present work focuses on recognizing affective facial expressions at a distance of 10 meters, regardless of the position of the face.

3. PROJECT DEVELOPMENT

This section presents the implementation, analysis, and detection of micro facial expressions. To do this, in the initial stage, the seven micro expressions classified by psychologist Paul Ekman are considered: anger, disgust, contempt, happiness, fear, sadness, and surprise [18]. This can be seen in fractions of a second through the non-verbal behavior of people, such as the involuntary movements of the muscles of the face that respond to an emotion in certain situations. Analyzing these micro-expressions is quite useful for detecting lies, knowing a person's emotions more accurately, and other applications.

Once the micro-expressions are classified, a set of images obtained from a dataset is loaded (Figure 1) and transformations are performed on them, such as cropping, flipping, blurring, and rotation. The original images are then combined and placed into a single data set, which is subsequently prepared to analyze and train machine learning models.

Figure 1. Image sets are used for training and validation.

Once the batches of images are loaded, it is necessary to classify them concerning micro-expressions. For this, the "BaseClasificacionImagenes" class was created, which is used to create image classification models where calculations are carried out using tensors. In addition, it allows its execution on the GPU to speed up calculations. A function to calculate the accuracy of the model predictions is also included.

It is necessary to generate a Deep Learning model using a convolutional neural network (CNN) for image classification. CNN can have hundreds of layers, and each of them detects different particularities through filters applied to the training images; each convolved output is the input to the next layer. Filters start with characteristics such as brightness and edges and build into features that uniquely define the object. Figure 2 illustrates a CNN, made up of an input layer, an output layer, and several hidden layers.

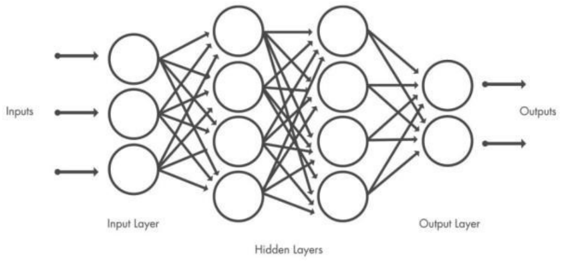


Figure 2. Example of a CNN

The most common layers are convolution, activation (ReLU), and pooling, which are repeated in all layers. See Figure 3.

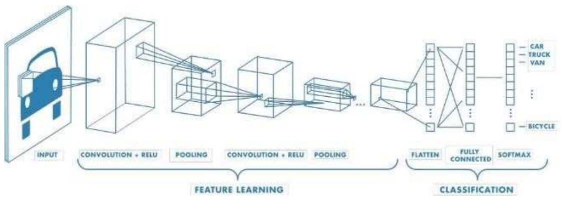


Figure 3. CNN network with multiple convolutional layers.

3.1 Shared weights and biases

In all the neurons of a CNN there are the same weights and shared bias values; meaning that the same image features are detected, so the network is tolerant to the position of the object within an image.

3.2 Classification layers

Once the characteristics are identified, we proceed to classify. The penultimate layer is fully coupled and forms K classes based on probabilities. The final layer uses a classification output layer [19].

The ModeloMERCNN class is defined, which inherits from the BaseClassificationImagenes class. The ModelMERCNN class implements a CNN model for image classification.

In the "init" method, the model is initialized, and the layers are configured. The base model used is ResNet-18, a pre-trained neural network. The final fully connected layer (self.base_model.fc) is modified to be an identity layer, which means that no transformation will be applied to the outputs of the convolutional network.

Subsequently, a layer sequence self.feature_reduction is defined which is in a linear layer followed by the ReLU activation function. This sequence reduces the dimensionality of the features extracted by the base model to a feature space of size 512. Then, another sequence of self-regularization layers is defined, which consists of a linear layer that produces the classified outputs. The forward method defines the forward flow of the network; t takes a batch of input data and passes the data through the base model, subsequently through the feature reduction layer, and finally through the regularization layer. Sorted outputs are returned as a result.

It is important to emphasize the use of pre-trained resnet18 (Figure 4) since it is a convolutional neural network architecture with 18 layers that uses residual connections to train deep networks in image classification tasks. In addition, tests were carried out with the neural network models, which were resnet 18, and 2 of our development.

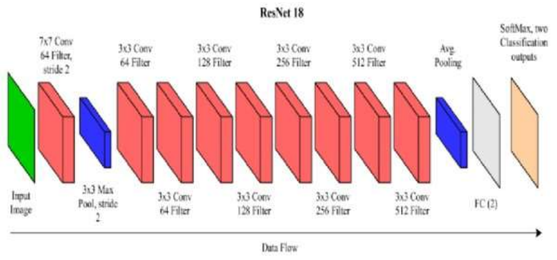


Figure 4. Resnet18 operation

3.3 CNN Model 1

CNN Model 1 is composed of 13 layers in total. It starts with two convolutional layers with ReLU activation functions to extract initial features from the micro-expression images. These convolutional layers are followed by max-pooling layers that reduce the spatial dimensionality of the obtained features and capture the most relevant features at a lower resolution.

Two additional convolutional layers are then added to capture more complex and abstract features of the micro-expressions. These convolutional layers are also combined with ReLU activation functions. Then, another layer of max pooling is applied to further reduce the dimensionality of the particularities.

After the pooling layers, the features are converted into a one-dimensional vector. This vector is passed through three fully coupled layers with ReLU activation functions, which transform the extracted features into a classification output. The final layer uses a linear activation function to produce the output of the last stage of the model, which represents the degree of membership in each micro-expression class.

CNN Model 1 is trained using a dataset, where images of micro-expressions are classified into different emotional classes. During training, the model adjusts the layer weights to minimize the loss function and maximize classification accuracy. Once trained, the model can be used to classify new unlabeled micro-expressions and assign them to the corresponding class.

3.4 CNN Model 2

CNN Model 2 is like CNN Model 1 in terms of its basic architecture. However, some modifications have been made to capture even more information from the micro-expressions and regularize the model.

Like CNN Model 1, CNN Model 2 starts with convolutional layers and ReLU activation functions to extract initial features from the micro-expression images. Then, max-pooling layers are applied to reduce the dimensionality of the extracted features.

The main difference in CNN Model 2 is the addition of convolutional layers. These convolutional layers are added to capture more details and complex patterns in the micro-expressions. Each convolutional layer is followed by a ReLU activation function.

Additionally, dropout layers have been introduced after the linear layers with 1024 and 512 units, respectively. These dropout layers with a rate of 0.5 help to regularize the model and reduce overfitting during training. Dropout randomly “turns off” a percentage of the units in each training step, which prevents the model from becoming overly dependent on specific features and improves its generalization ability.

When training CNN Model 2, an annotated dataset of micro expressions is applied, where images of micro expressions are associated with their respective emotional classes. During training, weights are adjusted, and distinctive patterns and characteristics of micro-expressions are recognized. Once trained, the model can be used to classify new micro-expressions and assign them to corresponding classes.

3.5 RESNET18 Model

The ResNet18 Model is based on the ResNet (Residual Networks) architecture, which is known for its ability to train deeper convolutional neural networks effectively. ResNet18 consists of 18 layers in total, including convolutional, pooling, and a fully connected layer.

The distinctive feature of ResNet is the use of residual connections, which allow the flow of information to be routed directly through the network without passing through multiple layers. This helps mitigate the performance degradation issue that can occur with increasing network depth.

In the case of ResNet18, the model starts with a convolutional layer and a pooling layer to extract initial features from the micro-expression images. Then, residual blocks are stacked, which are composed of two convolutional layers and one residual connection. These blocks allow features to persist and flow directly through the network, making it easier to learn more complex and abstract features.

After the residual blocks, pooling layers and a fully connected layer are used to transform the extracted features into a classification output.

The ResNet18 Model is trained using an annotated micro-expression dataset, where micro-expression images are classified into different emotional classes. Once trained, the model can classify new unlabeled micro-expressions and assign them to the corresponding class.

3.6 Model Comparison

ResNet18:

- Architecture: Convolutional neural network with 18 layers and residual connections.
- Convolutional layers: 16 convolutional layers divided into 4 blocks.
- Pooling Layers: Pooling layers (MaxPool2d) after each block.
- Fully connected layers: 2 linear layers (fully connected) at the end.
- Dropout layers: No dropout layers are used in ResNet18.

CNN Model 1:

- Architecture: Custom convolutional neural network.
- Convolutional layers: 9 convolutional layers divided into 3 blocks.
- Pooling Layers: Pooling layers (MaxPool2d) after each block.
- Fully connected layers: 3 linear layers (fully connected) at the end.
- Dropout layers: No dropout layers are used in the CNN 1 model.

CNN Model 2:

- Architecture: Custom convolutional neural network.
- Convolutional layers: 12 convolutional layers divided into 4 blocks.
- Pooling Layers: Pooling layers (MaxPool2d) after each block.
- Fully connected layers: 3 linear layers (fully connected) at the end.
- Dropout layers: Two dropout layers with a dropout rate of 0.5 are used after the linear layers with 1024 and 512 units, respectively, in the CNN model 2.

3.6 Device Selection

Once the code is fragmented, functions and a class are provided that facilitate device management, allowing you to select a GPU if available and move data tensors to the selected device.

The get_default_device() function checks if a GPU is available. If the GPU is available, a device is returned, otherwise a device of type 'CPU' is returned. The selected_device(data, device) function is used to move data tensors to the selected device. If the data is a list or tuple, the function is applied recursively to each element of the list or tuple. PyTorch's to () method is then used to move each tensor to the specified device.

The “DeviceDataLoader” class is used to prepare an existing data loader to move data to the selected device. It receives a dataloader and a device as arguments in its constructor. The class implements methods that allow the data loader to be iterable and return the batches of data moved to the selected device.

Once the device is selected image class predictions are made. This part of the code defines the “predict_image” function that is used to make class predictions on an image using the previously trained model. See Figure 5.

```
Etiqueta Original: happiness , Etiqueta Predicción: disgust
```

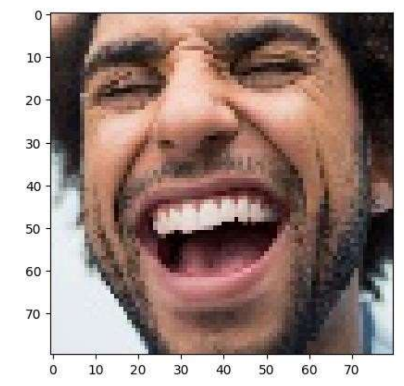


Figure 5. Results when executing the code. Shows the prediction on the test set.

At the last stage of development, the code is already capable of giving a prediction of an image captured by the camera of the computer where the program is running. This code has several functions and steps:

1. Import the necessary modules.
2. Loads a trained model called ModelMERCNN and saves it in the model variable.
3. Load the trained model weights from the “MicroModel.pth” file using model.load_state_dict().
4. Define a function called predict_image that performs the prediction of an image using the loaded model. The function applies transformations to the image, passes it through the model, and returns the predicted class.
5. Define a function called detect_faces that uses the Haar cascade classifier to

detect faces in an image. The transform function converts the image to grayscale and uses the classifier to detect the coordinates of the faces.

6. Defines a function called capture_image that you use to capture an image from the camera. The function displays the frame in a window and waits for the "Space" key to be pressed to capture the image.
7. Defines the enhance_image function that enhances the image quality and crops the detected face using the coordinates provided by the detect_faces function.
8. Capture an image using the capture_image() function, enhance it using the enhance_image() function, and display the resulting image using matplotlib.pyplot.imshow().
9. Perform the prediction on the image using the predict_image() function and display the predicted class in the console.

4. RESULTS AND DISCUSSION

To facilitate the analysis and comparison between the models used, we created a table and recorded the "accuracy" results of the training and testing. In this, RESNET18 is a preset model, CNN1 and CNN2 are self-made models or custom models. With this table, we could determine, in practical terms, which model would be most convenient to use to detect micro-expressions of human emotions. Time is included, so the row that refers to the generation order of the models was added, just like a list of finishing positions in a race. It has been mentioned that the best model is sought in practical terms, not only in terms of scoring but also in terms of time. See table 1.

Table 1. Comparison of training, testing, and generation order accuracy between the RESNET18, CNN1, and CNN2 models.

	RESNET 18	CNN1	CNN2
Train	94.20%	91.20 %	90.05 %
Test	80.81%	82.53 %	81.83 %
Generation order	#3	#1	#2

The previous decision was made given that the other models required much less time to generate, so such a loss of time that could be used to improve parameters and perform more tests for CNN1 and CNN2 was not permissible.

On the other hand, the next discard will be the analysis of CNN1 and CNN2, since they are personalized models. In this case, it is visible that CNN1 is superior in every way to CNN2, both in score and generation, and this is the one that was generated the fastest.

Finally, deciding between RESNET18 and CNN1 seemed to be complicated because the former outperformed the latter in training accuracy, but the opposite happened in the case of test accuracy and generation order, and the latter meant that its computational cost was lower as it had been generated earlier. Therefore, it was decided to give greater relevance to these last two aspects, since they end up compensating for the difference with training, which means that the model that will be used to detect human emotions with facial micro-expressions will be the personalized model CNN1.

Additionally, in Figure 6, we can see the classification report of the CNN1 model, where it can be seen that good predictions are made in general, except for the "fear" category, which is close to 50% random; Furthermore, and due to this imbalance, a bias towards the other classes will likely be generated, and that it will be quite difficult to achieve the detection of this label with little effort and attempts. However, if it is considered that the data set used did not contain enough quality or quantity samples to balance the training, a result like the one shown is acceptable.

Extensive evaluation on an independent test set using metrics such as classification reports and confusion matrices provides a quantitative evaluation of the model's performance in facial emotion classification. These metrics provide detailed information about accuracy, allowing you to understand the model's ability to discriminate between different emotions.

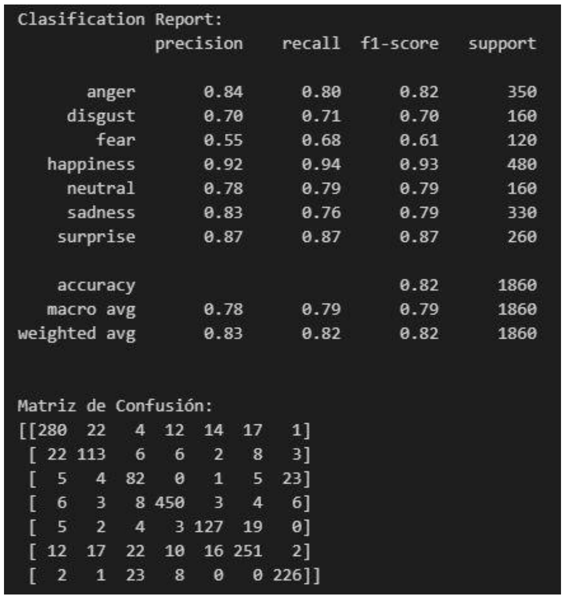
Table 2 explains the similarities between the evaluation of metrics explained above, showing us their formulas.

Table 2. Comparison table of evaluation metrics (accuracy, precision, recall, and f-measure)

Metrics	Formula	Meaning of variables
Accuracy	$\frac{tp + tn}{tp + tn + fp + fn}$	fp=Falsos positives
Precision	$\frac{tp}{tp + fp}$	tn=true negatives
Recall	$\frac{tp}{tp + fn}$	tp=True positives
F1-Score	$2 * \frac{presición * recall}{precision + recall}$	
Confusion Matrix	Na	

The Confusion Matrix, a very useful tool in measuring the performance of Machine Learning models, helps us define which is the best classifier, especially if we do not have balanced sets. However, the category with more data may classify better, and it will not do as well with the other category. And if we then validate this model and calculate its accuracy, we will not have a precise estimate of its performance, so the parameter can mask the results, because the proportions are not the same. For this reason, the confusion matrix is used.

As we saw in the previous example, it is not enough to simply estimate the number of correct answers, since we must be able to differentiate performance between classes; and the confusion matrix allows us to achieve this. Figure 6 shows us both the successes and failures through the confusion matrix for the affective states.



```
Clasification Report:
              precision    recall  f1-score   support

       anger       0.84      0.80      0.82       350
     disgust       0.70      0.71      0.70       160
        fear       0.55      0.68      0.61       120
   happiness       0.92      0.94      0.93       480
     neutral       0.78      0.79      0.79       160
     sadness       0.83      0.76      0.79       330
    surprise       0.87      0.87      0.87       260

    accuracy                           0.82      1860
   macro avg       0.78      0.79      0.79      1860
weighted avg       0.83      0.82      0.82      1860


Matriz de Confusión:
[[280  22   4  12  14  17   1]
 [ 22 113   6   6   2   8   3]
 [  5   4  82   0   1   5  23]
 [  6   3   8 450   3   4   6]
 [  5   2   4   3 127  19   0]
 [ 12  17  22  10  16 251   2]
 [  2   1  23   8   0   0 226]]
```

Figure 6. CNN1 model classification report.

4.1 Running model test

To demonstrate that the model correctly detects and predicts the human emotions established in the data set, the following images demonstrate this operation:

1. Sadness. In Figure 7, the recognition of the emotion of sadness. This is easy to achieve since its precision is high.

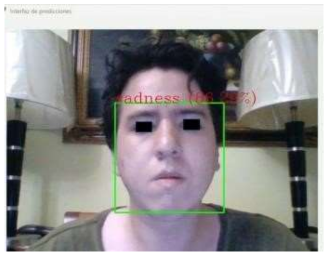


Figure 7. Detection of the emotion "sadness".

2. Happiness. In Figure 8, the emotion of happiness is detected. This is also easy to achieve since its precision is the highest.

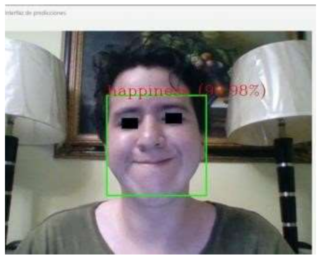


Figure 8. Detection of the emotion "happiness".

3. Neutral. Figure 9 shows the detection of the neutrality emotion. This is slightly complicated to achieve since it is the third category with the worst accuracy.

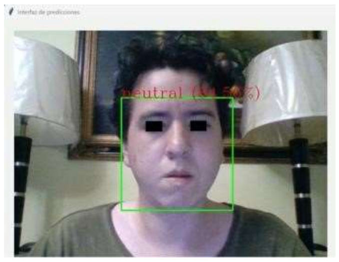


Figure 9. Detection of "neutral" emotion.

4. Angry. Figure 10 shows the emotion of anger or annoyance. This is easy to achieve since its precision is somewhere in the middle.

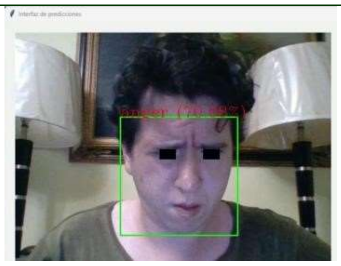


Figure 10. Detection of the "angry" emotion.

5. Surprise. Figure 11 shows the emotion of surprise. It is like "angry", it has a medium difficulty.

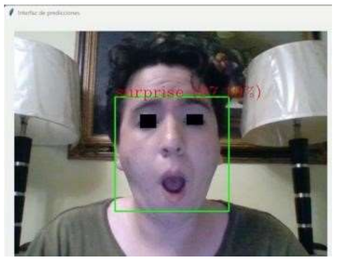


Figure 11. Detection of the "surprise" emotion.

6. Dislike Figure 12 shows the emotion of disgust. This is moderately difficult to achieve since its accuracy is the second worst in the classification report.

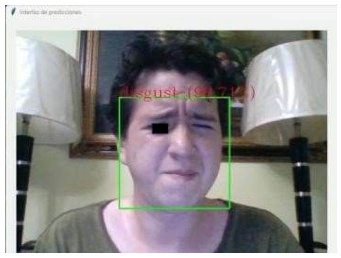


Figure 12. Detection of the emotion "disgust".

7. Fear. Figure 13 shows the emotion of fear. This is the most difficult to achieve by far, because it has the lowest precision in the classification report and, however, it is possible to achieve.

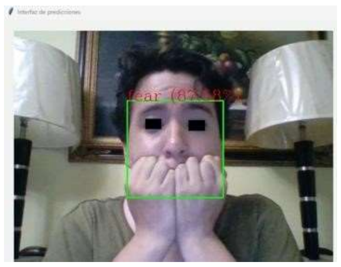


Figure 13. Detection of the "fear" emotion.

Finally, with what we have seen in the previous screenshots and the classification report in the previous section, we can affirm and demonstrate that, even though some categories of human emotion have relatively low precision, the program manages to carry out its function of detection and prediction.

The system focuses especially (unlike the references) on seven affective facial expressions, with faces located a maximum of 10 meters away regardless of the position in which the face is located; in accordance with the stated objectives. For this, existing models and two of our own were used that were better in the tests carried out.

Although the accuracy during training and testing exceeded 80%, during real-world implementation, according to Figure 6, problems arise with the facial expression of fear. Situation that will be addressed as future work through the use of thermal or infrared cameras and adapting various sets of images.

5. CONCLUSIONS

The presented work offers a comprehensive and versatile solution to address the challenge of facial expression classification in micro-expression images. By experimenting with different neural network models, including two custom models and one pre-trained model (RESNET18), it is possible to explore and compare different approaches that improve the accuracy and generalization capacity of the model.

Applying transformations to the data set, such as random cropping, horizontal flips, and Gaussian blurs, contributes to increasing the diversity of the data and improving the model's ability to recognize and classify different emotional expressions. These transformations allow us to simulate various image capture conditions and improve the robustness of the model to possible variations in the input data.

The code includes functions to make predictions on individual images and visualize the training process through precision and loss plots. These tools make it possible to understand and analyze the performance of the model at different stages of the process, facilitating the detection of possible overfitting or underfitting problems, and providing an overview of the model's progress throughout the training epochs.

In summary, a robust and flexible solution for facial expression classification in micro-expression images is provided. The obtained results, evaluated both qualitatively and quantitatively, provide a deep understanding of the performance and effectiveness of the models in the task of facial emotion classification. These advances have the potential to have significant applications in fields such as psychology, emotional research, human-computer interaction, and video analysis.

The results of the affective facial expression of fear are below those expected, this is because the set of images did not contain enough illustrative images of this facial expression, so a data set that provides greater diversity and number is sought. of records, or the creation of your own data set. For this purpose, future work proposes the use of thermal or infrared cameras and the implementation of convolutional neural networks with transformers.

ACKNOWLEDGE

The authors acknowledge the support received for the completion of this project from the IPN (Instituto Politécnico Nacional).

REFERENCES:

[1] C. Wang, M. Peng, T. Bi, and T. Chen, “Micro-attention for micro-expression recognition,” *Neurocomputing*, vol. 410, pp. 354–362, Oct. 2020, doi: 10.1016/j.neucom.2020.06.005.

[2] Y. Wang *et al.*, “Micro Expression Recognition via DualStream Spatiotemporal Attention Network,” *Journal of Healthcare Engineering*, vol. 2021, pp. 1–10, Aug. 2021, doi: 10.1155/2021/7799100.

[3] K. M. Irfan, “Micro_Expressions,” *Kaggle*, Apr. 25, 2022. https://www.kaggle.com/datasets/kmirfan/microexpressions

URL Date Stamp Time Stamp GMT and dd/mm/yyyy

[4]“PyTorch documentation — PyTorch 2.0 documentation.” https://pytorch.org/docs/stable/index.html.

[5]“torchvision — Torchvision 0.15 documentation.” https://pytorch.org/vision/stable/index.html

[6] Z. Yuan, Z. Wenming, C. Zhen, Z. Guoying, H. Bin, “Toward Bridging Microexpressions From Different Domains”, Ieee Transactions On Cybernetics, Vol. 50, No. 12, 2020.

[7] P. Y. Afonichkina and A. D. Frunze, "Comparative Analysis of Approaches to Developing Emotions with the Infrared Camera," 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), St. Petersburg, Moscow, Russia, 2021, pp. 177179, doi: 10.1109/ElConRus51938.2021.9396463.

[8] P. Flotho, C. Heiß, G. Steidl and D. J. Strauss, "Lagrangian Motion Magnification with Landmark-Prior and Sparse PCA for Facial Microexpressions and Micromovements," 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), Glasgow, Scotland, United Kingdom, 2022, pp. 2215-2218, doi: 10.1109/EMBC48229.2022.9871549.

[9] F. Xu, J. Zhang and J. Z. Wang, "Microexpression Identification and Categorization Using a Facial Dynamics Map," in IEEE Transactions on Affective Computing, vol. 8, no. 2, pp. 254-267, 1 April-June 2017, doi: 10.1109/TAFFC.2016.2518162.

[10] X. Shen and H. Sui, "The time frequency characteristics of EEG activities while recognizing microexpressions," 2016 IEEE Biomedical Circuits and Systems Conference (BioCAS), Shanghai, China, 2016, pp. 180-183, doi: 10.1109/BioCAS.2016.7833761.

[11] S. -J. Wang et al., "Micro-Expression Recognition Using Color Spaces," in IEEE Transactions on Image Processing, vol. 24, no. 12, pp. 6034-6047, Dec. 2015, doi: 10.1109/TIP.2015.2496314.

[12] S. Wang et al., “Micro-expression recognition with small sample size by transferring long-term convolutional neural network,” Neurocomputing, vol. 312, pp. 251–262, Oct. 2018, doi: 10.1016/j.neucom.2018.05.107.

[13] Y.-H. Oh, J. See, A. C. L. Ngo, R. C. -w. Phan, and V. M. Baskaran, “A Survey of Automatic Facial MicroExpression Analysis: Databases, Methods, and Challenges,” Frontiers in Psychology, vol. 9, Jul. 2018, doi: 10.3389/fpsyg.2018.01128.

[14] M. H. Yap, J. See, X. Hong and S. -J. Wang, "Facial Micro-Expressions Grand Challenge 2018 Summary," 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), Xi'an, China, 2018, pp. 675-678, doi: 10.1109/FG.2018.00106.

[15] I. Fibriani, R. Mardiyanto and M. H. Purnomo, "Detection of Kinship through Microexpression Using Colour Features and Extreme Learning Machine," 2021 International Seminar on

Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, 2021, pp. 331336, doi: 10.1109/ISITIA52817.2021.9502247.
[16] S. P. Teja Reddy, S. Teja Karri, S. R. Dubey and S. Mukherjee, "Spontaneous Facial Micro-Expression Recognition using 3D Spatiotemporal Convolutional Neural Networks," 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 2019, pp. 1-8, doi: 10.1109/IJCNN.2019.8852419.
[17] J. Li, C. Soladie, R. Seguier, S. Wang, and M. H. Yap, Spotting Micro-Expressions on Long Videos Sequences. 2019. doi: 10.1109/fg.2019.8756626.
[18] Mallitasig Arellano, Henry Wladimir. "Paul Ekman y las microexpresiones faciales de las emociones." (2018).
[19] Sánchez Martínez, Marina. "Detección de personas mediante técnicas de aprendizaje automático: SVM y CNN." (2018).