3DFER - Revised Report

Gerardo Festa
Alessandra Parziale
g.festa22@studenti.unisa.it
a.parziale8@studenti.unisa.it
https://github.com/GerardoFesta/3DFER_SE4AI

1 CONTEXT

Facial Emotional Recognition is an emerging field of research that uses artificial intelligence algorithms to detect and interpret human emotions. The vast majority of the research has focused on 2D images. At the same time, Emotion Recognition has been explored by also using various sources other than facial images, such as data from wearables, speech, or EEG. The 3D field faces some limitations due to the lack of available public datasets, and even the 2D ones that can be found have their drawbacks.

1.1 Datasets availability

There are not many 3D datasets available, and most of them are very small, such as Rap3D[7], which makes them not very suitable for training Artificial Intelligence models. Furthermore, most of them are captured in a lab environment since the capture requires specific equipment (for example, structured light in Bosphorus3D [1] and IR in Rap3D [7]). This means, obviously, that the Emotion Recognition System would likely not work outside of the lab environment and without that equipment. When it comes to 2D datasets, they are more widespread than 3D ones, but most of them are composed of very small grayscale images/video, such as in FER, FER2013, and CK+ datasets [10] [4].
This could lead to a bias toward the type of capture, thus making the model work in a real-world environment is a challenge by itself. The best ones, by means of best-suited for the project objectives, are those named "In the wild", like Aff-Wild2 [3] and AffectNet [6], which consist of images taken in various situations, often scraped from social media. Despite being promising, they are hard to access (we are waiting for the owners to get back to us). Please note that by availability, we specifically refer to emotion-labeled data; there are many 3D and 2D facial datasets, but most of them do not provide manual emotion annotations. We must also consider that datasets do not always use the same emotional representation. While we are always talking about a classification problem, there are multiple theories about the specter of emotions; the most used is one with 7 different classes, namely "Angry", "Disgust", "Fear", "Happy", "Sad", "Surprise", "Neutral". Some datasets use more, some use less and some use completely different classes.

Authors' address: Gerardo Festa, g.festa22@studenti.unisa.it; Alessandra Parziale, a.parziale8@studenti.unisa.it;, https://github.com/GerardoFesta/3DFER_SE4AI.

1.2 3D reconstruction

As explained before, the availability of 3D data is pretty limited. To simulate how a model could work on 3D images, we could think of reconstructing a 3D face from a 2D facial image. This arises an issue of the 3D data: there is not just one 3D representation or even a single method for treating 3D data. 3D face reconstruction is a very active research topic, that requires a vast knowledge of image processing and properties. The most problem-suitable representations and transformations that can be done are:

- **3D Landmarks**: the reconstruction works by detecting multiple face landmarks. This technique is often used for AR filters, such as social media filters [5]. By also working on smartphones, this method is usually light and fast and is the base of many more complex 3D reconstructions. It is achievable via multiple different solutions, and it has been proved that 2D landmarks can work quite efficiently when it comes to expression recognition [8]. Shown in [Figure 2], one of the most widespread landmark recognition is MediaPipe by Google [5], which is able to recognize 478 3D facial landmarks, normalized for the dimension of the face.
- **3D Voxels**: a voxel is the equivalent of a pixel in a 3D space. This means that instead of squares, we are dealing with cubes. It is possible to transform a point cloud (generated, for example, by a 3D Landmark detection) in a Voxel Grid and use it as a cubic matrix. Breaking it down, if we have a set of n 3D points (x,y,z coordinates, so an nx3 matrix) we can normalize them in a desired interval [0-k]. This allows us to create a cubic k x k x k matrix and use said normalized coordinates as indexes of the three-dimensional matrix, thus allowing for the remapping of the two-dimensional matrix in a more complex 3D representation, as shown in [Figure 3]. Voxelization has also proven to be a valid simplification of 3D Meshes [9] in the context of Facial Recognition.
- **3D Mesh**: a mesh is composed of triangles and vertices. This is by far the hardest representation to work on since we are not dealing with pixels, voxels, or even matrices. An example of a simple - in terms of surface details - mesh of a human face is shown in [Figure 1]. The models to reconstruct a mesh from a 2D image, in particular with facial images, can be quite expensive both to train and to use, and the process of working on a mesh is very challenging, due to its representation complexity. A technique that has been proven to work quite well on segmentation [2], is a convolution method that works on an edge and the four edges of the incident triangles. Pooling is applied by collapsing said edges while retaining a connected topology of the surface.

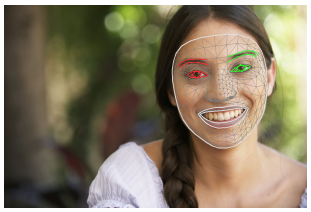

Figure 1: 3D Mesh

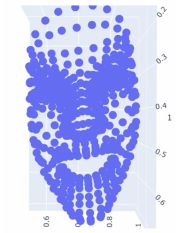

Figure 2: Landmarks

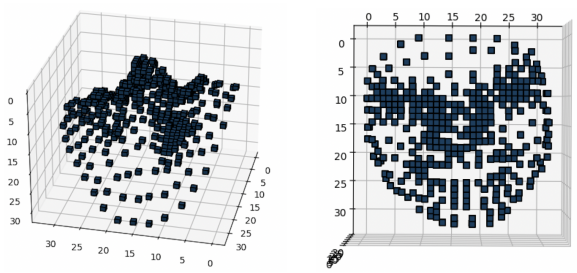

Figure 3: Landmark Voxelization

2 GOALS

The project idea is to use machine learning algorithms to train a deep neural network model that can recognize emotions in facial images. The main focus is put on creating a 2D model that can work as a benchmark for 3D model(s) (which dataset is created via reconstruction) so that we can enlighten the differences in performance between them.
The system will also incorporate user feedback mechanisms to improve accuracy and adaptability to new scenarios. We will try to ensure that the system works in a real-world scenario. Prioritizing these goals, we report the following hierarchy.

Basic Goals

- Build a model and an image processing pipeline that can work in a real 2D Facial Emotion Recognition problem
- Experiment with at least one 3D reconstruction model
- Analyze and compare the performances of the 2D and 3D model

Advanced Goals

- Experiment with more than one 3D reconstruction model
- Implement user feedback
- The 3D reconstruction is part of the pipeline
- A comprehensive analysis of multiple performance indicators is given.

2.1 Research Questions

The *goal* of the study is to evaluate how deep learning models can work on 3D reconstructed facial images, with the *purpose* of understanding whether said models can compete with 2D facial image recognition deep learning models; the *perspective* of the study is that of researchers: they could be interested in understanding the effectiveness of 3D emotion recognition in absence of a native 3D emotion-annotated facial images dataset, and if there is a 3D-reconstruction that works better than the others in said task.
In order to pursue our main goal, we establish a set of questions that we will try to provide an answer to. First, we need to build a neural network model that can work on 2D images, in order to have a first assessment of how deep learning works on the facial emotion recognition task. Although it is already part of other studies, this will be a baseline to which we can compare the 3D reconstruction models.

> **RQ1** *To what extent can deep neural network models recognize emotions in 2D images?*

We are also going to experiment with the preprocessing pipeline, in order to assess whether it plays an important role or not when it comes to the model's performance.
Once we have a baseline, we can experiment with 3D models based on reconstructed 3D images. After that, we will be able to give an answer to RQ2, explained as follows.

> **RQ2** *How do deep neural network models built with 2D images compare to the models built with 3D images?*

The different representations presented in subsection 1.2 have different complexity, both in terms of the reconstruction process and suitable models. For this reason, we split RQ2 into three different RQs.

> **RQ2$_1$** *How do deep neural network models built with 2D images compare to the model built for 3D Landmarks?*

> **RQ2$_2$** *How do deep neural network models built with 2D images compare to the model built for 3D Voxels?*

> **RQ2$_3$** *How do deep neural network models built with 2D images compare to the model built for 3D Mesh?*

The same principles used when we prioritized the goals apply here, so we may not assess all the research questions, but we are going to answer RQ1, as it is propedeutic to evaluate the 3D models of RQ2, and we are going to answer RQ2 by evaluating at least one of its sub-questions.

3 MODELS CONSTRUCTION

In order to conduct the study, we will need to build at least two models, one for the 2D baseline and one or more for the 3D part of the project. Also shown in Figure 4, in this section we present the steps to be taken to build such models.

(1) **Dataset Selection -** First things first, to build a model we have to know the data we are working with; this means that we have to select one or more datasets for the experiment. As we made explicit earlier, we are waiting to receive a few

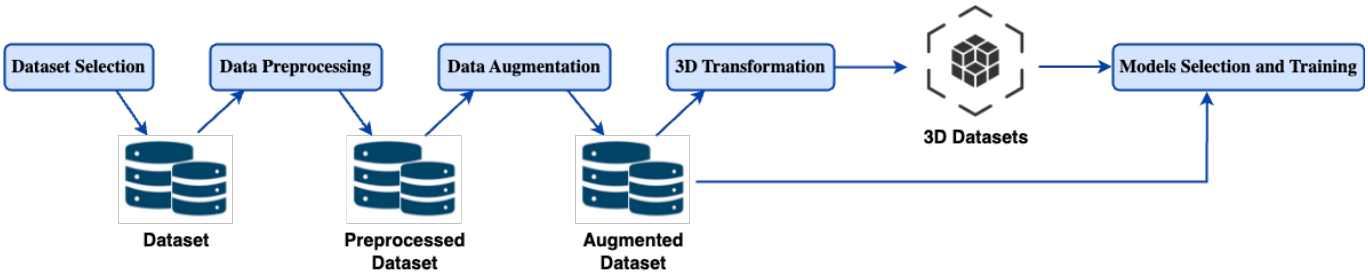

Figure 4: Models construction pipeline

datasets that we will evaluate. We may decide to integrate more datasets, as it could add variety to the captures.
In that case, the preprocessing pipeline would play a fundamental role. The dimension of the images is, of course, a key feature to consider when building a deep learning model, since it influences the structure of the network. Another thing that influences the structure of the model is the number of classes, since it influences the output layer of the learner.

(2) **Data Preprocessing -** Since we have to assess that the model works in a real-world scenario, but the data available are mostly not similar to what we would obtain today with a photo, we have to put our first focus on normalizing the images before both training and inferring of the model.
This means that, in our idea, the preprocessing pipeline would be executed both on the training images and the images provided during the use phase (infer phase) of the model. The rationale behind this phase is that if we can bring all the images in very similar status, the deep neural network could be able to work on all of them, despite the capture conditions. The main steps are the following: uniforming the image color scale (RGB, grayscale, etc.), trying to extract only the facial portion of the image, adjusting the rotation, trying to uniform the lighting conditions, resize the images. Since we are in a very early phase, we may find out that some techniques are not valid for this case, or we could find other better solutions.
We would also like to asses whether this pipeline is actually useful or not, so we plan to train the same model twice, once with the starting dataset and once with the preprocessed dataset, so that we can compare the results.

(3) **Data Augmentation -** If needed, we might use image augmentation techniques to improve the adaptability of the model, e.g. applying noise, rotating images, adjusting lighting, and inverting the color matrix, while trying not to undo the effects of the previous phase.

(4) **3D Transformation -** In this phase, we will experiment with 3D reconstructions, trying to understand which one could work better and more efficiently and what are the limitations when trying to apply these transformations. Although this is not directly connected to model creation, it is a step that allows us to create different representations of the same datasets. These representations will be the input for the 3D models, thus influencing the models themselves.

(5) **Models Selection and Training -** After these steps, we will proceed to model training. First, we will need to split the dataset for training, testing, and validation. After that, likely, we are going to use a CNN for the 2D model, while for the 3D section of the project, we expect to use a 3DCNN (basically a CNN with a 3D kernel) when working with voxels and a modified MeshCNN [2] when working with meshes. We might use some typical architectures, such as ResNet or Inception Network, tuning them to better suit the purpose of the project.

Since our project is somehow directed toward experimentation, the outcome of the project will be a prototype composed of various scripts, structured so that it would be easy to replicate the results obtained by following a detailed procedure.

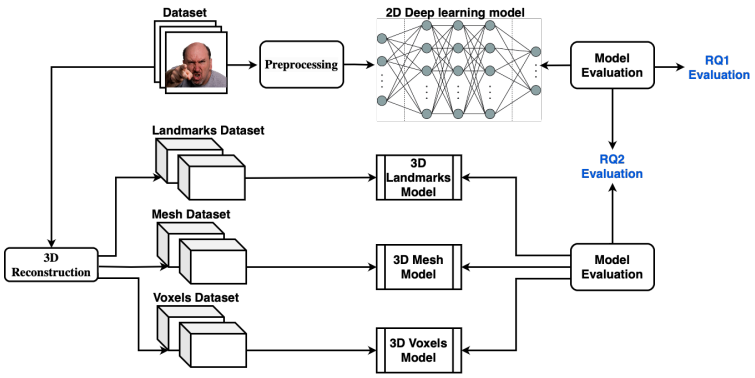

Figure 5: Methodical steps

4 METHODICAL STEPS

In this section, we outline the methodical steps we will take in order to address our research questions, defined in subsection 2.1. We start by discussing the context of our study and data collection, then move on to data analysis, where we explain how we will address our RQs using deep learning models and specific evaluation metrics. By doing this, we aim to obtain insights into the performance of 2D and 3D-reconstruction deep learning models in facial emotion recognition tasks, which will help us understand the potential benefits of using 3D models in this domain.

4.1 Context of the study and Data Collection

Our starting asset will likely be composed of the FER-2013 dataset. This dataset is widely used for emotion recognition tasks and it provides more than 35'000 annotated images, in a 48x48 pixel grayscale format. This dataset comes mostly from internet scraping and has been fully manually labeled. We might want to move on to experiment with a dataset containing bigger images (namely, the "In-the-wild" datasets) if we can get one. This is mainly because of the 3D reconstruction phase, which could be influenced by the starting quality of the images. FER-2013 offers 7 different classes, as explained in subsection 1.1. The dataset is not evenly distributed across the classes, with the "Happy" class being prevalent among the others. For this reason, augmenting the data could be a good strategy. A brief insight into the distribution can be seen in Table 1.

Table 1: FER2013 data distribution

Class	Sample percentage
Happy	25.1%
Neutral	17.3%
Sad	16.8%
Fear	14.3%
Angry	13.9%
Surprise	11%
Disgust	1.5%

4.2 Data Analysis - Addressing the RQs

The steps to be taken in order to address the RQs revolve around the evaluation of the models, as shown in Figure 5. More in detail:

- **Addressing RQ1 -** In order to address RQ1, we are going to develop and train a 2D deep learning model based on a 2D facial image dataset, as explained in section 3. We will perform an evaluation of the model using some specific metrics such as accuracy, precision, F1 score and AUC-ROC (sensitivity, 1-specificity). These metrics are going to be the key to fulfilling RQ1, as they will allow us to understand the extent to which deep neural networks can recognize emotions in 2D images.
- **Addressing RQ2 -** In order to address RQ2 and its sub-RQs, we are going to use the same starting 2D dataset and we will perform 3D reconstructions to obtain the 3D datasets on which we will develop the 3D deep learning models. For each of the models developed, we will perform an evaluation using the metrics listed for RQ1 with the addition of the processing time metric. These metrics, and the comparison between them and those obtained by addressing RQ1, will allow us to fulfill RQ2.

5 IMPLEMENTATION

In this section, we discuss how the project was implemented.

5.1 Dataset Selection

In the end, the chosen dataset was Fer2013, mainly because of the lack of available alternatives. For the same reason, we couldn't join two or more datasets, especially because other (smaller) datasets available use different labels. The image size, unfortunately, is a problem, as 48x48 images mainly introduce a bias toward the transformations, causing them to be less accurate. This is a point to keep in mind when drawing conclusions.

5.2 Preprocessing and Augmentation

We tried to set up a preprocessing pipeline, but we encountered several problems. First, in order to crop the images to just the facial portion, we tried to use DLib. The problem with that, and a few other options we tried, was the quality of the images (that had to be upscaled to 80x80 to make DLib work), which made the face detector very imprecise. While we could have kept them and tried to use them for the 2D models, the crops made the Landmark Extractor discard too many pictures, as the certainty with which it could localize the landmarks often dropped below 50%, resulting in the operation to abort. Since we could not do context-aware changes to the dataset, we just stuck to testing the normalizations of the pixels. We tried standard normalization and min-max, and both had little to no impact on the models. When it comes to augmentation, we did not encounter any problems with the 2D models, since we could add transformations to the dataloader. Within the typical transformations, we found that random rotation/flips are the ones that most benefitted the models, especially when it comes to the more imbalanced classes. The 3D models augmentations are a problem because there is no generator able to augment 3D data. Yet, we managed to use SMOTE on the Landmarks, by temporarily reshaping the tensor in a 2-dimensional one, and then reshaping the tensor back to 3 dimensions. This worked fine for Landmark models, but worsened the overall performance of the other 3D models so much that the slight improvement in the oversampled classes was not worth keeping.

5.3 3D Transformation

We managed to complete all three transformations. The main component used is Google MediaPipe. This allowed us to obtain 478 3D landmarks, normalized for the facial width, height and estimated depth. It is important to notice that, during the transformation, MediaPipe could not always identify the face contained in the image, thus reducing the dataset size to 26'700 training images and 6'678 test images. Then again, this was caused mainly by the size of the images (low detail). The distribution of the labels is approximately the same as it was before. We performed the voxelization starting from the landmarks, following what was explained in subsection 1.2. For the meshes, we followed the format of vertices, edges and faces. The vertices are represented by the landmarks, the edges are provided by MediaPipe, in the form of connections between couples of indexes representing vertices and are fixed for every image, while we derived the faces as triangles generated by the connections. The fact that the edges are fixed means that the links between the landmarks are always the same (e.g. the landmark associated with the start of the left eyebrow is always connected with the landmark associated with the end of the left eyebrow, and their number [0-477] is always the same for every image). This did not allow us to use some of the classical models on meshes, since that would result in operating on landmarks with the addition of a simple fixed part, that is likely to be ignored by the networks.

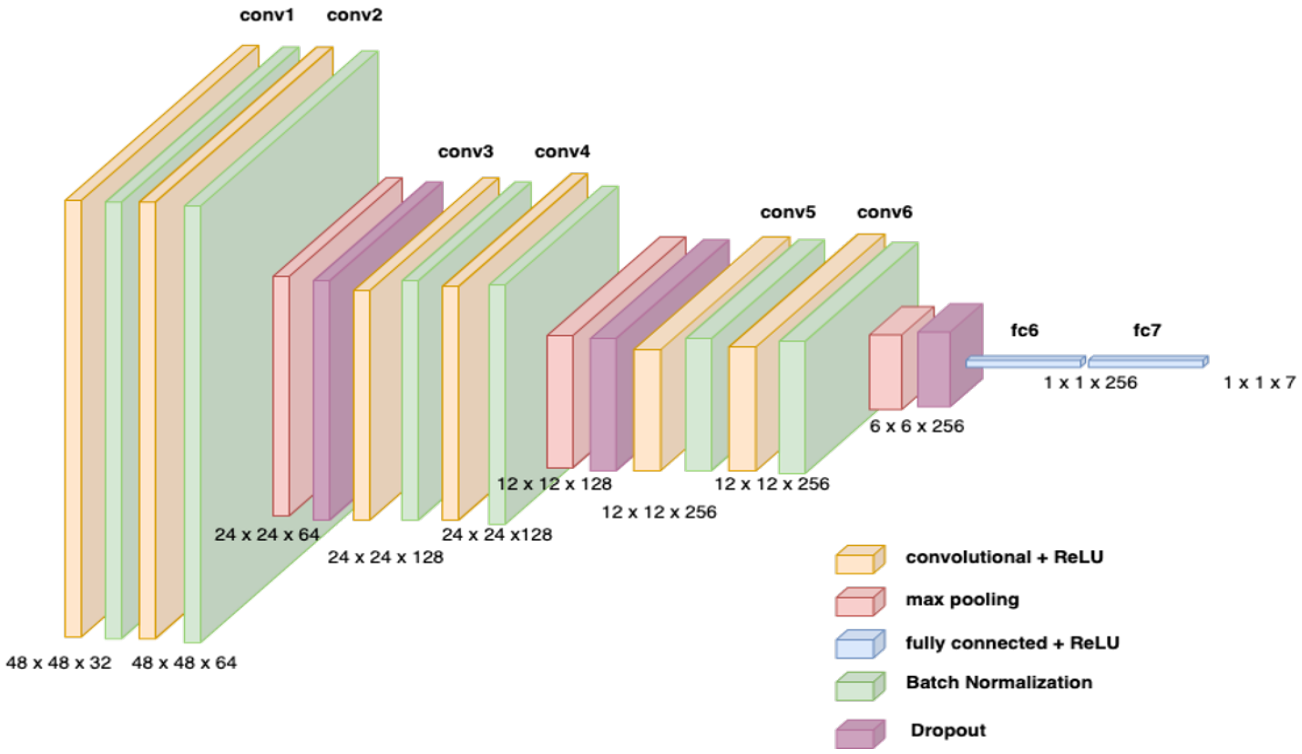

Figure 6: 2DCNN - Architecture

5.4 Models selection

We experimented with different models, both for the 2D dataset and the 3D datasets.

5.4.1 ***2D Models.*** For the 2D datasets, we created three models:

- CNN: the network is composed of 6 convolutional layers, followed by batch normalizations and activation functions (ReLu showed the best results). There are multiple dropout and pooling layers. The architecture is shown in Figure 6. In the end, there are 2 fully connected layers. The images, being in grayscale, only have one channel (although they can be used with three).
- ResNet50: we employed one of the most used architectures for image recognition since the residual connections were introduced. This model was trained from scratch. To suit the architecture, we used a 3 channels representation (RGB) of the grayscale images.
- VGG16: we modified the VGG architecture to suit our purposes and performed **transfer learning**, using the weights of a face recognition task, namely VGGFace. We tried both freezing the pre-trained layers and not doing so, with the latter resulting in better overall performances.

5.4.2 ***3D Models.***

- Landmark MLP: the network is composed of 4 fully connected layers.
- Landmark CNN: there are 6 convolutional layers (1-dimensional in this case since data are in the shape of 478x3). There are two max-pooling and two dense layers.
- Voxel CNN: the 3 layers of convolution are 3-dimensional and are followed by just one pooling layer and 3 fully connected layers. Going for depth or adding pooling layers did not result in any improvement.
- Distance CNN: we exploited the connectivity of the mesh landmarks to compute distances between the points, and used a model with 3 convolutional and 3 fully connected layers. Input is in the shape of [Nx2626], where 2626 are the distances between the connected landmarks.

Unfortunately, we were not able to run MeshCNN on the meshes reconstruction. We tried to use Pytorch3D and its GraphConvolution as an alternative, but it did not suit the purpose of the experiment. MeshCNN is quite difficult to modify, especially on Colab, and is built to work for segmentation tasks. With more time, we could have been able to successfully complete the experiment.

5.5 Models training

We used the standard splits already given by the dataset. The loss we chose is the cross-entropy loss: in this loss function, the outputs of the networks undergo softmax in order to obtain the probability distribution of the classification task. Given

(1) y as the vector of class labels (ground truth).
(2) $\hat{y}$ as the vector of predicted probabilities for each class.
(3) i as the i-th element of the vector y.

$$\text{CrossEntropyLoss}(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i)$$

For almost every model, we performed **Grid Search** to tune batch size, learning rate, weight decay, and momentum. The values chosen for each of the parameters are selected from a set of typical values. Each of the training iterations was logged on **MlFlow** so that we can have all the experiments under control. To avoid overfit and time waste, we employed early stopping on the validation loss.

6 RESULTS

In this section, we collect the results and answer the Research Questions.

6.1 RQ1 - 2D Models

We have experimented with various models and configurations, and the best three of these, namely 2DCnn, ResNet50, and pre-trained VGG16, resulted in the latter being the best when evaluated on all the metrics, achieving on the test set an accuracy of 67%, as visible in Figure 12. The other metrics are graphically shown in Figure 9 (F1-Score), Figure 10 (precision), Figure 11 (recall). The training time it took to train this configuration is 1h, for just 7 epochs. Note that we employed early stopping, meaning that we could have obtained slightly better results by keeping the model in the training loop, but that would be very time-consuming. There might also be a better configuration of the model: because of the expensive training time, we were not able to test all the possible configurations of typical hyperparameter values. Yet, we believe that our solution does not come far from the best one, since other grid search iterations did not show any major difference, even across all models. The CNN takes second place, with an accuracy of 62%. The distribution of the metrics across the classes (F1, precision, recall) is very similar to the VGG16 one. It is important to notice that although the performances are worse, the training time is significantly better, with the best configuration taking less than 10 minutes to train. The number one reason is the image size. In fact, 2DCnn takes 48x48 images in input, with one single channel, while, in order to perform transfer learning, we had to upscale the images to 224x224 with three channels for the VGG16. With a decent score, we can state that **2D deep learning models are capable of recognizing emotions in facial images**, with the extent testified by the metrics we collected.

6.1.1 Explaining the results. About the distribution of the metrics on the classes, we can notice how the class *"Happy"* is the best classified one, as expected since it has the most samples out of the dataset. *"Surprise"* is a short second, but it's the class with the second to last amount of samples. This tells us that yes, the distribution matters, but not as much as we expected. About the heavily undersampled class, namely *"Disgust"*, it is not the one with the worst performances, although they are not great. The precision for this class is quite high, but the recall is low, meaning that the model does not misclassify often other classes as *disgust*, but it misclassifies disgust as other classes quite often (considering the number of images with that label). As we can see in Figure 7, not many misclassification patterns can be observed, except for the *neutral* and *fear* classes, often predicted as *sad*. In order to understand why, we tried to use **GradCam as an explainability tool** to visualize the regions of the images that impacted the prediction more. Unfortunately, we were not able to find any common pattern between the classes. Shown in Figure 8 are a few examples of the activation maps, where we can see the model focusing on the eye, eyebrows and mouth regions. This is not always the case, as in some images (partially also here), the zones that influenced the most the decision are outside of the facial area or in portions of the face where, likely, a human would not be able to judge the emotion.

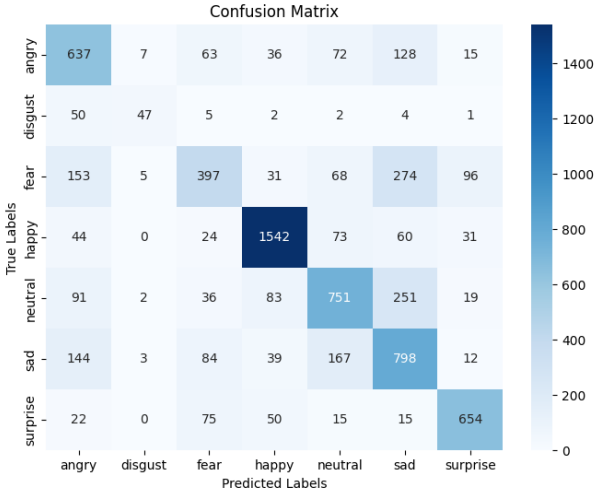

Figure 7: Confusion Matrix - VGG16

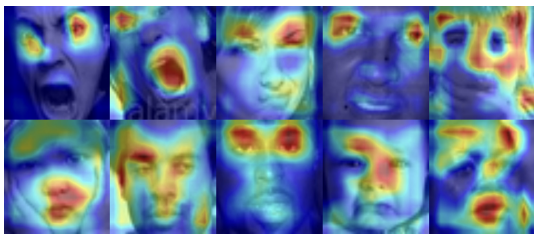

Figure 8: Gradient Activation Map

6.2 RQ2 - 3D Models

6.2.1 $RQ2_1$ – Landmarks. The models on landmarks performed worse than the 2D models, as the CNN outperformed the MLP but still achieved around 55% accuracy, with the other metrics shown, in comparison with other models in Figure 9 (F1-Score), Figure 10 (precision), Figure 11 (recall). We can see how the trend across the classes is quite similar to the 2D models, yet it suffers more on the undersampled classes. The training time was of about 16 minutes. The MLP, on the other hand, was faster (and simpler, but adding layers or incrementing the number of neurons in the hidden layer did not impact on the metrics) but is less accurate, with a 51% score in accuracy achieved in a little over 2 minutes. Overall, we can see how on this dataset **2D models work better than the Landmarks ones when it comes to emotion recognition**

tasks, yet we can assume that working with high-quality images could counterbalance the lack of accuracy of the models; with better images, the very fast landmark extractor would work with way more precision and we can assume that the performances could improve. On the other hand, of course, with better images, a 2D model would be more accurate than the ones we created, but the bigger the input size, the slower the model (as we could already see with the VGG). The balance would be given, in that case, by the training time of the Landmarks models, since its input dimension would still be fixed at (478x3).

6.2.2 $RQ2_2 - Voxels$. The voxelization treatment did not improve by any means the performances with respect to the Landmarks models. In fact, the various configuration we tried got us to a maximum of 43% accuracy, with the remaining of metrics visible, as usual, in Figure 9 (F1-Score), Figure 10 (precision), Figure 11 (recall). This means that changing the dimension of the landmarks does not benefit the data, and resultingly, the metrics. If there was a way to achieve voxelization from a facial image without reshaping by hand the 3D landmarks into cubes, we think the results might be better. The training time is slightly over 18 minutes, but it is clear how **voxelization is not a good alternative to 2D images when it comes to facial emotion recognition tasks**.

6.2.3 $RQ2_3 - Mesh$. As said before, unfortunately, we were not able to use MeshCNN, yet we exploited the connectivity of the mesh to build a landmark distance-based model. The model works slightly better than the landmarks models, achieving an accuracy of 57%, with the other metrics shown, in comparison with the other models in Figure 9 (F1-Score), Figure 10 (precision), Figure 11 (recall). This result suggests that **working on meshes could be a suitable alternative to the 2D models when it comes to emotion recognition tasks**, but it would require more time to build a model that can actually use the entirety of the meshes information, rather than just the topology of the links.

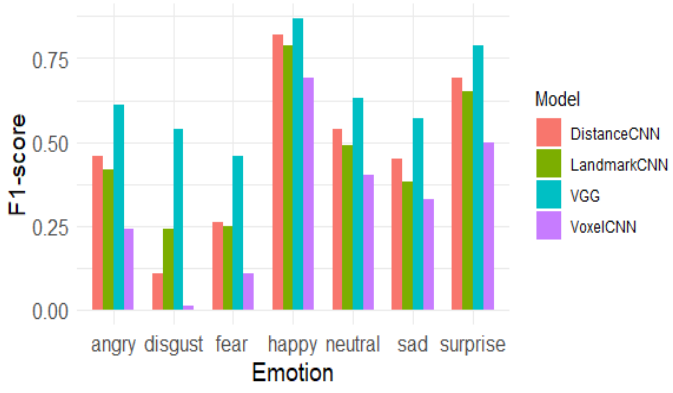

Figure 9: F1 Score - Models overview

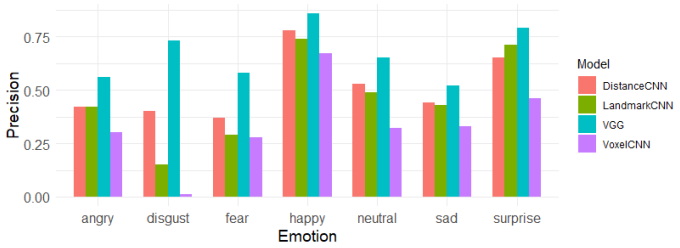

Figure 10: Precision - Models overview

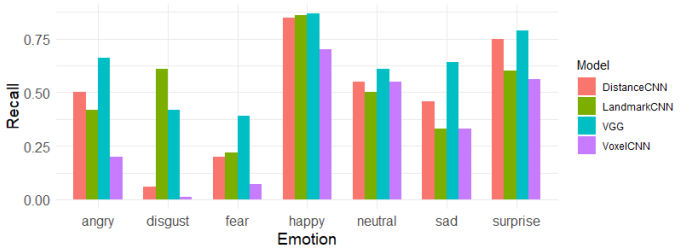

Figure 11: Recall - Models overview

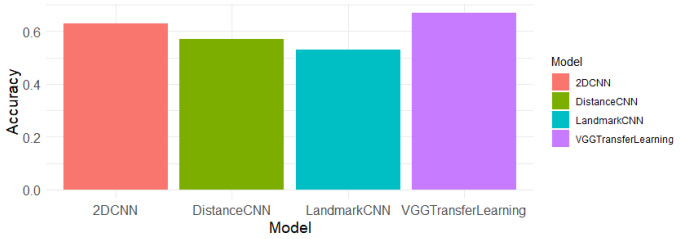

Figure 12: Accuracy - Models overview

7 CONCLUSIONS

Deep Neural Networks are certainly a good choice when it comes to facial emotion recognition in 2D images. We achieved a decent result, testified by a 67% accuracy across 7 different emotions, but we are sure that, with a higher quality dataset we would have obtained significantly better scores. Of course, when it comes to facial images we have to consider the privacy problem, and, in this specific case, the effort to annotate the images. But do 3D reconstructions hold up to the 2D images? We cannot be sure about that: the metrics we obtained showed that right now, at least on low-quality images, they do not. Yet, the Landmark-based models could be a suitable option when, while working with bigger images, the amount of time spent training the model is a constraint. On the other hand, Voxelization did not prove to be a good alternative. In the end we feel like Mesh reconstruction could be, if correctly exploited, the best alternative to classical 2D images. This also means that we believe that it would be possible in the future, as topics like the metaverse get bigger and bigger, and datasets start to become available, to recognize emotions in 3D facial images with good performances.

REFERENCES

[1] Berk Gökberk, Albert Ali Salah, Neşe Alyüz, and Lale Akarun. 2009. *3D Face Recognition: Technology and Applications.* Springer London, London, 217–246. https://doi.org/10.1007/978-1-84882-385-3_9

[2] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. 2019. MeshCNN. *ACM Transactions on Graphics* 38, 4 (jul 2019), 1–12. https://doi.org/10.1145/3306346.3322959

[3] Dimitrios Kollias and Stefanos Zafeiriou. 2018. Aff-wild2: Extending the aff-wild database for affect recognition. *arXiv preprint arXiv:1811.07770* (2018).

[4] Patrick Lucey, Jeffrey F. Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. 2010. The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*. 94–101. https://doi.org/10.1109/CVPRW.2010.5543262

[5] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. 2019. MediaPipe: A Framework for Building Perception Pipelines. arXiv:1906.08172 [cs.DC]

[6] Ali Mollahosseini, Behzad Hasani, and Mohammad H. Mahoor. 2019. AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild. *IEEE Transactions on Affective Computing* 10, 1 (jan 2019), 18–31. https://doi.org/10.1109/taffc.2017.2740923

[7] Eros Piemontez, Rafael; Comunello. 2020. RAP3DF V2 dataset. https://doi.org/doi:10.17632/kpdkpcs8zb.4

[8] Yinghong Qiu and Yi Wan. 2019. Facial Expression Recognition based on Landmarks. In *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. 1356–1360. https://doi.org/10.1109/IAEAC47372.2019.8997580

[9] Sahil Sharma and Vijay Chahar. 2020. Voxel-based 3D face reconstruction and its application to face recognition using sequential deep learning. *Multimedia Tools and Applications* 79 (07 2020). https://doi.org/10.1007/s11042-020-08688-x

[10] Lutfiah Zahara, Purnawarman Musa, Eri Prasetyo, Irwan Karim, and Saiful Musa. 2020. The Facial Emotion Recognition (FER-2013) Dataset for Prediction System of Micro-Expressions Face Using the Convolutional Neural Network (CNN) Algorithm based Raspberry Pi. 1–9. https://doi.org/10.1109/ICIC50835.2020.9288560