Spatio-Temporal Emotion Recognition

Xènia Salinas Ventalló

Abstract

Emotion recognition is the process of identifying human emotion, most typically from facial expressions. The goal of this project is to compare different video analysis techniques to see their performance recognizing facial expressions associated to human emotions in video samples (sequences of frames). For this purpose, four different techniques have been studied. All of them are based on deep learning, the difference is how the temporal dimension is introduced. The first technique is the simplest one, analyze frame by frame and then extrapolate the individual results to the full sequence. Another studied technique is the fusion of temporal information at different levels. Another one is based on 3D convolutions to exploit the temporality. Finally, the last technique is based on Recurrent Neural Networks. As expected, introducing temporal information the performance is improved.

Index Terms

Emotion recognition, Deep Learning, Time Fusion, 3D Convolution, Recurrent Neural Networks.

I. Introduction

Nowadays technology is experiencing a revolution introducing automatic systems which interact with humans in the society. In order to improve the humans experience, it is important that computers interpret human emotions. For this reason emotion recognition is being studied by companies and universities around the world.

In the field of Artificial Intelligence and Computer Vision, emotion recognition is the process of identifying human emotion, most typically from facial expressions. Humans are able to automatically recognize others emotions dealing with the high variability between individuals.

As facial expressions evolve it is important to make a temporal analysis. In this project, four different video analysis techniques based on deep learning are compared. Every technique exploits temporal information in a different way.

The first technique, the simplest one, analyzes frame by frame separately and then extrapolates the individual results to the full sequence. The second one performs time fusion using time windows of different sizes. Another technique explodes the temporal information using 3D convolutions. Finally, the last one is based on Recurrent Neural Networks.

In this way, the goal of this project is compare deep learning techniques for video sequences applied to emotion recognition. This is important because it transfers the acknowledgement from other applications to the current one. To conclude the project, the algorithm with better results is selected and trained with the full dataset to get the final result in emotion recognition. The obtained model could be used in different environments such as robot-human interaction, analysis of humans reaction to advertisements or drivers mood among others.

II. Background

Deep learning is a general framework to perform end to end learning. The main advantage of deep learning against previous frameworks, based on feature extraction and a trainable classifier, is that the machine directly learns the features from data. Moreover, deep learning is able to learn multiple levels of representation.

Deep learning is based in neural networks which are a cascade of non-linear transformations called layers. Each layer transforms its input representation into a higher-level one. Low-level features are shared among categories while high-level features are more global and more invariant.

A neuron is a computational unit with **n** inputs and one output, parameters **W**, a bias term **b** and an activation function, it is represented in Figure 1. The neuron activation defines the output of that node given a set of inputs:

$$output = g\left(b + \sum_{i=1}^{n} w_i x_i\right) \tag{1}$$

The activation function (g()) introduces non-linearity in the model and clamps the signal to within a specified range. There are different activation functions, in this project Linear Rectifier (ReLU) is used. It is bounded below by zero but not upper bounded and is strictly increasing. This activation function has been selected because it does not saturate, converges fast and is computationally efficient.

$$ReLU\,(x) = max\,(0, x) \tag{2}$$

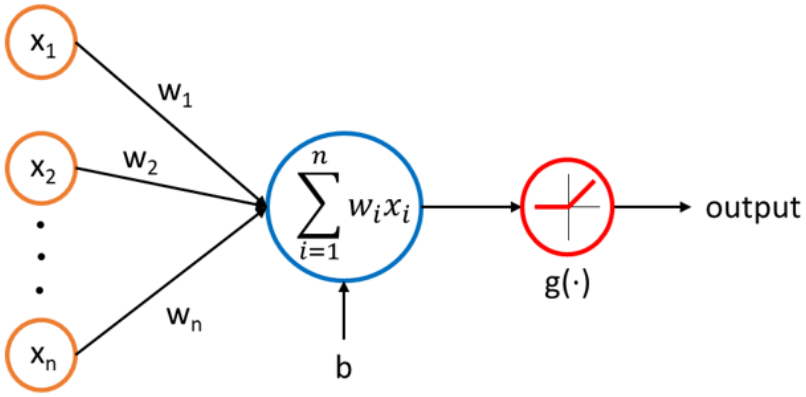


Fig. 1: Artificial neuron scheme.

Multiple artificial neurons form a layer, and multiple layers form a neual network. The first layer is called input layer, the last layer output layer and the intermediate layers are called hidden layers.

A. *Deep Learning with images*

There are different types of deep learning architectures depending on the properties of the data to analyze. One of this types is Convolutional Neural Netwroks (CNN), which are a specialized kind of neural network for processing data that has a known grid-like topology, like images.

Subsequently, the main types of layers that were used to conduct the experiments of this work are described:

1) Convolutional layer: A discrete convolution, Figure 2, is a linear transformation that preserves the notion of ordering.

$$f\left[x,y\right]*g\left[x,y\right]=\sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty}f\left[n_1,n_2\right]*g\left[x-n_1,y-n_2\right] \tag{3}$$

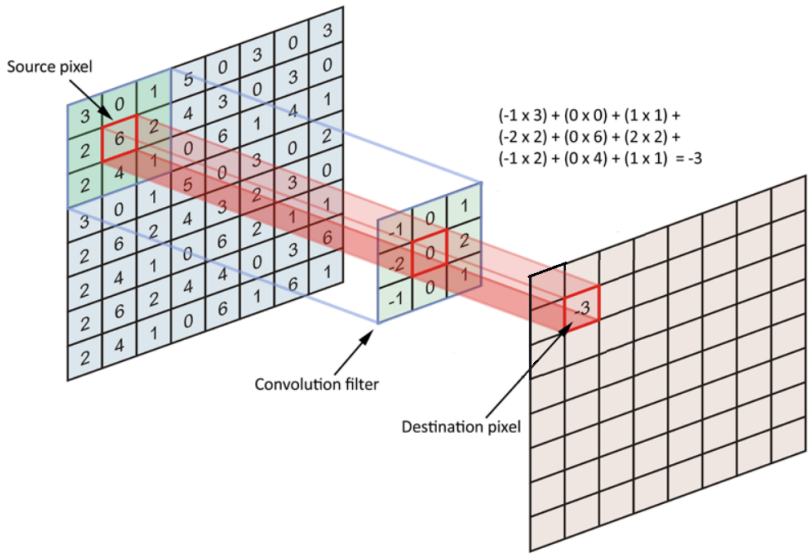


Fig. 2: Convolutional layer example with F=3, S=1 and P=1.

A convolutional layer is locally connected, it exploits spatial correlation in images and reuses parameters since weights are applied to multiple locations. It is sparse in the sense that only a few inputs contribute to a given output and it is stationary since statistics are similar at different locations.

Author: Xènia Salinas Ventalló, salinasxenia@gmail.com
Advisor 1: Sergio Escalera, Mathematics and Computer Science Department, UB and Computer Vision Center, UAB
Advisor 2: Andrés Cencerrado Barraqué, Computer Architecture and Operating Systems Department, UAB
Thesis dissertation submitted: September 2018

The convolution layer has four hyper-parameters: number of filters (D); filter size (F), determines the spatial extend of the connectivity, the higher the largest zone of the image will be connected to the convolution filter; stride (S), determines the step of the convolution filter; and amount of zero padding (P), related to the behavior of the filter at the borders of the image.

2D convolutions accept volumes of size $W1xH1xD1$, where W1 is the width of the image, H1 the height and D1 the number of channels. In this cases each feature map (channel) is convolved with a different kernel and then summed up element-wise to produce the output feature map. The produced output volume has size $W2xH2xD2$ where: $W2 = (W1 - F + 2P)/S + 1$, $H2 = (H1 - F + 2P)/S + 1$ and $D2 = D$.

3D convolutions accept volumes of size $K1xW1xH1xD1$, where K1 is the number of frames. In this cases a new hyper-parameter is introduced, the temporal filter size (T), how many frames are analyzed at the same time.

2D convolutions introduce $K * F^2 * D1$ trainable parameters and 3D convolutions $K * F^2 * D1 * T$ trainable parameters.

2) Max pooling layer: Max pooling layer, Figure 3, is a sub-sampling layer used to reduce the feature map spatial dimensionality. This layer introduces two hyper-parameters: spatial extent (P), the size of the max pooling filter, and stride (S). Considering and input volume with size $W1xH1xD1$ it produces an output with size $W2xH2xD2$, where $W2 = (W1 - P)/S + 1$, $H2 = (H1 - P)/S + 1$ and $D2 = D1$. Max pooling layers introduce zero trainable parameters to the CNN.

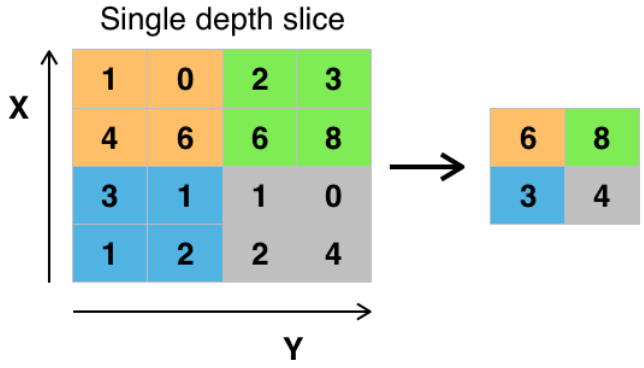


Fig. 3: Max Pooling layer example with F=2 and S=2.

3) Batch Normalization Layer: Batch Normalization Layer is used to reduce covariance shift, the difference in the training batch. During the training stage, this layer learns both the parameters γ, data scaling, and β, data shifting, and the mean and variance of the mini-batch.

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{4}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{5}$$

$$\hat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{6}$$

$$y_i = \gamma \hat{x_i} + \beta \tag{7}$$

It does not introduce any hyper-parameter, but $4 * D$ trainable parameters, where D is the number of input feature maps.

4) Fully Connected Layer: In Fully Connected Layers each neuron is connected to all activations in the previous layer. These layers are usually computed using matrix multiplications with a bias offset. Using this layer the spatial information is lost. It introduces one hyper-parameter, the number of units (Y), which is the output volume size, and $W1 * H1 * D1 * Y$ trainable parameters, where $W1xH1xD1$ is the input volume size.

5) Softmax: Emotion recognition is a multi-class classification problem, this means that the network must have one output per class, specifically a probability per class. Normalized exponential function or Softmax is the function used in the output layer, it is strictly positive and adds up to one. Finally, the predicted class is the one with highest predicted probability.

$$softmax\,(o) = \left[\frac{exp\,(o_1)}{\sum_c exp\,(o_c)} \;\ldots\; \frac{exp\,(o_K)}{\sum_c exp\,(o_c)}\right]^T \tag{8}$$

B. Training

Training is the process in which the deep learning model learns its internal parameters. This is an iterative process based on the minimization of the loss function, representing the price paid for inaccuracy of predictions in classification problems. The iterative process is divided in epochs, a training hyper-parameter. One epoch is when an entire dataset is passed both forward and backward through the neural network only once.

There are different types of training processes depending on the data. In these project, supervised learning is performed. This means that all the samples used during this process are labeled.

III. State of the art

Emotion recognition and emotion analysis are being studied by companies and universities around the world. Many different techniques have been studied to perform these analysis from facial feature tracking as in [1] to [2] where deep learning architectures are used and different streams are combined. In this project we are going to use deep learning architectures to analyze human emotions in sequences, so in this section we will focus in systems that follow this methodology.

In [3] they present a DNN architecture to recognize facial expressions. Specifically, the network consists of two convolutional layers interlaced with max pooling layers and then four Inception layers. In this case, they only use the last frame of the sequence and do not fit the network with the full image, but the face region. Furthermore, all the faces are previously normalized to an average face. This normalization is realized using the face landmarks to compute the necessary affine transformation to adapt the face to the average one. This technique achieves a 93.2% Top-1 accuracy in the CK+ dataset.

In [4] a two-part network for facial expression recognition in videos is proposed. It consists of a Deep Neural Network (DNN) architecture followed by a Conditional Random Field (CRF) module. The DNN architecture captures the spatial relation within facial images and the CRF module captures the temporal relation between the frames. The DNN is composed by convolutional layers followed by three InceptionResNet modules and two fully-connected layers. The CRF takes into account the whole input sequence to determine the most probable sequence of labels. Using the described architecture, they achieve a recognition rate of 93.04% in the CK+ dataset.

In [5] they present a system composed by two deep networks. The first one, deep temporal appearance network (DTAN), is based on a Convolutional Neural Network (CNN) and extracts temporal appearance features from image sequences. The second one, deep temporal geometry network (DTGN), is based in a fully connected DNN and extracts temporal geometry features from temporal facial landmark points. The models are combined using the joint fine-tuning method which consists in a softmax activation function with DTAN and DTGN outputs as inputs. This architecture achieves a 97.25% of accuracy in the CK+ dataset and 81.46% in the Oulu-CASIA dataset.

In [6] is proposed to adapt 3D Convolutional Neural Networks (3D CNN) with deformable action arts constraints to recognize muscle motions that enables to identify facial expressions. They perform a pre-processing which consists in face detection an normalization to the same size. The network is composed by two 3D convolutional layer with generic filters interlaced with max pooling layers. Then the previous feature maps are convolved by specific part filters obtaining the facial action part detection maps. After that, by summing up the part detection map and several deformation maps with learned weights, the deformable detection maps are obtained. Finally, a full connection layer is used to predict the class label. With this algorithm they get a 87.9% of accuracy in the CK+ dataset.

In [7] they introduce audio analysis proposing a Hybrid CNN-Recurrent Neural Network (RNN) architecture. The applied pre-processing consists in histogram equalization and faces detection and alignment. The full architecture consists in a CNN for extracting faces features, a RNN to aggregate frame features and an Support Vector Machine (SVM) for audio analysis. Then the features extracted from this three blocks are combined in the modality layers. Finally there is a decision layer which provides emotion predictions. The previous architecture gets a 52.87% of accuracy in the test set of EmotiW 2015 competition.

In [2] they combine four different streams. The first one is based in a CNN to analyze each video frame, the network outputs of all the frames are aggregated and fitted into an SVM to classify the full clip. The second method is based in a Deep Belief Network (DBN) to analyze the audio features. The third one consists in a relational auto-encoder which performs activity recognition. The last one is a Bag of mouth features, the mouth region is extracted for each face and a regression classifier is used to predict the emotion. Finally, the predictions from the four streams are averaged to obtain the final classification. Following this procedure, they achieved 47.67% of test accuracy in the EmotiW 2014 competition.

In [8] the proposed system is composed by five sub-systems, each one processes different features but all of them use the same prediction classifier architecture. These five feature sets are audio feature, face shape feature, face appearance feature, mouth appearance feature and eye appearance feature. Each of these sub-systems uses a DBN based on a regression model. The prediction results from the five sub-systems are combined with a multimodal-temporal fusion to get the final prediction result. The proposed architecture achieved a 54.99% of accuracy for the AVEC 2014 test set.

In [9] is proposed a unified deep network framework called spatial-temporal recurrent neural network (STRNN) in which they deal with electroencephalogram (EEG) and facial emotion recognition considering both spatial and temporal dependencies. To learn spatial dependencies it uses a quad-directional spatial RNN layer (SRNN) and to learn long-term temporal dependencies it uses a bi-directional temporal RNN layer (TRNN). Salient emotion activation regions can be effectively captured by introducing

sparse projection on those encoding hidden states, which can naturally bundle those co-occurred emotion activation regions by adaptively weighting them. This architecture achieves a 98.4% of accuracy in the CK+ dataset.

IV. Architectures

A. Single Frame

The Single Frame architecture is used as the baseline performance. The network is fitted with a single frame, the input size is 170x170x3 pixels.

The full architecture is represented in Figure 4, the first block is composed by a 2D Convolutional (2DConv) layer with 96 filters of 11 by 11 size with a 3 by 3 stride, followed by a Batch Normalization (BN) layer [10] and finally a Max Pooling (MP) layer which pools spatially in non-overlapping 2 by 2 regions. The second block is composed by a 2DConv layer with 384 filters of 5 by 5 size with a 1 by 1 stride, followed by a BN layer and a MP layer of 2 by 2 non-overlapping regions. The third block is composed by three 2DConv layers of 384, 256 and 256 filters respectively, all of them of 3 by 3 size with a 1 by 1 stride, followed by a MP layer of 2 by 2 non-overlapping regions. Then, in the last block, there are two Fully Connected (FC) layers with 4096 each one. The final layer is connected to a softmax classifier with dense connections.

Finally, the output of all the frames from the same sequence are voted to obtain the emotion related to the observed facial expressions for the full sequence.

Fig. 4: Single Frame architecture: Orange, green, blue, gray and red boxes indicate convolutional, normalization, max pooling, fully connected and softmax classifier layers respectively.

B. Time fusion

This technique was introduced in [11], where different approaches for fusing information over temporal dimension through the network were proposed. They use this approach to classify videos into different activities, whereas in this project is used to classify videos into different facial expressions.

1) Early Fusion: The Early Fusion architecture, shown in Figure 5, is almost the same as in the Single Frame approach. The difference is that in the input, instead of a single frame, it requires a sequence of 10 consecutive frames. With this approach, the information across an entire time window is combined on the pixel level allowing the network to detect local motion.

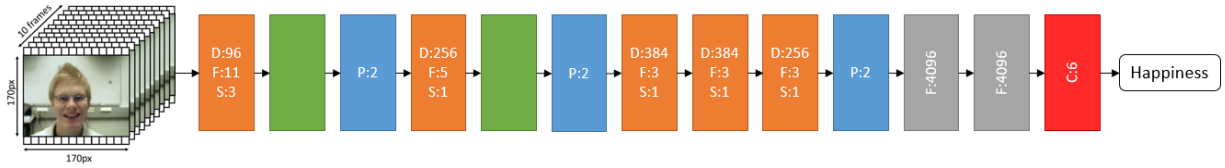


Fig. 5: Early Fusion architecture: Orange, green, blue, gray and red boxes indicate convolutional, normalization, max pooling, fully connected and softmax classifier layers respectively.

2) Late Fusion: The Late Fusion architecture, which is represented in Figure 6, is composed by to towers each on them fitted by a single frame, separated by 15 frames, of the sequence. Both towers are composed by the first, second and third block of the Single Frame architecture and share the parameters. The outputs from both towers are concatenated before the first FC layer. Therefore, the first FC layer can compute global motion characteristics by comparing outputs of both towers.

3) Slow Fusion: The Slow Fusion model is a balanced mix between the two previous approaches. The full architecture is represented in Figure 7. The first block is replicated into four towers, sharing parameters, which accept inputs of four consecutive frames. In order to generate the four inputs a sequence of 10 consecutive frames is used, extracting 4 consecutive frames with stride 2. The second block is replicated into two towers, also sharing parameters. The first two outputs from the previous block are concatenated to fit the first tower and the the others are concatenated to fit the second tower. Finally, the outputs from the two towers of the second block are concatenated to fit the third block. Following this approach the temporal information is slowly fused throughout the network such that higher layers get access to progressively more global information in both spatial and temporal dimensions.

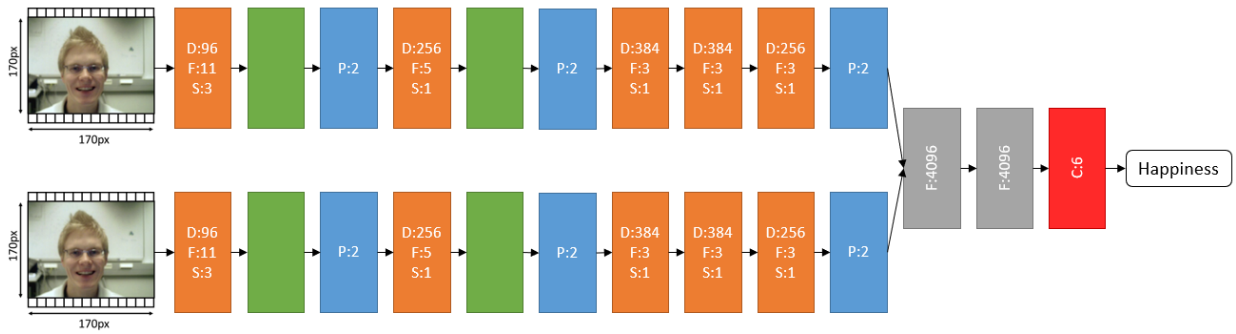


Fig. 6: Late Fusion architecture: Orange, green, blue, gray and red boxes indicate convolutional, normalization, max pooling, fully connected and softmax classifier layers respectively.

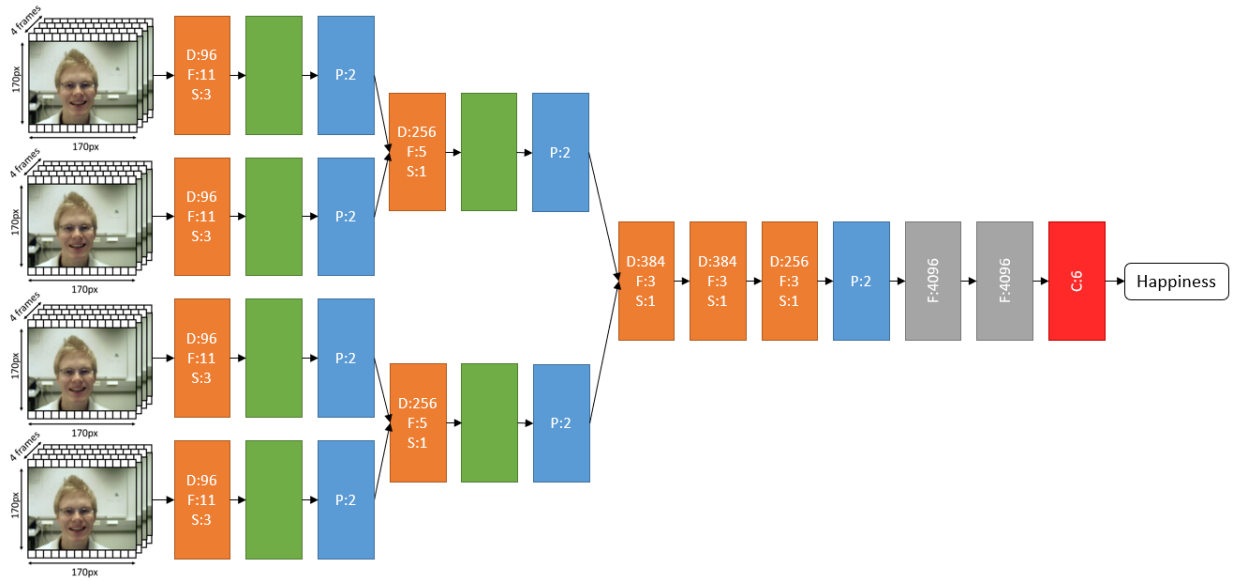


Fig. 7: Slow Fusion architecture: Orange, green, blue, gray and red boxes indicate convolutional, normalization, max pooling, fully connected and softmax classifier layers respectively.

C. 3D Convolution

The 3D Convolution model is also based on the Single Frame architecture, and it is shown in Figure 8. In this architecture, the 2D Convolutions have been replaced by 3D Convolutions and the 2D Max Pooling by 3D Max Pooling layers. The network expects groups of 5 consecutive frames with 170 by 170 and 3 channels at the input. The first block composed by a 3D Convolution (3DConv) layer with 96 filters of 11 by 11 by 5 size with a 3 by 3 by 1 stride, followed by a BN layer and finally a 3D MP layer which pools spatially in non-overlapping 2 by 2 by 1 regions. The second block is composed by a 3DConv layer with 384 filters of 5 by 5 by 3 size with a 1 by 1 by 1 stride, followed by a BN layer and a 3D MP layer of 2 by 2 by 1 non-overlapping regions. The last two blocks are exactly the same as in the Single Frame architecture. Finally the result of the different blocks from the same sequence are voted to obtain the full sequence facial expression.

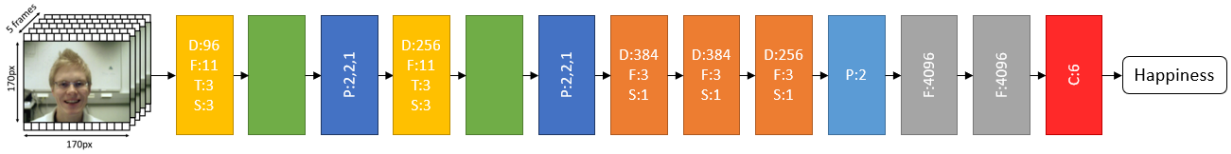
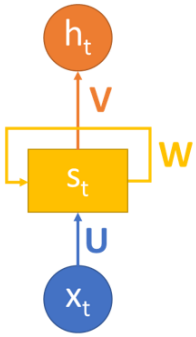


Fig. 8: 3D Convolution architecture: Yellow, green, dark blue, orange, blue, gray and red boxes indicate 3D convolutional, normalization, 3D max pooling, 2D convolutional, 2D max pooling, fully connected and softmax classifier layers respectively.

D. Recurrent Neural Network

Recurrent Neural Networks (RNN) are networks with loops, allowing information to persist. This makes RNN well suited for sequence analysis. In Figure 9 an RNN unit is represented schematically.



$$s_t = tanh\left(Ws_{t-1} + Ux_t\right) \quad (9)$$

$$h_t = f\left(Vs_t\right) \quad (10)$$

Fig. 9: Recurrent Neural Network unit: x_t is the input, s_t the state and h_t the output.

The main problem of RNNs is that the gradient becomes exponentially smaller/bigger with longer time differences. One solution to this problem is the introduction of vanishing gradient, done with Long Short Term Memory (LSTM) [12] and Gated Recurrent Unit (GRU) [13]. LSTMs and GRUs have many similarities between them, the main differences are that GRUs have two gates instead of the three from LSTMs and that hidden state and memory cell are merged while in LSTMs are separated. Cause of these differences, GRUs have less trainable parameters than LSTMs, so they need less time to train and generalize.

The used datasets are small, less than 100 samples for class, so GRUs should perform better than LSTMs. Instead of fitting the RNN with the full sequence, we fit the network with the output of the Single Frame CNN, see Figure 10.

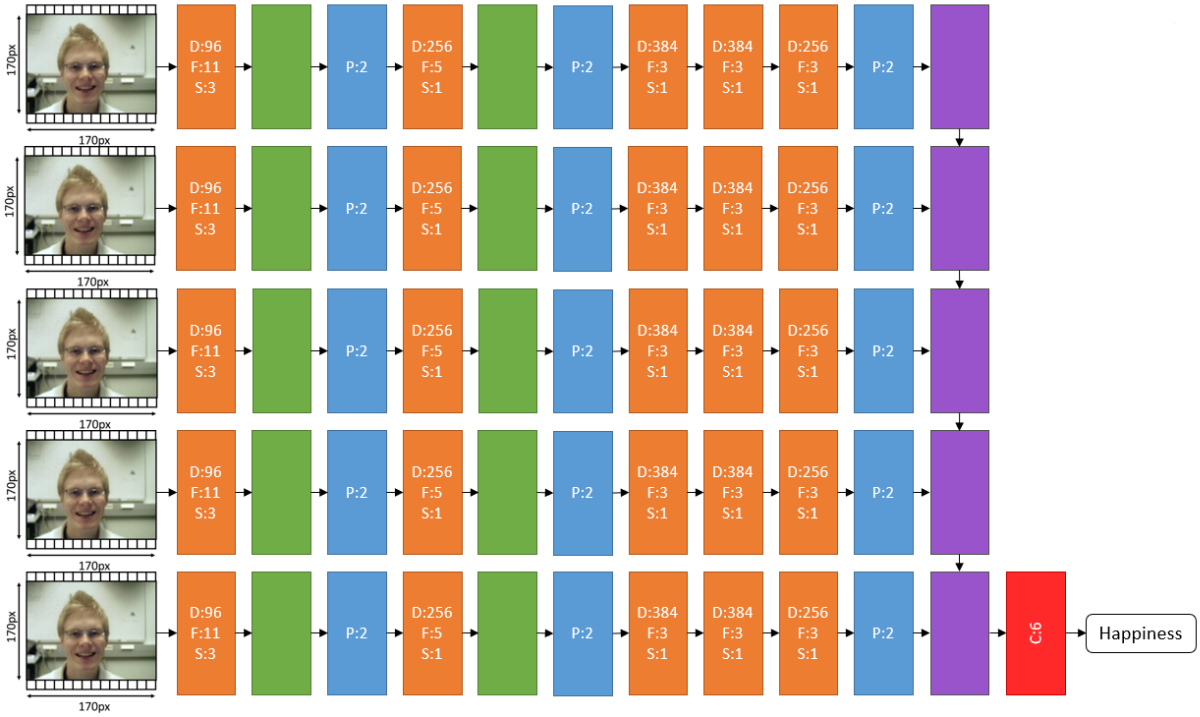


Fig. 10: RNN architecture: Orange, green, blue, violet and red boxes indicate convolutional, normalization, max pooling, GRU and softmax classifier layers respectively.

V. Methodology

A. Architecture programming

Nowadays exist many deep learning frameworks developed by different universities and companies, Figure 11.

Fig. 11: Deep learning frameworks.

For this project Keras 2 [14] running on top of TensorFlow [15] has been used. Keras is a Python Neural Network library designed to be minimalistic and straightforward yet extensive which is build on top of either Theano [16] or TensorFlow. Keras has been selected because is simple to get started and deep enough to build the required models. As Keras is developed in Python 3 [17] the project is implemented in this programming language.

B. Datasets and evaluation

For the experiments two different datasets have been used, the first one is Extended Cohn-Kanade Dataset and Oulu-CASIA VIS Dataset.

- **Extended Cohn-Kanade Dataset (CK+)** [18]: it contains 123 subjects and 593 frontal image sequences of posed expressions. Among them, 118 subjects are annotated with the seven labels: anger, contempt, disgust, fear, happiness, sadness and surprise. Each sequence begins with a neutral facial expression and ends with the apex of the expression. For this project contempt samples have been removed.
- **Oulu-CASIA VIS Dataset** [19]: it is a subset of the Oulu-CASIA NIR-VIS Dataset in which all the videos were taken under the visible (VIS) light condition. It includes 480 image sequences of 80 subjects taken under different illumination conditions: dark, weak and strong. They are labeled with one of the six basic emotion labels: anger, disgust, fear, happiness, sadness and surprise. As in the previous dataset, each sequence begins with a neutral facial expression and ends with the apex of the labeled expression. In this project only the strong illumination is used.

For experimental purposes, sequences shorter than 15 frames have been removed, the resulting classes distribution is showed in Figure 12.

C. Preprocessing

In the first instance, no preprocessing was applied to the images. In both cases the sequences are composed by frames in which the face is already frontalized and the face occupies a similar percentage of the full image, around the 20% of the image. Cause of these characteristics, represented in Figure 13, it was thought that preprocessing was not needed.

The first experiments showed that this first intuition was not right, although the network was able to obtain a perfect accuracy for the training set, the results in the validation set where extremely poor, see Figure 14. For this reason a face detection is performed as a preprocessing step, fitting the network with the cropped face instead of the full frame.

The face detection process is done using the improved face detection model [20] from OpenCV [21]. This framework provides a trained cascade classifier composed by 19 classifiers. All of them are trained with Gentle AdaBoost [22] to detect frontal faces using Local Binary Pattern (LBP) features [23].

A cascade of classifiers is a set of classifiers where each classifier considers the acknowledgement from the previous one to refine the result. The first classifiers are used to reject the majority of sub-windows. Then the following ones, take advantage from the previous classifiers acknowledgment to refine the result. A positive result from one classifier triggers next classifier

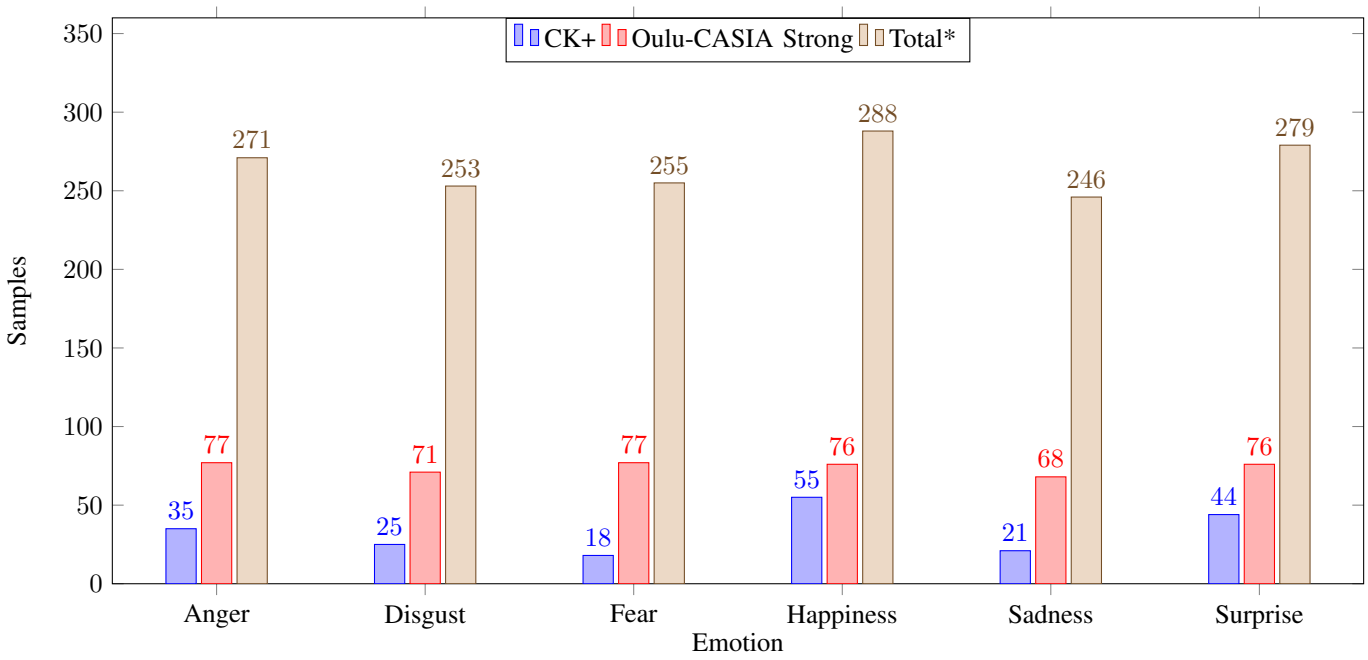
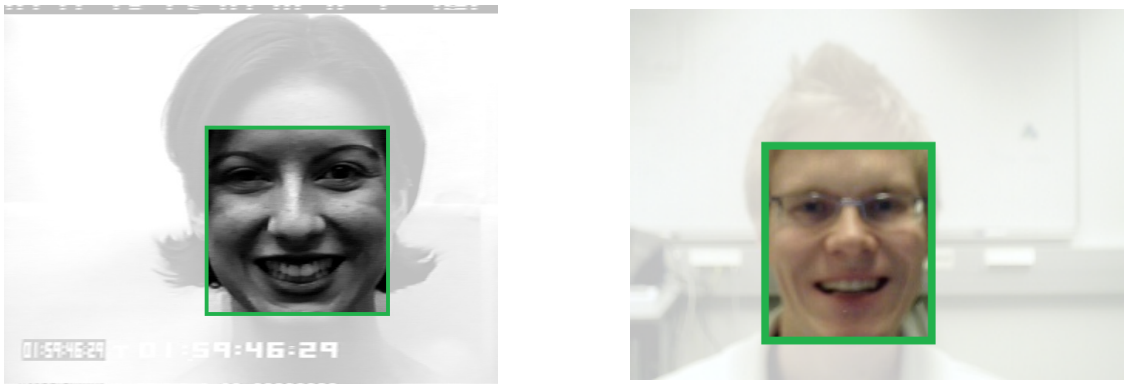


Fig. 12: Emotions distributions in the reduced dataset *(Total = CK+ and Oulu-CASIA Strong, Weak and Dark)



(a) CK+ dataset

(b) OuluCasia dataset

Fig. 13: Datasets frame example

which is adjusted to achieve a higher detection rate than previous one. On the other hand, a negative result is immediately rejected at any stage of cascade structure. At the beginning the algorithm produces a high number of sub-windows to search the faces, but the majority of input sub-windows are false detections, this method can significantly reduce the number of sub-windows which are processed with more complex classifiers, reducing the needed computational resources.

The LBP feature vector is created in the following manner, Figure 15:

1) Divide the image window into 16x16 pixels cells.
2) For each pixel in a cell, compare the pixel to each of its 8 neighbors. If the center pixel's value is greater than or equal to the neighbor's value, write "0". Otherwise, write "1".
3) Following the clockwise order, an 8-digit binary number is obtained which then is converted to decimal.
4) Compute the histogram over the cell, this can be seen as a 256-dimensional feature vector.
5) Concatenate the histograms of all cells obtaining the feature vector for the entire window.

The previously explained preprocessing process has been realized apart from the training and testing process, it requires 0.093s per frame. 253 faces of 36575 had not been detected with this algorithm, all of them correspond to sequences from OuluCasia dataset with dark and weak illumination conditions; the corresponding sequences have been deleted.

D. Computational resources

The cost of a convolution depends on the input, the filter size and the output. Considering that the input is composed by Q feature maps of M by N size, the desired output is composed by R feature maps and that the convolution filter has K by L

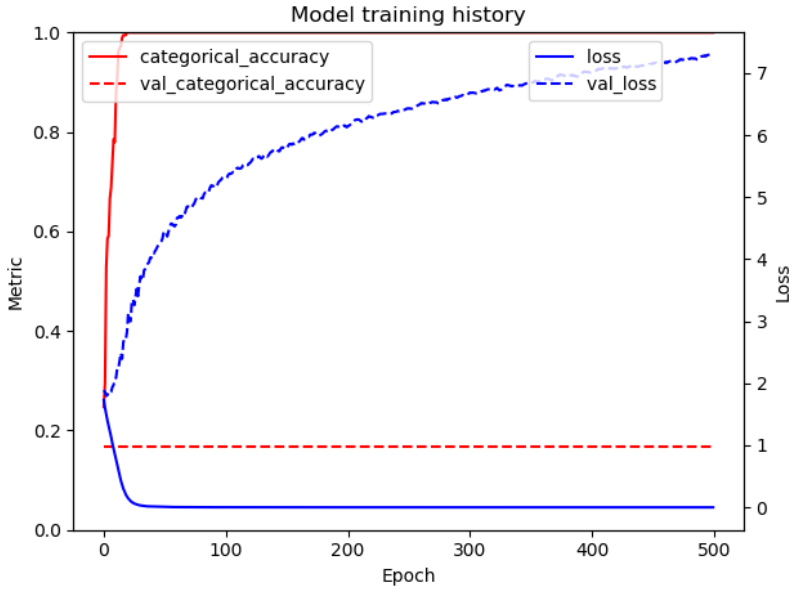


Fig. 14: Training process with the full frame.

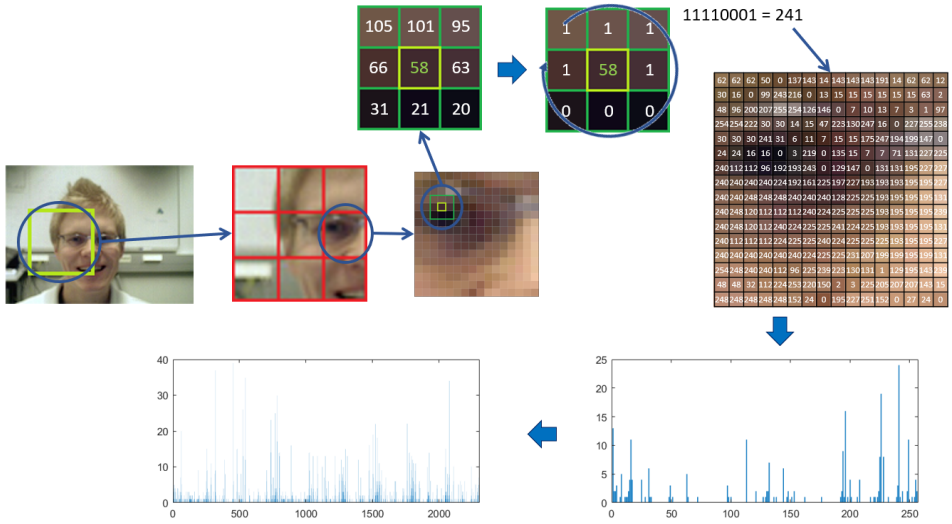


Fig. 15: LBP feature vector creation procedure.

size; the cost of this convolution is $R * Q * M * N * K * L$. This means that in the first layers, where the input feature map is the full image, the cost is extremely high.

Considering the cost of a single convolution, training deep learning models requires high computational resources. For this reason a Graphic Processing Unit (GPU) is used, specifically a NVIDIA GEFORCE 940M.

VI. Experiments

A. Experimental setup

In order to compare the different architectures, the training conditions must be the same or at least similar between them. The training conditions are related to the architecture and the optimizer.

The architecture conditions are ensured using deep learning models with a similar number of parameters. In Table I the number of trainable parameters of each architecture is showed, the difference between them is small enough to ensure similar architecture conditions. The RNN architecture is the only one that has a significant difference in the number of parameters. This is because it does not have any fully connected layer.

The optimizer conditions are ensured using the same hyper-parameters for all the architectures. In particular, the Adaptive Moment Estimation (Adam) optimizer [24] is used with the learning rate set to 0.00001 . Adam works well in practice and

Architecture	Trainable Parameters
Single Frame	$24,760,392$
Early Fusion	$25,074,024$
Late Fusion	$28,954,696$
Slow Fusion	$26,355,878$
3D Convolution	$26,050,694$
Recurrent Neural Network	$3,763,390$

TABLE I: Trainable parameters of each architecture

compares favorably to other adaptive learning-method algorithms as it converges very fast, the learning speed of the model is quite fast and efficient. Moreover, it rectifies problems of other optimization techniques such as vanishing learning rate, slow convergence or high variance in the parameter updates which leads to fluctuating loss function.

Furthermore, the corresponding parameters are learned from scratch during 200 epochs using batches of 32 samples in all the cases.

Each dataset is divided into training and validation sets using the Leave-One-Subject-Out protocol (LOSO) in order to be able to compare the obtained results with the state-of-the-art ones. The training set is used for learning the weights of the architecture layers. The validation set is used to tune the hyper-parameters of the architecture, specifically is used to select the best architecture.

The different architectures are evaluated using the categorical accuracy which can be expressed as:

$$acc = \frac{1}{N}\sum_{i=1}^{N}\left(y_{true}^{i} == y_{pred}^{i}\right) \tag{11}$$

where N is the number of samples, y_{true}^{i} is the ground truth emotion for the sample *i* and y_{pred}^{i} is the predicted emotion for the sample *i*.

B. Results

1) Single Frame: The training phase is carried out using the last frame of each sequence because, in every video, the emotion evolves from neutral to the corresponding emotion, so the last frame corresponds to the apogee. The validation phase is performed fitting the network with each frame of the sequence and voting all the results to obtain the sequence corresponding emotion. The voting step is done in two different ways: maximum probability and summing probabilities, Figure 16. The maximum probability option consists in selecting the emotion with the maximum probability for each frame and then taking the emotion assigned to more frames. The summing probabilities option consists in, as its name indicate, summing the probabilities obtained for each frame and selecting the emotion with the highest total probability.

The training phase for this model is represented in Figure 17, where the history is depicted. Analyzing the graphics from both datasets, we can see that the model learns faster in the Oulu-CASIA dataset. This was the expected behavior since this dataset is larger, so the model has more information.

Another difference between them is that in the case of Oulu-CASIA, the validation loss remains larger than the training one while in CK+ both are reduced to similar values. The graphics correspond to one particular subject, as LOSO is applied. The validation behavior is different between subjects in the same dataset. For this reason no conclusion can be extracted from this difference.

The validation result for both voting options and datasets is showed in Table II. In both datasets the validation accuracy is slightly better with the summing probabilities option. With this metric we can validate that as Oulu-CASIA dataset is larger, the model has more information to learn from, so better results are obtained.

Architecture	Validation accuracy	
	CK+ dataset	Oulu-CASIA dataset
Single Frame with maximum probability	0.379	0.560
Single Frame with summing probabilities	0.383	0.562

TABLE II: Validation accuracies of single frame model

In order to get a more comprehensible validation metric, the confusion matrix for both voting options is created. As we could see with the validation accuracy, there are minimum differences between the maximum probability voting option, Figure 18, and the summing probabilities option Figure 19. In the Oulu-CASIA dataset the confusion matrix diagonal can be clearly appreciated, with some exceptions, while in the CK+ dataset it is extremely blurred.

Probabilities	Frame 1	Frame 2	Frame 3
Anger	0,14	0,12	0,11
Disgust	0,15	0,14	0,14
Fear	0,13	0,15	0,16
Happy	0,23	0,21	0,28
Sadness	0,11	0,1	0,13
Surprise	0,24	0,28	0,18

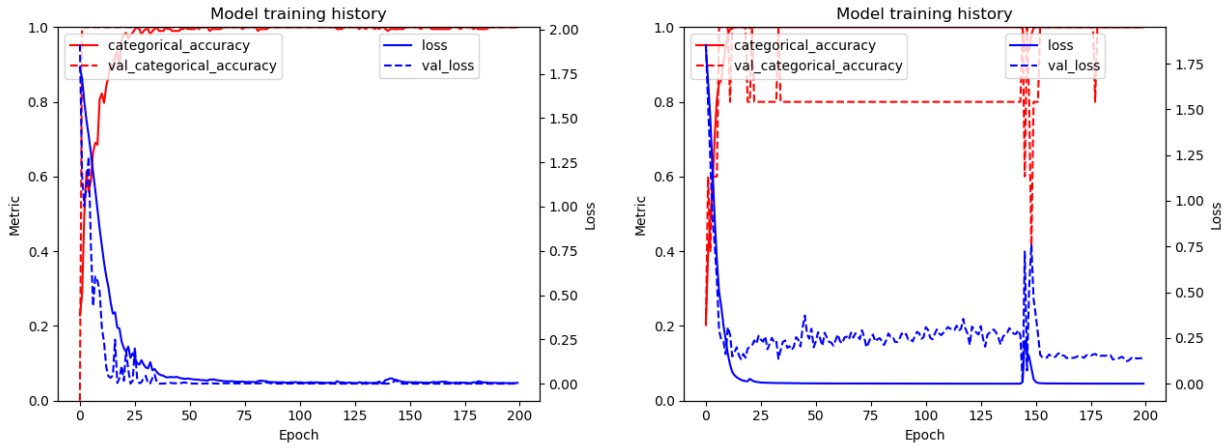
	Total probability
Anger	0,37
Disgust	0,43
Fear	0,44
Happy	0,72
Sadness	0,34
Surprise	0,7

Summing probabilities → Happy

	Frame 1	Frame 2	Frame 3
Maximum probability	Surprise	Surprise	Happy

Maximum probability → Surprise

Fig. 16: Voting options.



(a) CK+ dataset (b) OuluCasia dataset

Fig. 17: Training process for Single Frame model

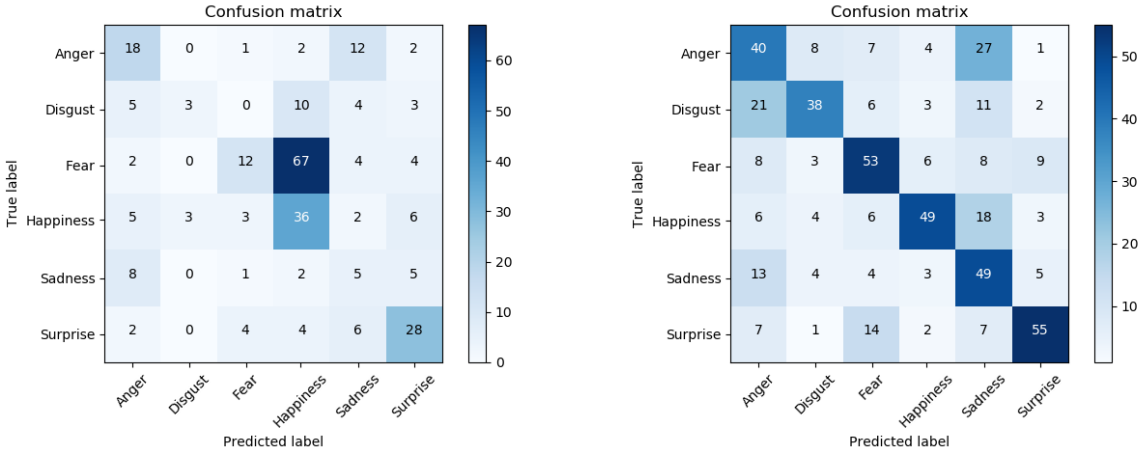
2) Early Fusion: For the training phase the last ten frames are used, the ones that correspond to the maximum facial expression of the emotion. In the validation phase the same window is used.

The history of the training phase is represented in Figure 20. As can be seen, in this case the model learns equally for both datasets. In the CK+ dataset the validation loss fluctuates during all the epochs, but as in the previous case this can not be extrapolated for the full dataset.

The validation accuracy, in Table III, has improved in both datasets. For the CK+ dataset the difference is 0.087 (22.72%) while in the OUlu-CASIA dataset the improvement is 0.013 (2.36%), much more lower.

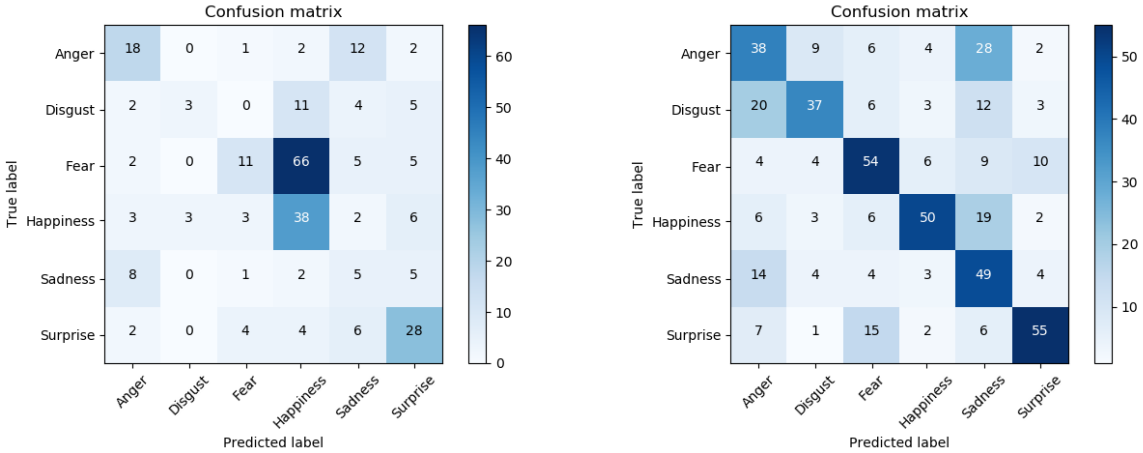
Architecture	Validation accuracy	
	CK+ dataset	Oulu-CASIA dataset
Early Fusion	0.470	0.575

TABLE III: Validation accuracies of early fusion model



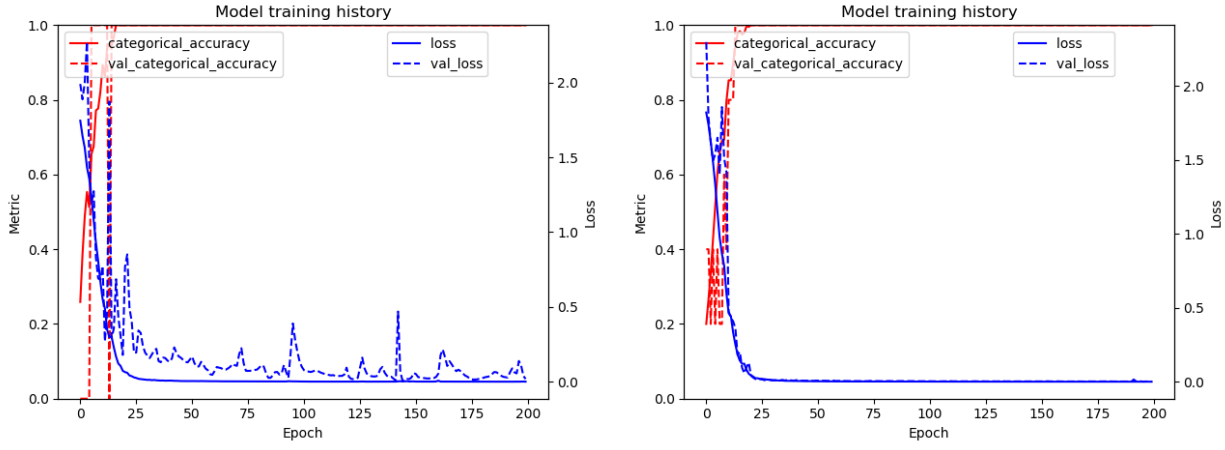
(a) CK+ dataset (b) OuluCasia dataset

Fig. 18: Confusion for Single Frame model with maximum probability



(a) CK+ dataset (b) OuluCasia dataset

Fig. 19: Confusion for Single Frame with summing probabilities

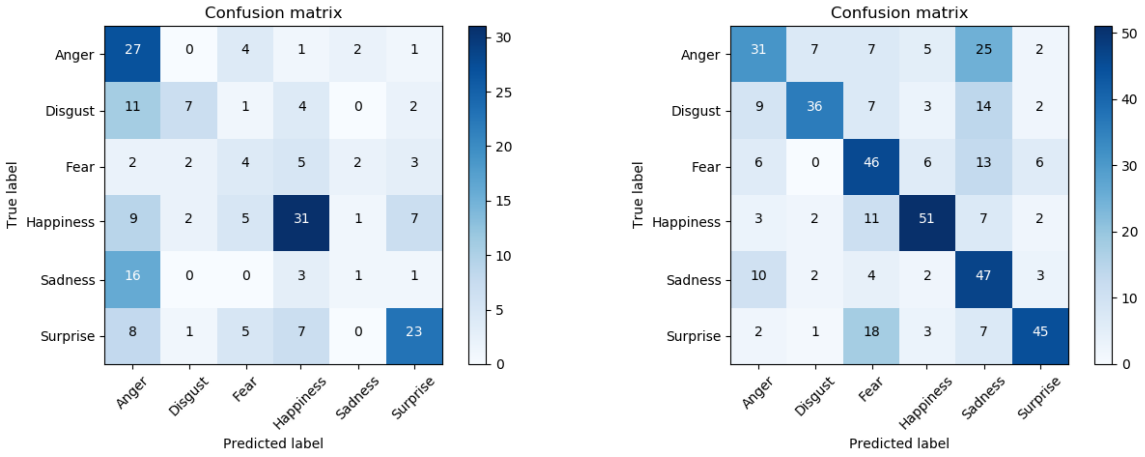


(a) CK+ dataset (b) OuluCasia dataset

Fig. 20: Training process for Early Fusion model

Inspecting the confusion matrices, Figure 21, we can see that in the case of Oulu-CASIA dataset it is similar to the one in

Single Frame model. In the case of CK+ dataset it has improved but the diagonal is still blurred.

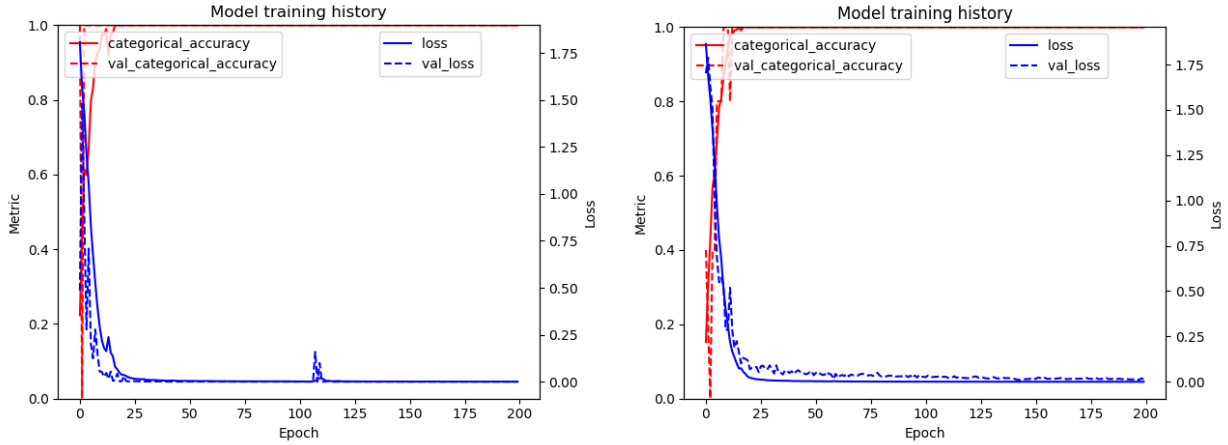


(a) CK+ dataset (b) OuluCasia dataset

Fig. 21: Confusion for Early Fusion model

3) Late Fusion: For the training phase the fifteenth frame beginning at the end and the last frame are used, these frames are selected to use the facial expression apogee and respect the window size. In the validation phase the same frames are used.

The training phase for this model is represented in Figure 22. Like in the previous case, Early Fusion, in this graphics we can see that both datasets have a similar behavior.



(a) CK+ dataset (b) OuluCasia dataset

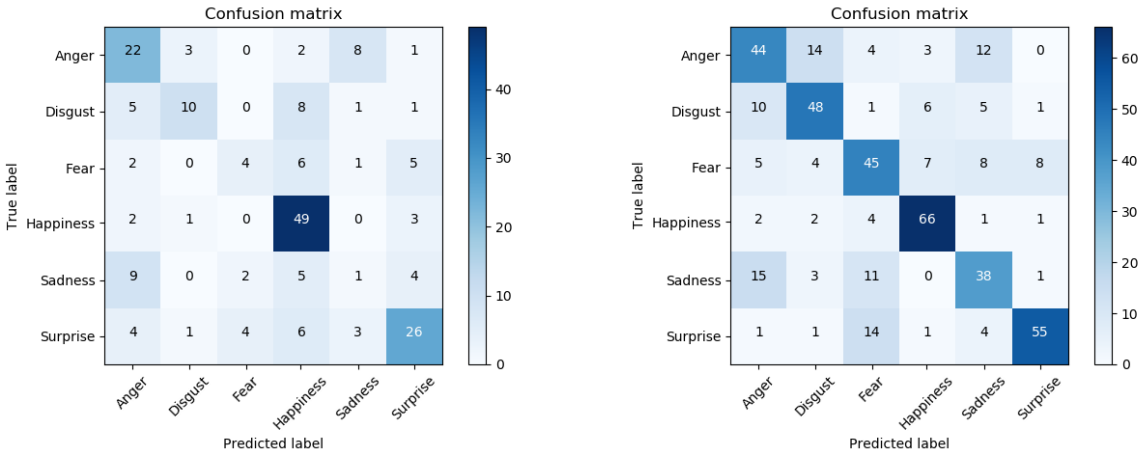
Fig. 22: Training process for Late Fusion model

The validation result, showed in Table IV, demonstrates again a better result for the Oulu-CASIA dataset. Although is not so differentiated in this case. Compared to the Early Fusion results, the improvement for the CK+ dataset is 0.093 (19.79%) and for the Oulu-CASIA dataset is 0.090 (15.65%).

Architecture	Validation accuracy	
	CK+ dataset	Oulu-CASIA dataset
Late Fusion	0.563	0.665

TABLE IV: Validation accuracies of late fusion model

Looking to the confusion matrices, Figure 23, in the CK+ dataset the diagonal is not so blurred, and for most of the emotions, the majority subjects are properly classified. In Oulu-CASIA dataset, the diagonal is even more marked than in previous models.

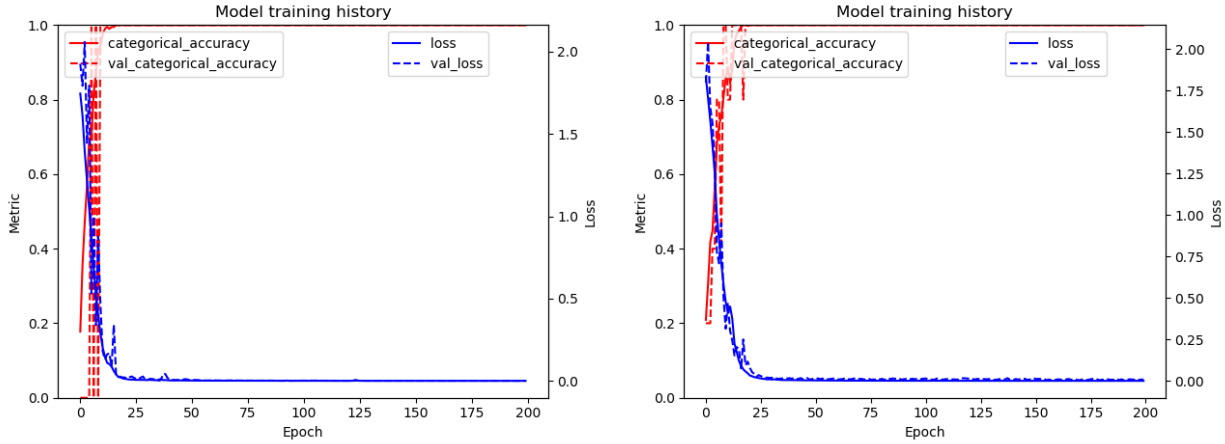


(a) CK+ dataset (b) OuluCasia dataset

Fig. 23: Confusion for Late Fusion model

4) Slow Fusion: For the training and validation phases the last ten frames are used, as in the Early Fusion case, but in this case the frames are grouped in four stacks of four frames each.

The history of the training phase is represented in Figure 24. As can be seen on the graphics, in this case the model learns faster with the CK+ dataset than with the Oulu-CASIA dataset.



(a) CK+ dataset (b) OuluCasia dataset

Fig. 24: Training process for Slow Fusion model

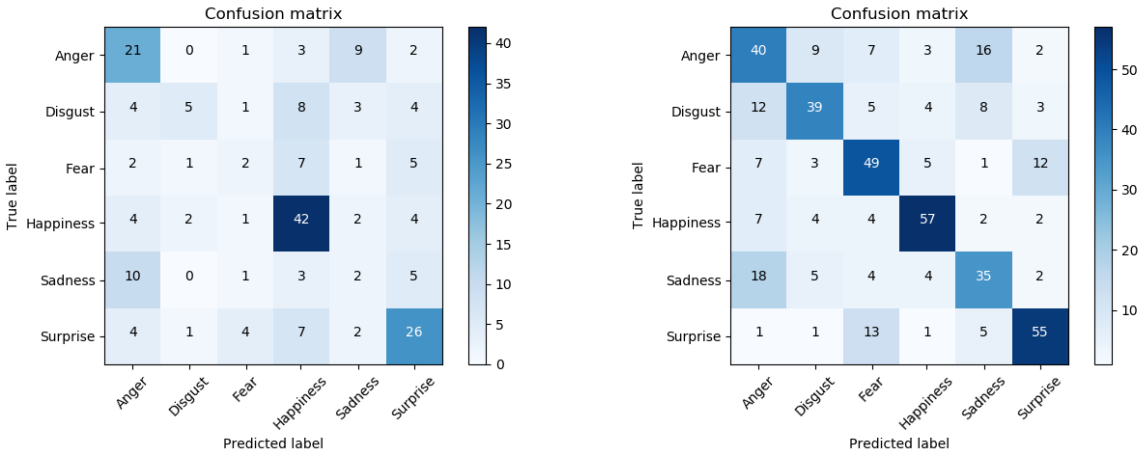
The validation accuracies, showed in Table V, manifest a deterioration with respect to the Late Fusion model in both datasets. Specifically, in CK+ dataset the accuracy is reduced by -0.071 (-12.61%) and in Oulu-CASIA dataset by -0.047 (-7.07%). Anyway, compared to the Single Frame model, the results are better. The improvement is 0.109 (28.16%) in CK+ dataset and 0.056 (9.96%) in Oulu-CASIA dataset.

Architecture	Validation accuracy	
	CK+ dataset	Oulu-CASIA dataset
Slow Fusion	0.492	0.618

TABLE V: Validation accuracies of slow fusion model

Looking to the confusion matrix, Figure 25, we can see that in the CK+ dataset the diagonal is marked for some emotions but with others is extremely blurred. In the Oulu-CASIA dataset, the diagonal is clear for all the facial expressions.

5) 3D Convolution: The training phase is carried out using the last five frame of each sequence. The validation phase is performed fitting the network with blocks of five frames with a stride of one frame and voting all the results to obtain the

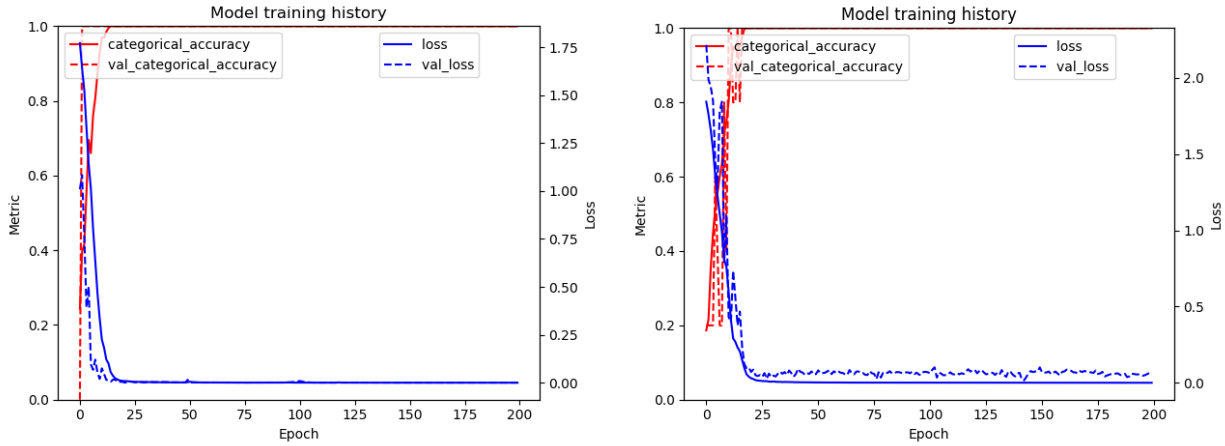


(a) CK+ dataset (b) OuluCasia dataset

Fig. 25: Confusion for Slow Fusion model

sequence corresponding emotion. The voting step is done using the same options than in the Single Frame model: maximum probability and summing probabilities.

The training phase for this model is represented in Figure 26. As in previous models, both datasets experiment a similar behavior during the learning process.



(a) CK+ dataset (b) OuluCasia dataset

Fig. 26: Training process for 3D Convolution model

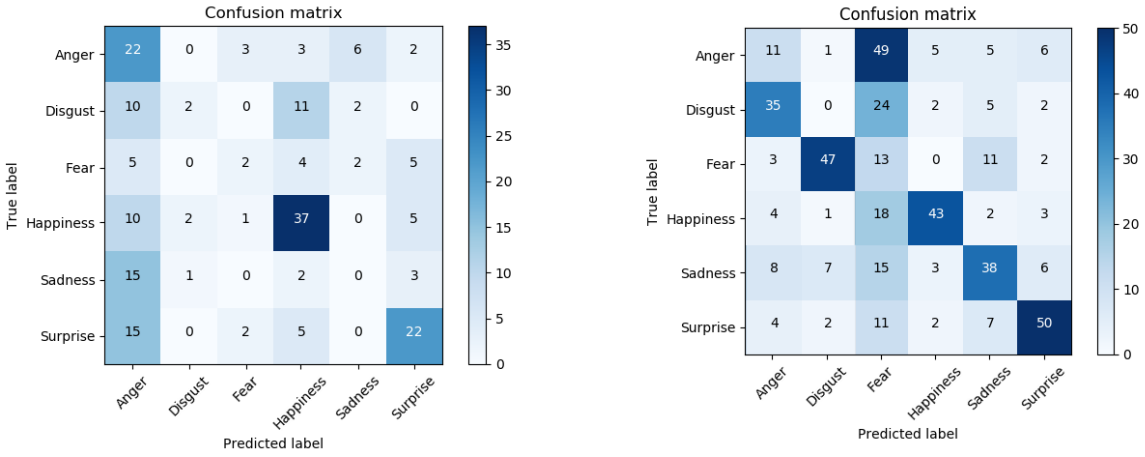
The validation result for both voting options and datasets is showed in Table VI. The performance of this model is very different between datasets, but in both cases the difference between voting options is small. Considering the CK+ dataset, the accuracy is higher for the maximum probability option while in the Oulu-CASIA dataset is higher for the summing probabilities option. Compared to previous models, the CK+ accuracy improves with respect to the Single Frame model, 0.079 (20.63%) and deteriorates with respect to the best time fusion model (Late Fusion), -0.101 (-17.94%). On the other hand, the Oulu-CASIA accuracy falls both compared to the Single Frame model, -0.187 (-33.27%), and the Late Fusion model, -0.29 (-43.61%).

Architecture	Validation accuracy	
	CK+ dataset	Oulu-CASIA dataset
3D Convolution with summing probabilities	0.462	0.375
3D Convolution with maximum probability	0.427	0.348

TABLE VI: Validation accuracies of 3D convolution model

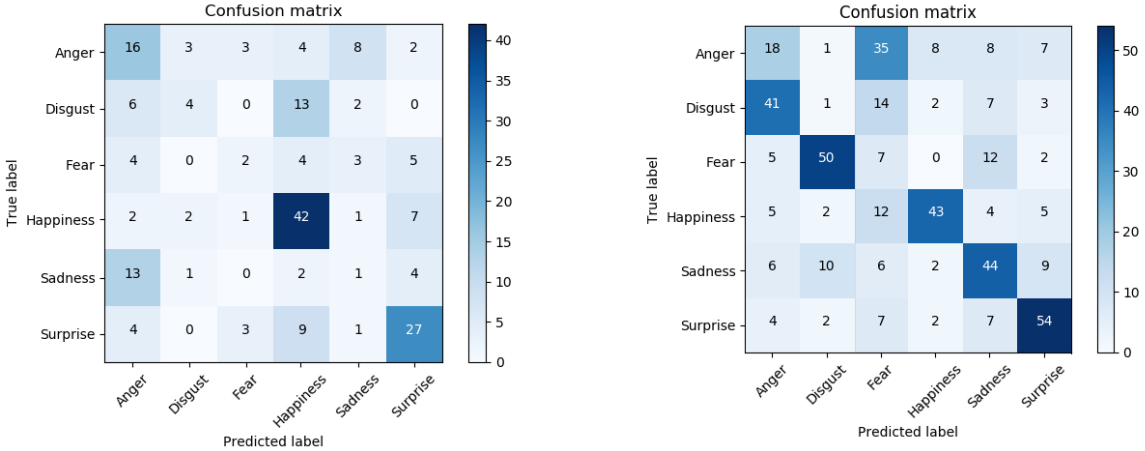
In order to get a more comprehensible validation metric, the confusion matrix is created. In this case, like in the single frame, the difference between the maximum probability option, Figure 27, and the summing probabilities one, Figure 28, is

irrelevant. In CK+ dataset, as the accuracy shows, the diagonal is blurred. In the Oulu-CASIA dataset the diagonal is marked for half of the facial expressions.



(a) CK+ dataset (b) OuluCasia dataset

Fig. 27: Confusion for 3D convolution model with maximum probability



(a) CK+ dataset (b) OuluCasia dataset

Fig. 28: Confusion for 3D convolution model with summing probabilities

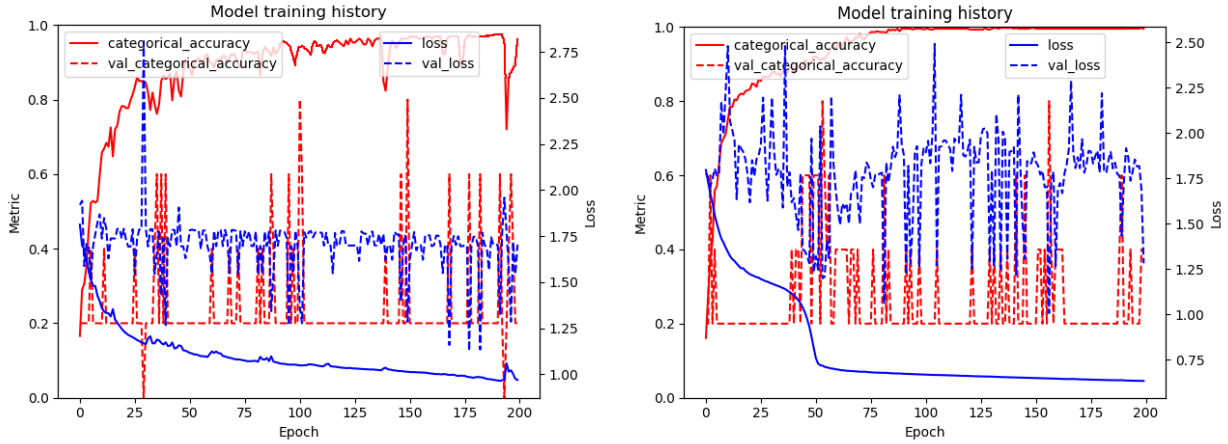
6) Recurrent Neural Network: Both training and validation phases are carried out using the last five frame of each sequence. Each frame fits one of the CNN and then, the output is analyzed by the GRU cell.

The training phase for this model is represented in Figure 29 where the history is depicted. In contrast to all the previous models, the validation curves do not follow the training pattern. This means that the model is not generalizing.

The validation accuracies for both voting options and datasets are showed in Table VII. Both in the CK+ dataset and the Oulu-CASIA dataset, the obtained accuracy is extremely bad. In CK+ dataset, compared to the Single Frame model the accuracy is reduced by -0.093 (-24.28%); compared to the Late Fusion model by -0.273 (-48.49%); and compared to the 3D convolution model by -0.172 (-37.23%). In Oulu-CASIA dataset, compared to the Single Frame model the accuracy is reduced by -0.282 (-50.18%); compared to the Late Fusion model by -0.385 (-57.89%); and compared to the 3D convolution model by -0.095 (-25.33%).

Architecture	Validation accuracy	
	CK+ dataset	Oulu-CASIA dataset
RNN	0.29	0.28

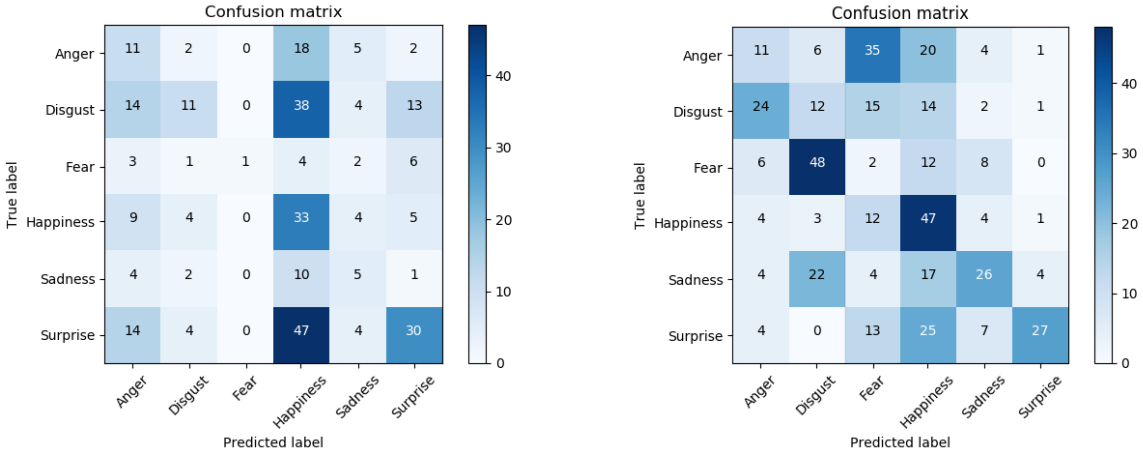
TABLE VII: Validation accuracies of RNN model



(a) CK+ dataset (b) OuluCasia dataset

Fig. 29: Training process for RNN model

As in the other models the confusion matrix is created, Figure 30. In both datasets the diagonal is completely lost. Taking into account the information extracted from the training history, these bad results were expected.



(a) CK+ dataset (b) OuluCasia dataset

Fig. 30: Confusion for RNN

7) Execution time: Apart from the results obtained for the different models, it is also important to take into account the execution time. In Table VIII, the time required for each subject during the training phase is detailed. As can be seen, the execution time for the Oulu-CASIA dataset is approximately twice the required time for the CK+ dataset. This is expected as the CK+ dataset has 198 sequences while Oulu-CASIA dataset has 445 sequences.

Another observation is that the time increases with the model complexity. As much operations must be done, more time is required, which is the logical relation.

Architecture	Training time per subject	
	CK+ dataset	Oulu-CASIA dataset
Single Frame	673.05 s	1675.67 s
Early Fusion	1176.95 s	2486.60 s
Late Fusion	1282.78 s	2708.49 s
Slow Fusion	1327.79 s	2815.70 s
3D convolution	2240.34 s	4908.47 s
Recurrent Neural Network	2088.28 s	4427.15 s

TABLE VIII: Training execution time per subject

VII. Conclusions

The goal of this project is the comparison of different video analysis techniques applied to emotion recognition. Considering the previous results, we can conclude that the introduction of temporal dimension improves the obtained results in most cases. This was the expected behavior as the facial expressions evolve over time.

The best results have been obtained with the Late Fusion model. Knowing that, the model has been trained with the global dataset (CK+ and Oulu-CASIA with all the illumination conditions). For the testing phase, a single Oulu-CASIA subject (with all the illumination conditions) has been used. The training process lasted 9575.46s for 1567 sequences and the testing process 0.97s for 18 sequences. The accuracy in the testing set is 66.67% and the confusion matrix is represented in Figure 31.

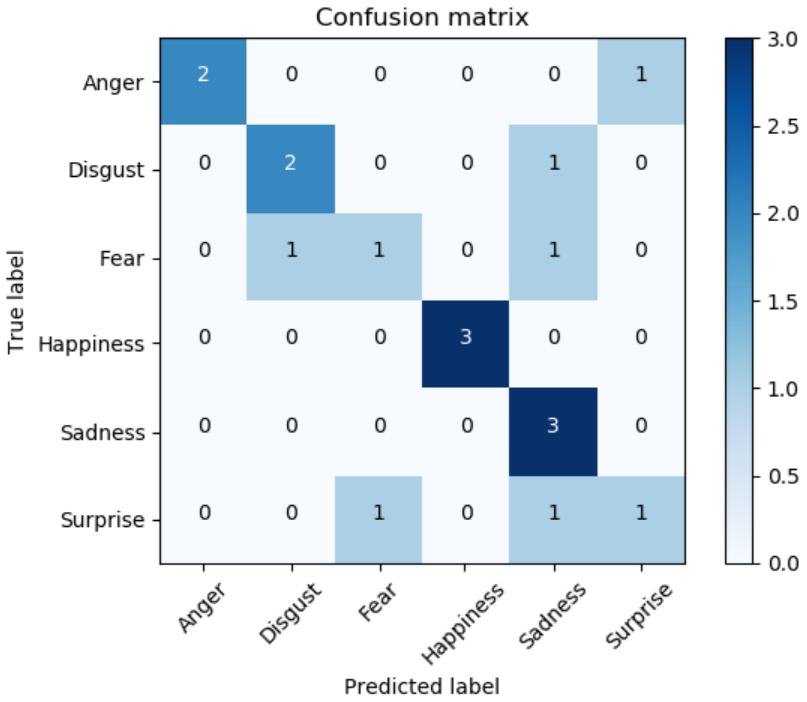


Fig. 31: Confusion for RNN

The final result is far from the state of the art results, which are around the 90%. It is worth mentioning that the goal of the project was the comparison of different architectures and no external data or architecture improvement was applied. For these reason, the final system of this project has a significant margin of improvement.

The face detector should be modified to improve the results. Some faces are wrongly detected; this fact introduces errors to the dataset which are propagated to the model. Moreover, in this project all the hyper-parameters have been fixed for comparison reasons; to improve the results, these parameters should be optimized.

Another possible improvement could be the introduction of a voting process at the end. Instead of using only two frames to determine the full sequence classification, repeat the prediction for different pairs of frames and then vote.

Finally, increasing the training dataset either with data augmentation or adding other databases, will help the model learning and will improve the obtained results.

Acknowledgment

The author would like to thank the contribution from both advisors, without them this project would not have been possible. Meysam Madadi and Ciprian Corneanu have also contributed to this project helping with the code programming and providing access to the datasets respectively. Finally, recognize the support received by family and friends during this project.

References

[1] Thibaud Senechal, Vincent Rapp and Lionel Prevost, "Facial feature tracking for emotional dynamic analysis," in *Advanced Concepts for Intelligent Vision Systems (ACIVS) 2011*, Berlin, Germany, November 2011, pp. 495–506. [Online]. Available: http://www.isir.upmc.fr/files/2011ACTI2001.pdf

[2] Samira Ebrahimi Kahou, Xavier Bouthillier, Pascal Lamblin, Caglar Gulcehre, Vincent Michalski, Kishore Konda, Sébastien Jean, Pierre Froumenty, Yann Dauphin, Nicolas Boulanger-Lewandowski, Raul Chandias Ferrari, Mehdi Mirza, David Warde-Farley, Aaron Courville, Pascal Vincent, Roland Memisevic, Christopher Pal and Yoshua Bengio, "Emonets: Multimodal deep learning approaches for emotion recognition in video," in *Computer Vision and Pattern Recognition (CVPR) 2015*, Boston, Massachusetts, USA, March 2015. [Online]. Available: https://arxiv.org/pdf/1503.01800.pdf

[3] Ali Mollahosseini, David Chan and Mohammad H. Mahoor, "Going deeper in facial expression recognition using deep neural networks," in *IEEE Winter Conference on Applications of Computer Vision (WACV), 2016*, Lake Placid, New York, USA, March 2016. [Online]. Available: https://arxiv.org/pdf/1511.04110.pdf

[4] Behzad Hasani and Mohammad H. Mahoor, "Spatio-temporal facial expression recognition using convolutional neural networks and conditional random fields," in *Computer Vision and Pattern Recognition (CVPR) 2017*, Honolulu, Hawaii, July 2017. [Online]. Available: https://arxiv.org/pdf/1703.06995.pdf

[5] Heechul Jung, Sihaeng Lee, Junho Yim, Sunjeong Park and Junmo Kim, "Joint fine-tuning in deep neural networks for facial expression recognition," in *IEEE International Conference on Computer Vision (ICCV), 2015*, Santiago, Chile, December 2015. [Online]. Available: https://pdfs.semanticscholar.org/4e02/f986906364c94fd4269270a893069570465d.pdf

[6] Mengyi Liu, Shaoxin Li, Shiguang Shan, Ruiping Wang and Xilin Chen, "Deeply learning deformable facial action parts model for dynamic expression analysis," in *Asian Conference on Computer Vision (ACCV) 2014*, Singapore, November 2015. [Online]. Available: https://pdfs.semanticscholar.org/b6c5/3891dff24caa1f2e690552a1a5921554f994.pdf

[7] Samira Ebrahimi Kahou, Vincent Michalski, Kishore Konda, Roland Memisevic and Christopher Pal, "Recurrent neural networks for emotion recognition in video," in *ICMI '15 Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, Seattle, Washington, USA, November 2015, pp. 467–474. [Online]. Available: http://www.professeurs.polymtl.ca/christopher.pal/RNN-emotions-kahou.pdf

[8] Linlin Chao, Jianhua Tao, Minghao Yang, Ya Li and Zhengqi Wen, "Multi-scale temporal modeling for dimensional emotion recognition in video," in *AVEC '14 Proceedings of the 4th International Workshop on Audio/Visual Emotion Challenge*, Orlando, Florida, USA, November 2014, pp. 11–18. [Online]. Available: http://speakit.cn/Group/file/2014_Multimodal%20Emotion%20Recognition_ACM%20MM_EI.pdf

[9] Tong Zhang, Wenming Zheng, Zhen Cui, Yuan Zong and Yang Li, "Spatial-temporal recurrent neural network for emotion recognition," in *IEEE Transactions on Cybernetics*, June 2018. [Online]. Available: https://arxiv.org/pdf/1705.04515.pdf

[10] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.

[11] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar and Li Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR) 2014*, Columbus, Ohio, USA, June 2014. [Online]. Available: https://cs.stanford.edu/people/karpathy/deepvideo/deepvideo_cvpr2014.pdf

[12] Sepp Hochreiter and Jrgen Schmidhuber, "Long short-term memory," in *Neural Computation*, November 1997, pp. 1735–1780.

[13] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk and Yoshua Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Conference on Empirical Methods on Natural Language Processing*, October 2014.

[14] Chollet, François and others, "Keras," https://keras.io, 2015.

[15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Vigas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu and Xiaoqiang Zheng, "Tensorflow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[16] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley and Yoshua Bengio, "Theano: A cpu and gpu math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, 2010. [Online]. Available: http://www.iro.umontreal.ca/~lisa/pointeurs/theano_scipy2010.pdf

[17] Guido van Rossum, "Python," https://www.python.org, 1991.

[18] Patrick Lucey, Jeffrey F. Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar and Iain Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2010*, San Francisco, California, USA, June 2010, pp. 94–101. [Online]. Available: http://www.pitt.edu/~emotion/ck-spread.htm

[19] Matti Taini, Guoying Zhao, Stan Z. Li and Matti Pietikainen, "Facial expression recognition from near-infrared video sequences," in *19th International Conference onPattern Recognition (ICPR), 2008*, Tampa, Florida, USA, December 2008, pp. 607–619. [Online]. Available: http://www.cse.oulu.fi/CMV/Downloads/Oulu-CASIA

[20] Puttemans Steven, Can Ergun and Toon Goedeme, "Improving open source face detection by combining an adapted cascade classification pipeline and active learning," in *12th International Conference on Computer Vision Theory and Applications*, February 2017.

[21] Gary Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[22] Jerome Friedman, Trevor Hastie and Robert Tibshirani, "Additive logistic regression: a statistical view of boosting," in *The Annals of Statistics*, 2000, p. 337374.

[23] Timo Ojala, Matti Pietikinen and David Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *12th IAPR International Conference on Pattern Recognition (ICPR 1994)*, 1994.

[24] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations*, San Diego, 2015. [Online]. Available: https://arxiv.org/pdf/1412.6980.pdf