Emotion recognition using Convolutional Neural Network

Susanna Brambilla
940755

Abstract—This paper describes a project that applies Convolutional Neural Networks (CNNs) to the purpose of ricognizing emotions in Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) dataset.

To address the problem, two types of data were considered: visual and acoustic. After extracting the features from the audio and video data, several models were made: initially a CNN for audio data only and one for video data only, finally the two CNNs were merged into a single model with the aim of improving the effectiveness of the classifier. Since human emotions are expressed through both audio and video information, the combination of this two types of data should achieve better performance.

The language Python with the support of its libraries was used for the developement of the project.

1 Introduction

This project addresses the study of emotion recognition using Convolutional Neural Networks (CNN). The models built was tested on audio and video data and have the purpose of discriminating between eight different emotions: neutrality, calm, happiness, sadness, anger, fear, disgust, surprise. These emotions correspond to those proposed as primary by P. Ekman, with the addition of calm and neutrality.

CNNs are a class of models applicable to tasks such as image classification, as they use filters that preserve the spatial characteristics of pixels by extracting salient features from the image. They apply convolution and subsampling operations to the images and create a more useful representation of them, identifying, for example, outlines, lines and shapes.

The study on audio data focuses on how to determine a person's emotional state based on vocal characteristics: that is, not by what the person says but by how s/he says it. About videos, faces were taken into consideration, then facial expressions.

Importance of the problem: The presence of software able to recognize and evaluate human emotions is increasingly important in various fields, for example in the fields of health, safety, education and entertainment. Emotion recognition can be useful in helping computers have the ability to interact more naturally and intelligently with people.

S. Brambilla, A/A 2019-2020, Universitá degli Studi di Milano, via Celoria 18, Milano, Italy
E-mail: susanna.brambilla@studenti.unimi.it

Images and sounds are the main channels through which people acquire information. Paralinguistic features and facial expressions can help understand a person's mental state (emotion). Information concerning this type of characteristics can be extracted from audio and video data.

Approach adopted: In the project, after a preprocessing phase, different methods of audio and video feature extraction were implemented. In the case of audio data, the effect of two different types of features was compared: Log-Mel Spectrogram and Mel-Frequency Cepstral Coefficients (MFCCs). A pre-trained model based on the VGG19 architecture (https://github.com/WuJie1010/g Facial-Expression-Recognition.Pytorch.git) was used to extract the facial features from the videos.

Later, Pytorch-based CNNs were used and have the task of classifying and recognizing emotions. The subject's emotional state is initially determined using two separate sources of information: audio and video. The results are compared to evaluate the most efficient resource for the recognition of emotions. Three different models were developed, all using a CNN. The first model was trained using only the voice characteristics (features), the second model using only the facial expressions captured in the videos and the last one using both types of data concatenating them.

Contributions: The following study aims to:

- compare the effectiveness of different types of features for the description of emotion;
- use and analyze the accuracy of CNNs to address the problem of emotion recognition;
- test architectures based on different types of data to analyze the accuracy of the classification in different cases.

2 State of the art

There are several works that analyze the recognition of emotions through Neural Networks using audio and video data, an important source used to face the problem is the thesis of Salem Bin Saqer AlMarri [1].

To choose the audio features to extract, the paper by K. Venkataramanan and HR Rajamohan [2], which studies the recognition of emotions in spoken language, was considered. For videos it has been useful to analyze the work by Z. He, T. Jin , A. Basu, J. Soraghan, G. Di Caterina and L. Petropoulakis [3], which deals with video preprocessing in

this area. The article by C Shorten and T. M. Khoshgoftaar cite shorten2019survey provided research on video data augmentation.

3 Theoretical model

Artificial neural networks are mathematical models that are inspired by biological neural networks and are used to solve Machine Learning problems related to different technological fields, using a learning method that exploits mechanisms similar to those of human intelligence.

The basic unit of computation in the neural network is the neuron and knowledge is stored in the parameters of the network. Each input to the network has an associated weight, assigned on the basis of its relative importance with respect to the other inputs. Each neuron applies a function to the weighted sum of its inputs. There is also another parameter b, bias, whose main function is to provide each node with a constant value to learn. The f function is non-linear and is called the activation function. The output of each neuron is sent to the other nodes or to the output of the network. Indicating with $\boldsymbol{x} \in R^n$ the input vector, with $\boldsymbol{w} \in R^n$ the weight vector, with y the output of the neuron and with f the activation function, the basic structure of the neuron can be formally represented as in Eq. 1:

$$y(x) = f(\sum_{i=1}^{n} w_i x_i + b) \tag{1}$$

The multilayer perceptron (MLP) is a neural network model that has at least one hidden layer, where the processing of information received from the input layer takes place, which is then sent to the output node. For the learning process the back-propagation algorithm is used, which works in an iterative way, modifying the weights of the connections between nodes until the optimal result is achieved. In Fig. 1 the basic architecture of an MLP with a single hidden layer is shown.

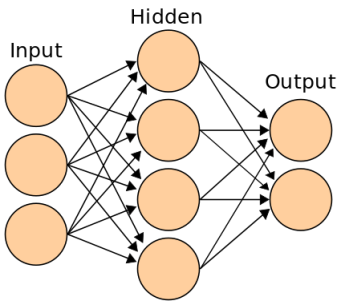


Fig. 1: Basic architecture of a neural network with one input layer, one hidden layer and one output layer.

In classification problems, the input usually consists of a vector of features and the output is a discrete-value variable that indicates membership in a class.

3.1 Training

In supervised learning, the dataset is labeled: the network is supplied with a set of inputs which correspond to known outputs (labels). The network learns the link between inputs and outputs so that it can infer correct associations on unknown inputs.

The first step in learning in a neural network is the forward step, from the input nodes, through the hidden nodes to the output nodes. Weights are initially assigned randomly. During the forward-propagation process the CNN output is observed and compared to the desired (known) output. The error, which has to be minimized by adjusting the weights of the connections in the network, is propagated backwards through the error back-propagation algorithm, to the previous level. In this process, the gradient of the error function is calculated, based on the rules of derivation of complex functions. The gradient shows how much the value of the weights must change, in a positive or negative direction, to minimize the error. Once the error is calculated, the weights are adjusted appropriately. The weight adjustment at each iteration through the gradient descent algorithm is defined in Eq. 2, where $\nabla E(\boldsymbol{w^k})$ is the gradient of the error E in the weight vector for the current iteration $\boldsymbol{w^k}$ and the scalar $\eta > 0$ is the learning rate and defines the step towards the minimum.

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \eta \nabla E(\boldsymbol{w}^k) \tag{2}$$

This process repeats until the error is lower than a certain threshold. After learning the features in the previous levels we move on to the classification. The last level of the network is fully connected and can use the Softmax activation function. The Softmax function, shown in Eq. 3, generates a vector of size equal to the number of classes between which the network must discriminate, which contains the probability of each data to belong to a given class. It squeezes an input vector $\boldsymbol{x}$, n-dimensional, of real values into a vector $softmax(\boldsymbol{x})$, n-dimensional, of values in a range $(0, 1)$ whose sum is 1.

$$softmax(\boldsymbol{x})_i = \frac{e^{x_i}}{\sum_{n=1}^{N} e^{x_n}} \tag{3}$$

For a multiclass classification problem it is often used the Negative Log-Likelihood, which must be minimized, as a cost function to be optimized. When the algorithm ends the model has been learned and can be tested on new unknown outputs.

The whole learning process includes the following steps:

1) inizialize weights with random values;
2) the network receives an input, performs the forward step and calculates the probabilities of each input to belong to each class;
3) since you work with the training set in a supervised way, the correct output is known: the error is calculated as the difference between the correct output and the output provided by the model;
4) back-propagation of the error to calculate the gradient of the error with respect to the weights in the network and minimization of the output error through an optimization algorithm;
5) steps 2-4 are repeated until convergence.

3.2 Regularization

One problem that can arise in a classification model is the overfitting. This phenomenon occurs when the model fits

the observed data and is unable to generalize to new data providing low performance on the test set. It can occur when the number of training examples is limited. Regularization techniques can be used to reduce this kind of problems. These techniques operate on the cost function and generally involve the addition of a penalty factor on the parameters.

Some possible solutions to overfitting:

1) *Dropout*: at each iteration allows to randomly select some neurons and exclude them together with their incoming and outgoing connections. Dropout prevents over-adaptation of neurons. The application of this technique during learning produces at each iteration a different reduced network of the starting model, made up of nodes that have not been temporarily excluded. The probability of choosing how many nodes to be removed is specified as a hyperparameter, and is $p > 0$.
2) *Data augmentation*: consists of increasing the size of the dataset by generating new synthetic data.
3) *Early stopping*: stops the learning when the model performance on the validation set starts to deteriorate or remains constant.

3.3 Convolutional Neural Networks

CNNs are particularly useful and efficient in learning the content of an image, video or voice signal through the image of their waveform.

Like an MLP, a CNN is composed of an input layer, one or more hidden layers, which perform calculations using activation functions, and an output block. The difference is the presence of convolution levels, where the process is optimized by using convolution filters to extract the analyzed features. CNN learns and classifies based on these characteristics.

The most common levels in a CNN are:

- *input*: to this level is passed the data vector representing the input;
- *convolutional*: in these levels each neuron calculates the product between its weights and the region to which it is connected with the other neurons. The convolutional filter activates the features while preserving the spatial relationships: the filter moves along the surface of the input producing an activation map;
- *activation*: uses a function that introduces a non-linearity into the network. An example is the ReLU, represented in Eq.4, which replaces all negative values in the feature map with zeros and keeps positive values;

$$f(x) = max(0, x) \tag{4}$$

- *pooling*: the pooling level performs a down-sampling operation along the spatial dimensions, reducing the number of parameters that the network must learn and therefore the computational load;
- *fully-connected*: level in which all neurons are connected, it operates on the output volume of the preceding layer and provides a vector useful for the final classification.

4 Simulation and Experiments

The audio and video data are initially manipulated in order to be given as input to CNNs. Several tests were carried out to understand which audio features are most useful for the purpose of emotion recognition. The aim of the project is to analyze accuracy using three models: a CNN for audio data, a CNN for video data and a CNN using both data. After having trained the three models with their respective training sets, accuracy has been evaluated using their respective test sets.

4.1 Dataset

The dataset used is the *Ryerson Audio-Visual Database of Emotional Speech and Song* (RAVDESS [4]): a multi-modal and validated database. RAVDESS consists of audio and video data; 24 professional actors (12 men and 12 women) perform vocalizations with emotions that include: happiness, sadness, anger, fear, surprise, disgust, calm and neutrality. Each actor says two sentences for each emotion. These phrases are in discursive form or in the form of songs and recorded in different levels of intensity.

Regarding the audio data, we have:

- 1440 speech files;
- 1012 song files.

Regarding the video data, we have:

- 2880 speech files (in audio-video and video only mode);
- 2024 song files (in audio-video and video only mode).

The collection contains 7536 files overall.

For the purpose of this project, audio data and video data without audio were used, with the aim of distinguishing between eight different emotions.

4.2 System architecture and implementation details

The general and complete structure of the emotion recognition system is represented in Fig. 2.

Preprocessing e data augmentation: Both types of data were preprocessed before the training phase: an exploration of methods for extracting features was carried out. Before extracting the features, a data augmentation procedure has been necessary to reduce the problem of overfitting. More samples were generated via data augmentation because the classes from the original dataset are unbalanced, in order to equally distribute the data. The steps taken in analyzing the audio data are:

- load the file into an array using Librosa (with sampling rate = 44100 Hz), so that each audio lasts 5 seconds;
- data augmentation procedure consisting of three operations: adding noise, changing the pitch, random shifting. In this way new files were synthetically generated to add to the dataset;
- class balancing;
- extraction of MFCCs and log-Mel features, which can be represented as images, as you can see in Fig. 3.

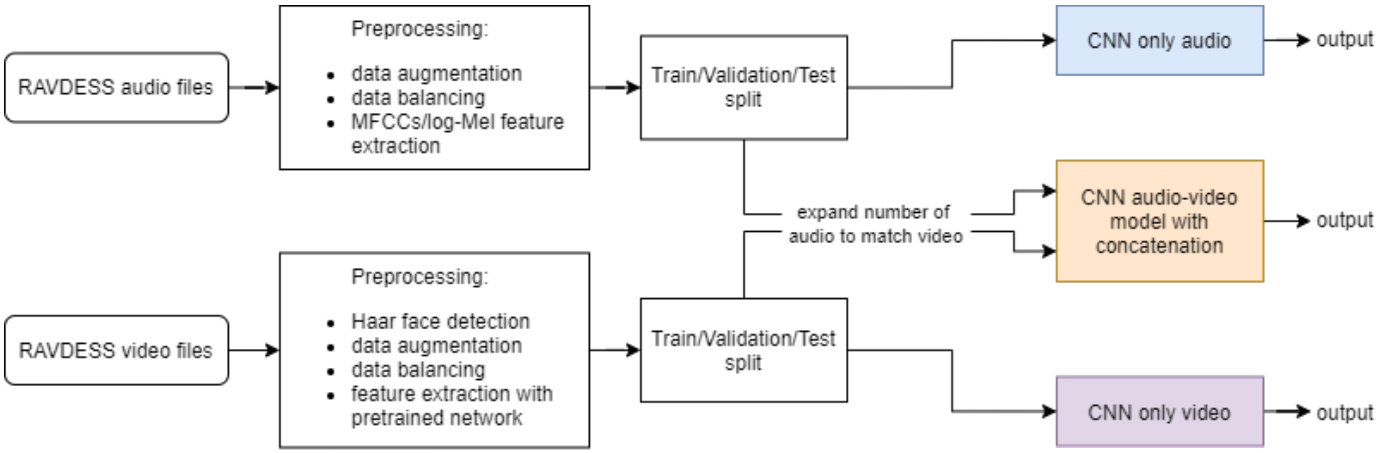


Fig. 2: General structure of the system.

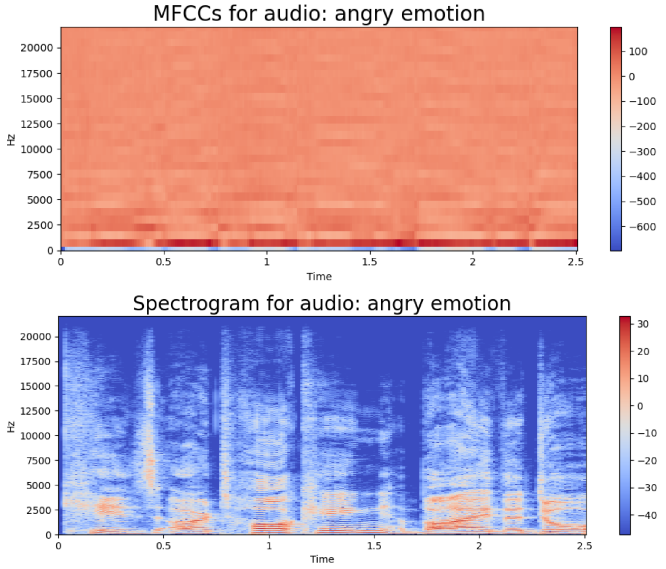


Fig. 3: Spectrogram of the MFFCs and log-Mel features extracted from an audio belonging to the "Angry" class.

The steps taken in the video data analysis are:

- load the file and extract 32 consecutive frames from each video;
- face detection in each video using Haar's algorithm, which implementation details are discussed in Appendix A section, and cropping of localized region to avoid background noise. An example of face detection on 32 extracted frames is presented in Fig. 4;
- data augmentation procedure consisting of three operations: adding noise, changing the contrast, changing the gamma;
- class balancing;
- feature extraction via pretrained neural network.

Generation of sets: Both datasets were split between training, validation and test set. For the test set the actors 23 and 24 were used, for the validation set the 21 and 22 and all the others belong to the training set. This subdivision allows to obtain more objective results avoiding that the model learns specific characteristics of an actor. All data were normalized using mean and standard deviation.

CNN architecture: The CNNs for the different types of data were chosen with simple architectures and more complex architectures did not show significant improvements. The two models, one implemented for audio features and one implemented for video features, were used as the basis for the final architecture that combines them.

- CNN for audio only and video only: the input for the audio network consists of a vector of dimensions [32, 1, 128, 216] in the case of features Mel and a vector of dimensions [32, 1, 30, 216] in the case of MFCCs, where the dimensions represent [batch size, channels, Mel bands/MFCCs, audio length]. CNN input for video is a vector of sizes [32, 1, 32, 512], where the sizes represent [batch size, channels, number of frames, feature space size].
  Individual networks built for audio data only and video data only have the same structure. They consist of a convolutional layer with 16 filters, a size kernel of size 3, a stride of 1 and zero padding. The convolutional level is followed by a ReLU activation function and a 2D max pooling level. For both of them the dropout was applied with a probability of 0.5 and finally there is a fully-connected level that provides the output.
- Joint model: to ensure the one-to-one mapping between the two types of data it was necessary a step in order to obtain for each class the same number of audio and video features: the audio files were multiplied so that their number coincides with the number of video files. The inputs supplied to the network belong to the same class (if the audio belongs to the "Sad" class, the corresponding video also belongs to the "Sad" class). The outputs of the two CNNs represented above (excluding the fully connected layer) have been concatenated and the resulting vector passed into a fully connected layer.

The parameters are shared by all the models: the Adam algorithm is used for the optimization, the learning rate is 1×10^{-5} and the cost function is PyTorch's CrossEntropy-

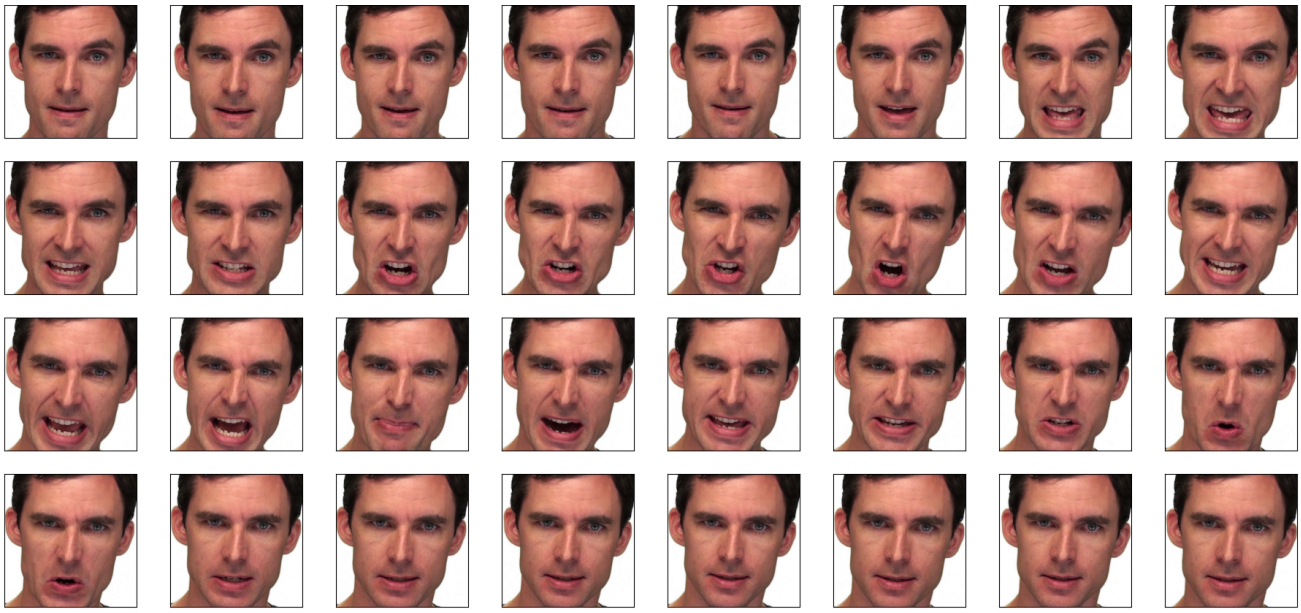


Fig. 4: An example of 32 frames extracted from a video belonging to the ”Angry” class. For each frame the face was revealed with the Haar Cascade algorithm, then cropped to obtain the final image.

Loss, which combines Log-Softmax with Negative Log-Likelihood.

To further counter the problem of overfitting it was useful to use the Early Stopping technique, as implemented by Bjarten (available at the link: https://github.com/anshulrai/early-stopping-pytorch), in which the validation loss is tracked during the learning process. A checkpoint is saved every time the validation loss decreases. The network checkpoint is used to evaluate performance on the test set. The *patience* argument allows to specify after how many epochs without improvement of the validation loss we want to stop the training process. In this case 10.

5 Results

The audio-only and video-only models were initially trained using only data from the original dataset. The results showed a strong overfitting problem, which is the reason why it was decided to use data augmentation and regularization strategies.

The following feature vectors were provided as input at various times to test the CNN on the single audio data:

- MFFCs features for the prediction of the 8 classes;
- log-Mel features for prediction of the 8 classes.

To balance the data and allow the network to learn and correctly classify in the right way the classes in which the number of examples provided was smaller (in particular ”Surprise”, ”Neutral”, ”Disgust”), a further data augmentation was carried out exclusively for the above classes, in order to obtain 1504 examples for each class. The MFFCs and log-Mel features were then extracted again on the dataset with balanced classes and the resulting vectors supplied as input to CNN. The best result, evaluated on the corresponding test set, is obtained using the log-Mel features on the balanced dataset: in Fig. 5 the accuracy values of the training set and of the validation set for different epochs are shown.

The Fig. 7 shows the confusion matrices, normalized between 0 and 1, obtained by supplying to the model the various inputs belonging to the test set. These matrices allow you to understand in detail how accuracy is distributed between classes. After balancing the data it can be seen an improvement in accuracy, particularly in the ”Surprise”, ”Disgust” and ”Neutral” classes.

The video model was trained with 32 consecutive frames extracted from each example. Also in this case, the tests carried out took into consideration the original dataset and the dataset with the balanced classes through data augmentation (the number of examples present in the ”Surprise”, ”Neutral”, ”Disgust” classes was lower). The graph in Fig. 8 shows the accuracy of the training set and the validation set of the best result obtained, evaluated for different epochs.

The normalized confusion matrices, which show the results achieved using balanced and unbalanced data from the test set, are represented in Fig. 9.The accuracy evaluated on the test set shows an increase after the classes have been balanced, especially in the three classes indicated above.

The Tab. 1 shows the results of the experiments carried out using only audio data and only video data.

	unbalanced	balanced
audio MFCCs	43	47
audio log-Mel	45	49
video 32 frames	51	53

TABLE 1: Percentage of accuracy using different feature vectors as input for audio-only and video-only CNNs. In red the best result for audio and in green the best result for video.

Increasing the depth of both networks by gradually adding convolutional layers did not show significant improvements in model performance.

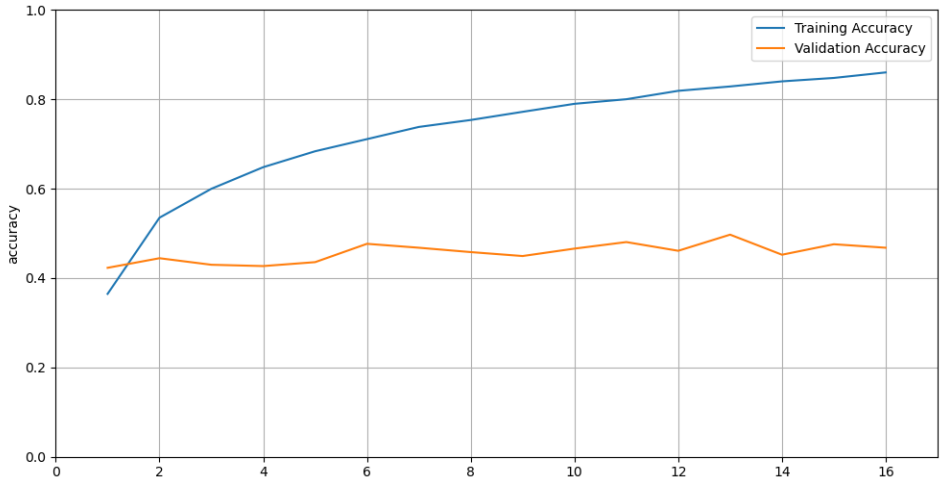
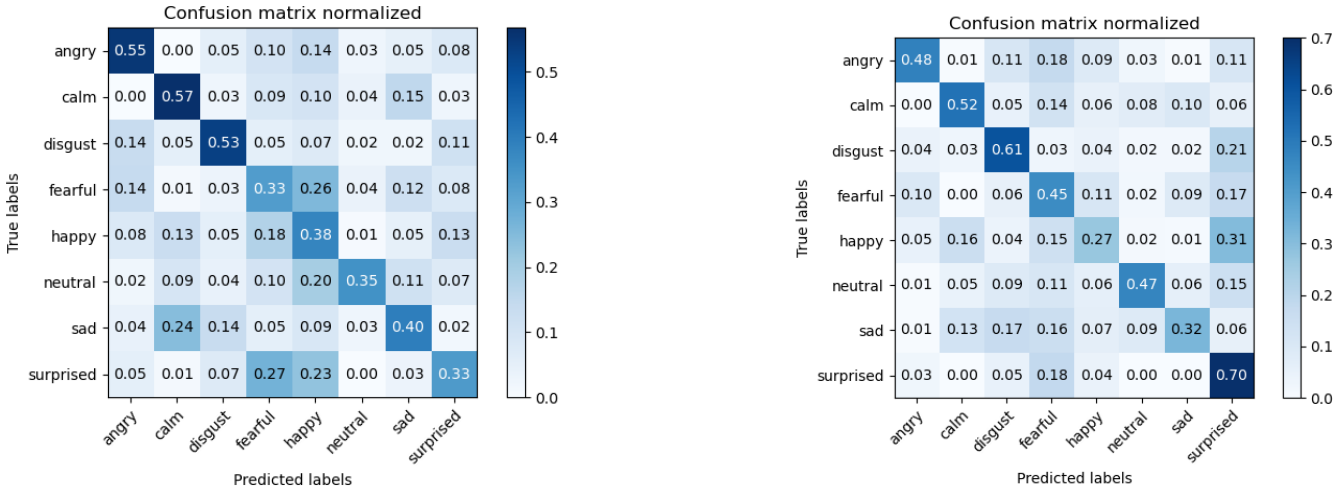


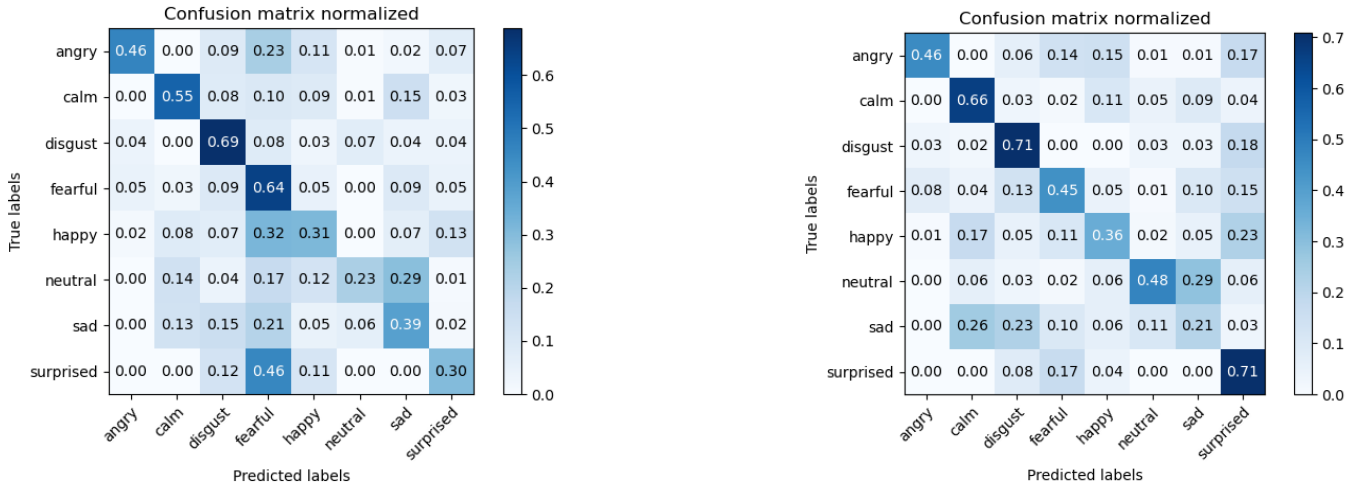
Fig. 5: Accuracy on audio training and validation set using the model that showed the best performance.



(a) Confusion matrix obtained with unbalanced data.

(b) Confusion matrix obtained with balanced data.

Fig. 6: Confusion matrices obtained using the MFCCs feature test set with the audio CNN model.



(a) Confusion matrix obtained with unbalanced data.

(b) Confusion matrix obtained with balanced data.

Fig. 7: Confusion matrices obtained using the log-Mel feature test set with the audio CNN model

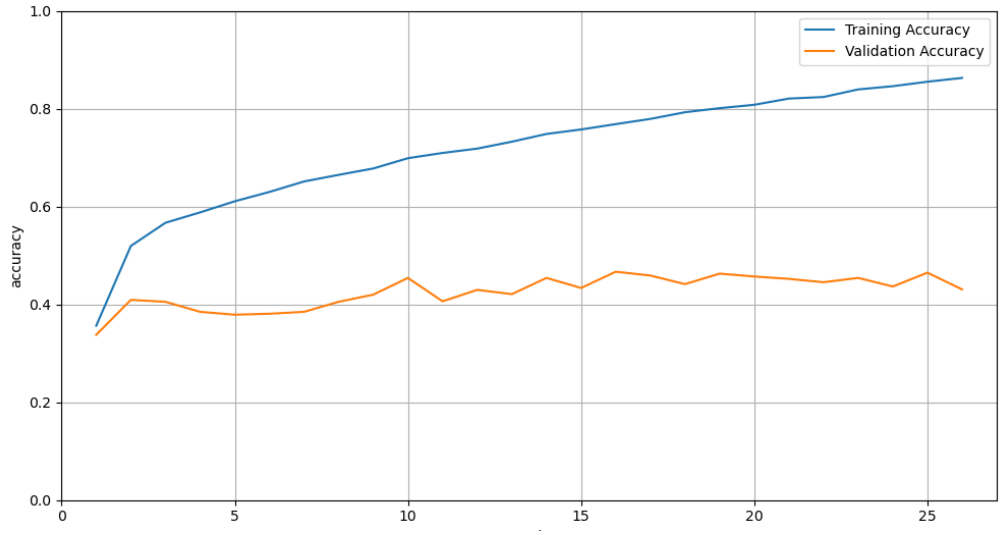
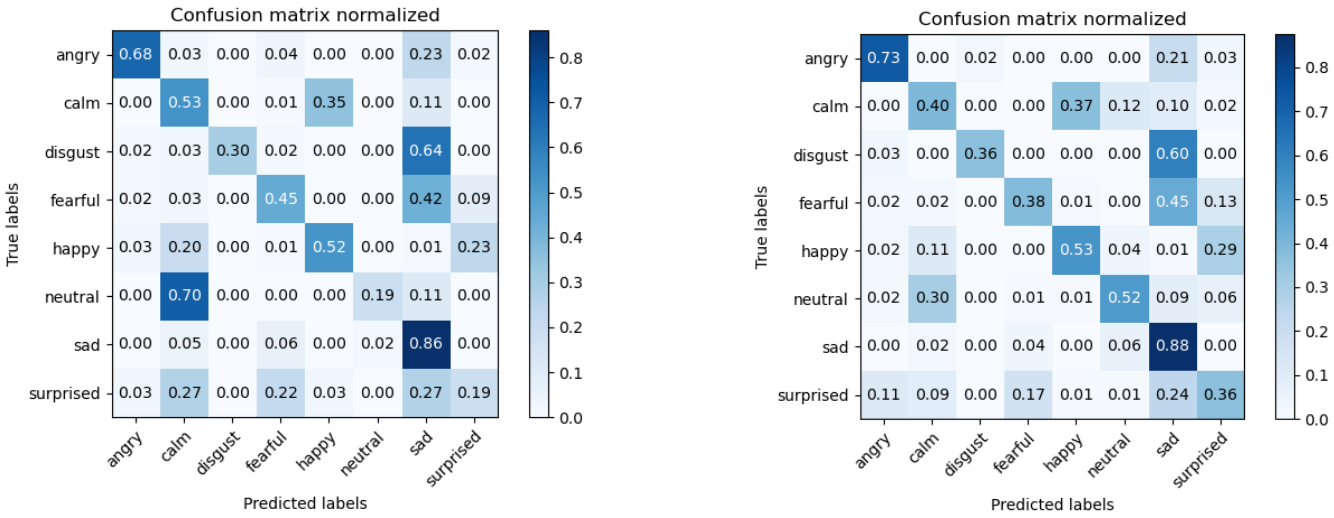


Fig. 8: Accuracy on training and validation video sets using the model that showed the best performance.



(a) Confusion matrix obtained with unbalanced data.

(b) Confusion matrix obtained with unbalanced data.

Fig. 9: Confusion matrices obtained using the test set with video model with 32 consecutive frames.

The selected models with the features that achieved the best performance were used to build the integrated CNN with audio-video input. Fig. 10 shows the progress of the learning process of this architecture. The model with concatenation of audio and video features reaches an accuracy of **67%** measured on the test data, the resulting confusion matrix is shown in Fig. 11.

As assumed, it can be seen that providing audio-video information by combining the data allows the network to learn better and easily offer predictions on new data, compared to those offered using the two modalities individually.

In all the models we can see a strong discrepancy between accuracy evaluated on the validation set and accuracy evaluated on the training set. This difference indicates the presence of overfitting, revealing the need for a further increase in the size of the dataset and/or more regularization.

6 Comments

From the results obtained we can conclude that, since audio and video contain different information, concatenating the feature vectors of the two types of data helps in the goal of classifying emotions, facilitating the learning process for the network.

Despite an extensive regularization process all the presented models show a strong tendency to overfitting. One question that arises is: are the learned features generic enough to extend the model to a different dataset? The dataset used was created using actors who speak exclusively in English (North American). The model was then trained to identify specific features of a given language in an unnatural context. Combining RAVDESS with other datasets with similar characteristics could be useful to help in generalization and further decrease overfitting.

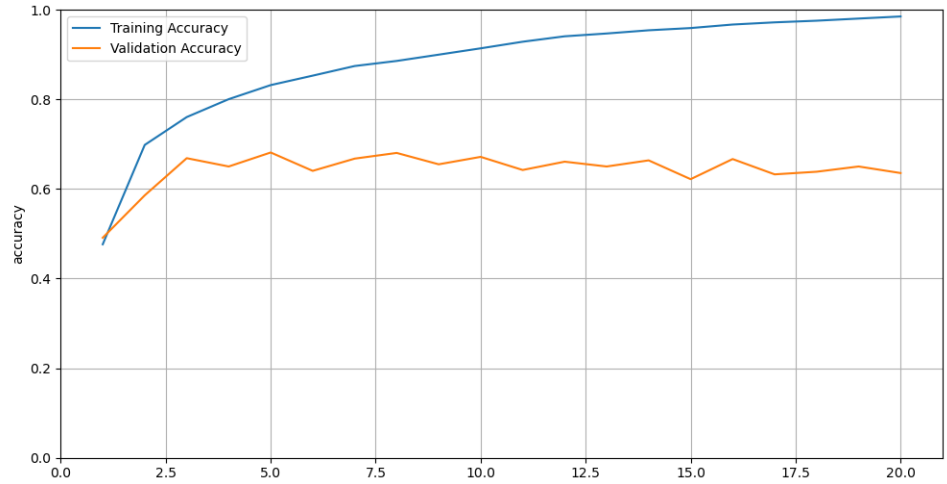


Fig. 10: Accuracy on audio-video training and validation set using the model that showed the best performance.

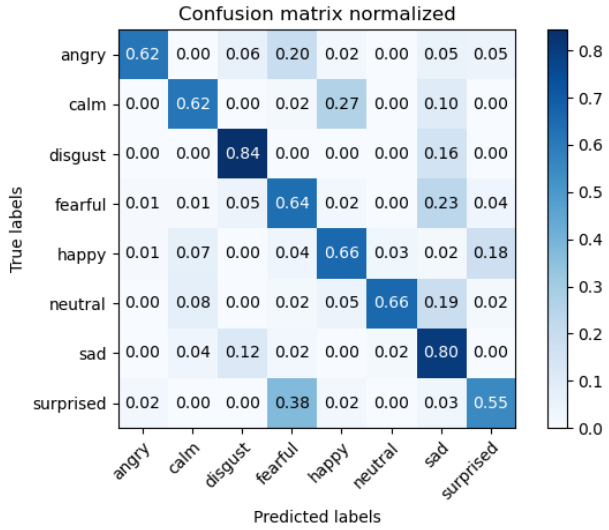


Fig. 11: Confusion matrix obtained using audio and video data together.

APPENDIX A
NOTE ON HAAR-CASCADE

This section shows the piece of code related to the Haar-Cascade classification algorithm offered by OpenCV.

```
#for each frame img upload the file
#containing the classifier data
face_cascade = cv2.CascadeClassifier(
   'haarcascade_frontalface_default.xml')
#convert the image to grayscale
gray = cv2.cvtColor (img, cv2.COLOR_BGR2GRAY)
#apply the classifier to the image where:
# 1.1 is the scale factor;
# 5 is the number of neighboring regions that
# the algorithm must consider for each
# of the areas recognized as faces
face = face_cascade.detectMultiScale(
                          gray, 1.1, 5)
# draw the rectangle with coordinates
#x, y, w, h that contains the identified faces
for (x, y, w, h) in face:
    cv2.rectangle (img, (x, y), (x+w, y+h),
                    (255, 255, 255), 1)
#crop the face contained in the image
img = img[y:y+h, x:x+w]
```

REFERENCES

[1] S. B. S. AlMarri, "Real-time facial emotion recognition using fast r-cnn," 2019.

[2] K. Venkataramanan and H. R. Rajamohan, "Emotion recognition from speech," 2019.

[3] Z. He, T. Jin, A. Basu, J. Soraghan, G. Di Caterina, and L. Petropoulakis, "Human emotion recognition in video using subtraction pre-processing," in *Proceedings of the 2019 11th International Conference on Machine Learning and Computing*, 2019, pp. 374–379.

[4] S. R. Livingstone and F. A. Russo, "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)," Apr. 2018, Funding Information Natural Sciences and Engineering Research Council of Canada: 2012-341583 Hear the world research chair in music and emotional speech from Phonak. [Online]. Available: https://doi.org/10.5281/zenodo.1188976