Face Emotion Recognition using Sequence Models

Authors :

TERBOUCHE Hacene
hacene.terbouche@gmail.com

CHAOUI Abdenour
abdennour.chaoui@gmail.com

Supervisors :

PETROVSKA Djiana
dijana.petrovska@telecom-sudparis.eu

EL-YAKOUBI Mounim
mounim.el_yacoubi@telecom-sudparis.eu

HMANI Mohamed
mohamed.hmani@telecom-sudparis.eu

2020-2021

Contents

List of Figures

Introduction

FER (Face Emotion Recognition) is an important research field which has presented and applied in several areas such as safety, health and in human machine interfaces. In the e-health field, for instance, usage cases include monitoring autistic children or people with autonomy loss, endowing avatars with emotion-like capabilities to interact naturally with elders in virtual coach applications.

Emotion recognition usually involves identifying emotion cues from face, verbal or non-verbal (body) expressions. There is a consensus that there are seven emotions classes: Joy, Anger, Anxiety, Pensiveness, Grief, Fear, Fright (but other categorizations exist as well).
Deep Learning for FER During the last decade, Deep Learning has revolutionized numerous tasks of classical machine learning specially in computer vision field. the state of the art has been dominated over the last years by the efficient Convolution Neural Networks (CNN). The goal of the project is to investigate the use of recurrent neural networks (RNN) for emotion recognition from face videos in order to model the temporal aspect of the emotions through the videos. In particular, the emphasis will be put on endowing the RNN by LSTM units that have proven to be very robust in numerous tasks where the temporal aspect is involved (speech recognition, handwriting recognition, gesture recognition, etc.).

Organization This report is organized as follows:

- Chapter I provides a background on the context of face emotion recognition. It gives also a review of the literature.
- Chapter II describes machine learning methodology used in this project such as dataset, explory data analysis, preprocessing and architectures.
- Chapter III presents the proposed models,explains the implementation of the project and all related technical details. Finally, it shows the results, makes some comments and draws the conclusions.
- The conclusion lays out a brief summary of what this report is about. In addition, it explores the challenges that we faced during implementation. It also explores contributions that could be made in future works.

Chapter 1

Background

1.1 Introduction

This chapter covers the important concepts related to Facial expression recognition **FER** systems. First we start by an overview of the problem these systems are trying to solve as well as some important terminology needed in the following parts. Then we introduce the most used datasets. Finally we explore the state of the art methods for dynamic FER tasks.

1.2 Problem formulation

Facial expression recognition **FER** is an important topic in computer vision and Artificial intelligence . It is also a key part in human-computer interaction. Emotion recognition based on facial expressions usually involves identifying Basic emotions **BEs** which are happiness, surprise, anger, sadness, fear,disgust, and neutral.
Several important concepts are crucial in the context of FER and plays an important role in research. First, The facial action coding system **FACS** that encodes the movements of specific facial muscles called action units (AU),which code the fundamental actions (46 AUs) of individual or groups of muscles typically seen when producing the facial expressions of a particular emotion. Next, Facial landmarks **FLs** which are visually salient points in facial regions such as the end of the nose, ends of the eye brows, and the mouth, Finally, Micro expressions **MEs** which indicate more spontaneous and subtle facial movements that occur involuntarily, they tend to reveal a person's genuine and underlying emotions within a short period of time [1].

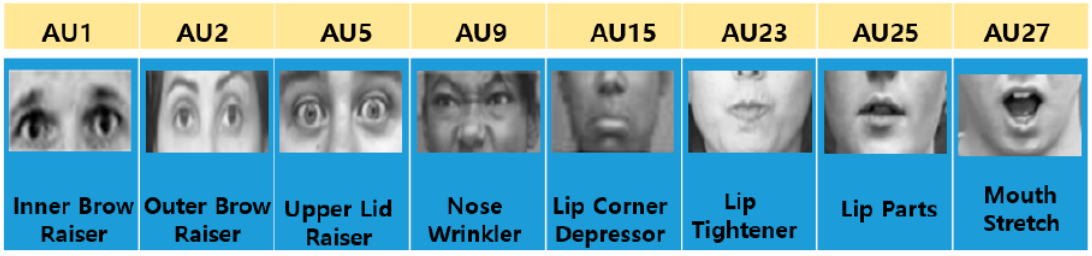

Figure 1: Various AUs (upper and lower face) [1]

Due to the impressive success achieved by deep learning techniques, the state of the art has been dominated over the last years by deep neural network approaches, these methods,

however encodes features representation with only spatial information from the current single image or frame and are called static based methods, on the other hand dynamic based methods considers the temporal relation among contiguous frames in the input facial expression sequence [2].
The goal of this project is to explore the effect of using dynamic based methods for emotion recognition from face videos.

1.3 FER Datasets

Deep learning based FER models requires Sufficient sufficient labeled training data that include as many variations of the populations and environments, so in this section we introduce the dynamic FER dataset used in the literature.

CK+ is a laboratory controlled database, it contains 593 video sequences from 123 subjects. The sequences vary in duration from 10 to 60 frames and show a shift from a neutral facial expression to the peak expression. Among these videos, 327 sequences from 118 subjects are labeled with seven basic expression labels (anger, contempt,disgust, fear, happiness, sadness, and surprise) [3]

MMI is a laboratory controlled database, it contains 32 . Among these videos, 213 sequences are labeled with six basic expression labels(without "contempt") sequences in MMI are onset-apex-offset labeled, i.e., the sequence begins with a neutral expression and reaches peak near the middle before returning to the neutral expression. [2]

AFEW 7.0 The Acted Facial Expressions in the Wild 7.0, contains video clips collected from different movies with spontaneous expressions, various head poses, occlusions and illuminations,it is divided into three data partitions in an independent manner in terms of subject and movie/TV source: Train (773 samples), Val (383 samples) and Test (653 samples). [2]

1.4 SOTA algorithms

In this section, we will discuss different dynamic-based methods used to consider the temporal relation in the input sequence. These methods are in general based on convolutional networks as an encoder for a single frame. Then, different fusion schemes exist.

1.4.1 Frame-aggregation

As a video is a time series of frames, we can treat each frame as an image. Then, we'll return back to the static-based model and find some method to aggregate results in order to improve performance. Various methods can be used to aggregate the static-base model output for all frames of the sequence. These methods can be categorized into two groups: decision-level frame aggregation and feature-level frame aggregation. Note that those methods ignore the temporal order of frames but take into account the whole sequence.

Decision level

The decision-level aggregation approach consists in using a CNN architecture for feature extraction and classification for every frame in the sequence which produces a probability

distribution over the classes. A simple and obvious way to aggregate these probabilities is toke the average, but other statistics such as weighted average can be used.

Feature level

This approach uses a CNN model to encode each frame f_i of the sequence into a fixed-length feature vector V_i (eg. $V_i \in R^{512}$), resulting in a set of vectors $V_1, V_2, ..., V_T$, where T is the length of the sequence. These vectors are then passed to a feature fusion module to produce a global feature vector. Many approaches have been proposed including statistical-based encoding such as mean, variance, minimum and maximum or even concatenating many of these statistics in one vector as done in [4]. It's also possible to concatenate all vectors in one big vector, but this will results in huge number of parameters in the classification model. Alternatively, matrix-based models such as eigenvector, covariance matrix and multi-dimensional Gaussian distribution can also be employed for aggregation. Next, a classification model such as SVM or MLP is used to output the probabilities of each emotion. figure 2 depicts the whole pipeline of this process.

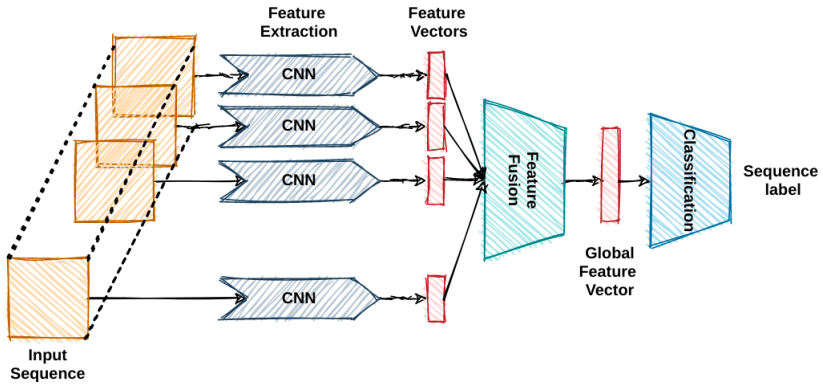

Figure 2: Feature level frame aggregation approach

1.4.2 3D-CNN based

Deep 3-dimensional convolutional networks (3D ConvNets) were first introduced in 2015 by Du Tran et al. [5]. It's more suitable for spatiotemporal feature learning compared to 2D ConvNets. 3D-CNN achieves the state-of-the art results in many domains such as activity recognition and sport recognition. The Sports-1M dataset contains 1.133.158 videos which have been annotated automatically with 478 Sports labels using the Youtube Topics API, which allows to apply fine-tuning on models trained on this dataset, the same way as we did with images using the ImageNet dataset.Figure 3 illustrates the difference between 2D convolutions and 3D convolutions, 2D convolution applied on an image will output an image, 2D convolution applied on multiple images also results in an image, whereas 3D convolution preserves the temporal information of the input signals resulting in an output volume.
In 2019, Sai Prasanna Teja Reddy et al. [6] proposed two 3D-CNN methods: MicroExp-STCNN (Micro-Expression SpatioTemporal Convolutional Neural Networks) which considers the full spatial information, and MicroExpFuseNet (Micro-Expression Fusion Network) which based on the 3D-CNN feature fusion of the eyes and mouth regions. For illustration, Fig 4 depicts the proposed MicroExpSTCNN architecture, which consists of 3D convolution

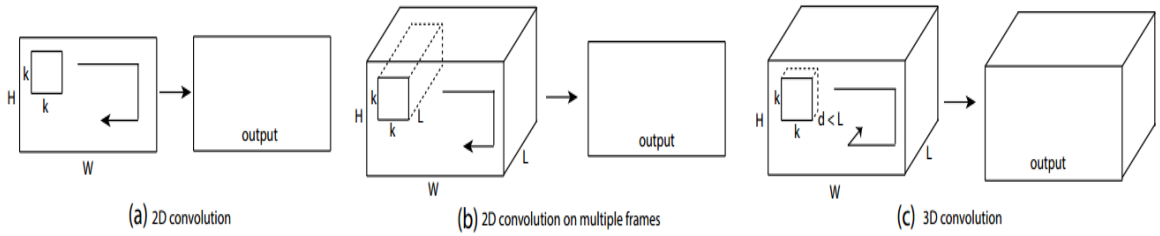

Figure 3: 2D and 3D convolution operations. a) Applying 2D convolution on an image results in an image. b) Applying 2D convolution on a video volume (multiple frames as multiple channels) also results in an image. c) Applying 3D convolution on a video volume results in another volume, preserving temporal information of the input signal. [5]

layer followed by 3D Max-pooling and some fully-connected layers and finally the softmax function to get the probability distribution over the 3 classes.

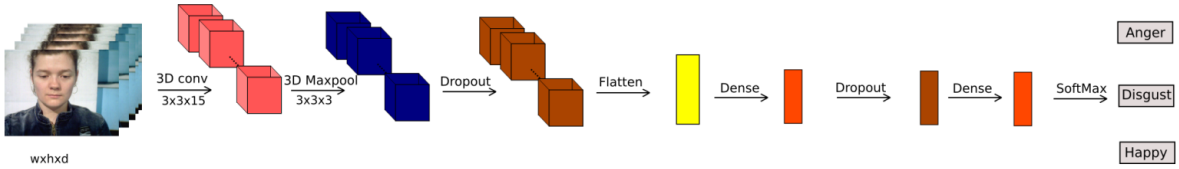

Figure 4: The proposed MicroExpSTCNN architecture for micro-expression recognition using 3D-CNN.

1.4.3 2D CNN-RNN based

CNN have been extensively used in FER models achieving remarkable results due to their representative power, However, these models focus on the spatial feature representation and can be referred to as image-based methods, they may not handle well the dynamic nature of facial expressions.
Several approaches were tried to take the temporal information into account. RNN has emerged as a preferable choice due to their ability to robustly derive information from sequences by exploiting the fact that feature vectors for successive data are connected semantically and are therefore interdependent, and yield to state-of-the-art performance on a variety of sequence analysis tasks.
Samira Ebrahimi Kahou et al. [7] proposed to model the spatio-temporal evolution of facial expressions of a person in a video using a Recurrent Neural Network (RNN) combined with a Convolutional Neural Network (CNN) in an underlying CNN-RNN architecture. First, A CNN is trained to classify static images containing emotions for the purpose of feature extraction. Then, The RNN (IRNN) is trained on the higher layer representation of the CNN. The final model is shown in figure 5

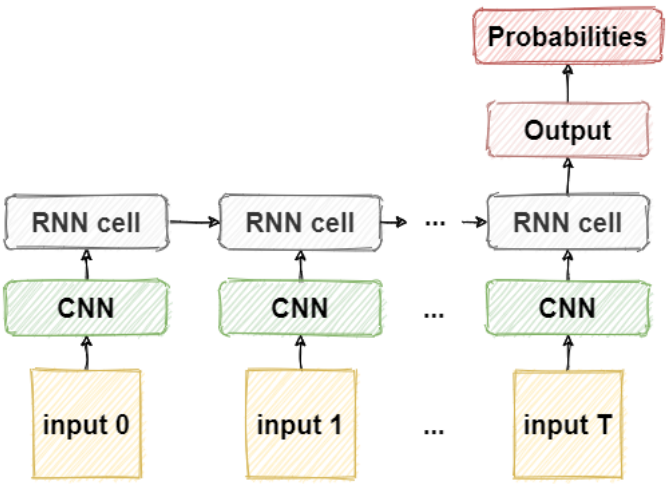

Figure 5: CNN-RNN

1.4.4 3D CNN-RNN based

To enhance the spatio-temporal modeling of facial emotion recognition, some approaches use a 3D CNN to extract spatial and short-term temporal features and recurrent networks such as LSTM or GRU to best model the relation between different frames in the video. Zhenbo Yu et al. [8] proposed a novel an end-to-end architecture called STC-NLSTM (Spatio-Temporal Convolutional features with Nested LSTM). The motivation of that paper is that hierarchical representation contained in early layers would be more effective than the features contained in the last layer only, which is the most used approach in literature, where only a feature vector of some length is used as an encoding of the image. According to that paper, this allows to learn multi-level appearance features of the frames. Figure 6 depicts the proposed architecture which can be divided into three major components. First a 3D-CNN module that consists of multiple convolution layers and responsible for extracting convolutional features taking as input a sequence of images. Second, multiple temporal LSTM (T-LSTM), one for each feature level of the CNN, which encode the appearance features as well as the temporal dynamics. Third, a convolutional LSTM (C-LSTM) that integrate the outputs of all T-LSTMs, where the time axis corresponds to the feature level in the CNN. Finally, a softmax classifier is used on the last hidden state off the C-LSTM to output one of the six basic emotion.

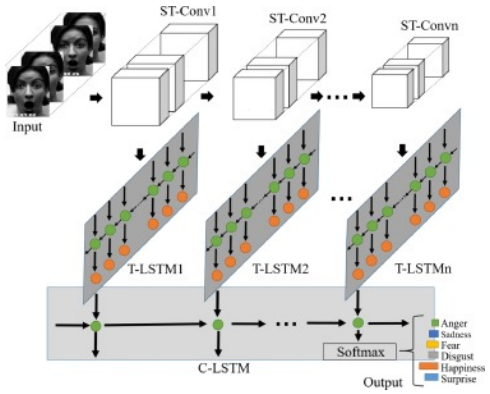

Figure 6: Architecture of the proposed STC-NLSTM [8]

1.4.5 Two-stream CNN based

In 2014, Karen Simonyan et al. proposed the first two-stream convolutional networks for action recognition in videos [9]. The idea is all about how to capture the spatial appearance in frames as well as the motion between frames. The proposed architecture involves two-stream CNN which incorporates the *Spatial stream ConvNet* that operates on individual video frames, and the *Temporal stream ConvNet* also called *Optical flow ConvNet*; which takes as input the stacked optical flow displacement fields between several consecutive frames. Thus, the second stream describes the motion between video frames.

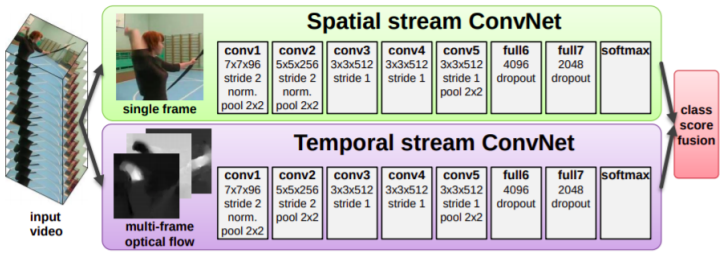

Figure 7: Two-stream architecture for video classification. [9]

Based on this idea, many works have been done in the facial emotion recognition task. Dandan Liang et al. [10] proposed a hybrid network that jointly learns spatial features and temporal dynamics for FER. Given the image sequence of an expression, spatial features are extracted from each frame using a deep network, while the temporal dynamics are modeled by a convolutional network, which takes a pair of consecutive frames as input. Finally, the framework accumulates clues from fused features by a BiLSTM network. Figure 8 illustrates the proposed architecture.

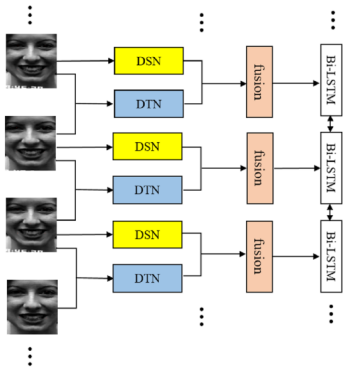

Figure 8: Architecture of the proposed framework. [10]

1.5 Conclusion

In this chapter we elaborated the basic concepts and terminology of FER systems, their types and the fundamental elements needed to conceptualize these systems, we then finally introduced the state of the art methods categorized by their building blocks.

Chapter 2

Methodology

Introduction

In this section, we will present our methodology for this projects. We will start by presenting the EMMPATHICS database and some statistics to understand its structure. Then, we will move to the preprocessing pipeline. After that, the Xception classfication model as well as the Xception Feature Extractor (XFE) are presented. Finally, we present the proposed architectures for sequence modeling.

2.1 EMMPATHIC database

EMMPATHIC database is a private database common between three research entities, in France, Norway and Spain. The dataset is constituted of facial videos taken during conversations, the videos are annotated by segments according to the emotion.

The videos are divided according to the country of their origin. Table 1 shows the number of videos as well as persons for each country

	France	Norway	Spain	Total
Number of videos	76	62	134	272
Number of individuals	38	31	67	136

Table 1: Table

Before presenting the data's statistics, let's introduce the terminology we are going to use in the next parts, first a **video** is the complete footage of the conversation, **frames** are individual pictures of a video ,**segments** are continuous parts of the video with the same annotation, whereas **sequences** are fixed length parts of one segment.

Figure 9 shows the 7 classes present in the dataset, discarding the None class and observing the absence of sufficient segments for surprise, angry and others, we only keep the three classes Neutral, Happy and pensive.

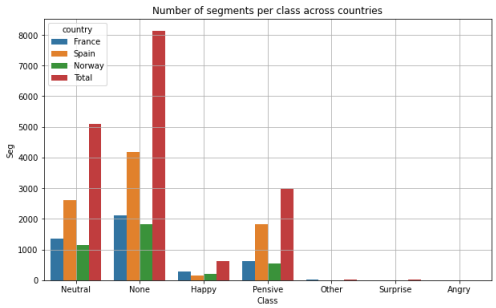

Figure 9: Number of segments per labels across countries

in order to understand the content and distribution of the data in regards to the remaining classes , we plot figure and to show the number of segments and Frames in the data

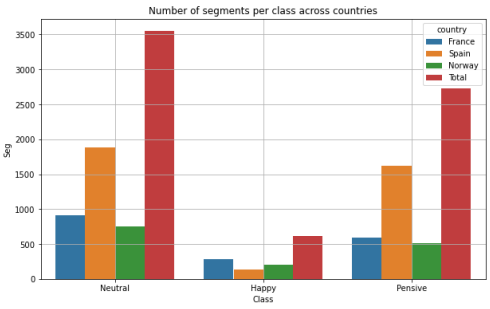

Figure 10: Number of segment per class across the kept 3 classes neutral' ,'happy', 'pensive'

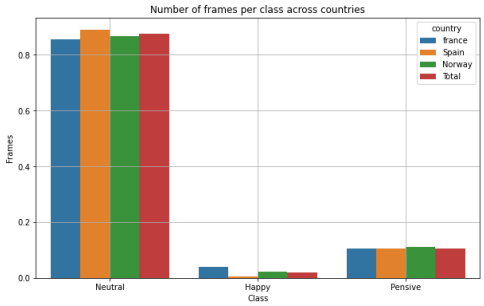

Figure 11: Number of segment per class across the kept 3 classes neutral' ,'happy', 'pensive' standardized by country

As shown in the previous figures, compared to the number of segments the imbalance in the number of frames is bigger, so in order to investigate this imbalance in segments length note that we represent the length of a segment by the number of frames this can be easily translated time unit knowing the sampling frequency ,figure is the boxplot of the lengths per class and figure show the histogram of length per class

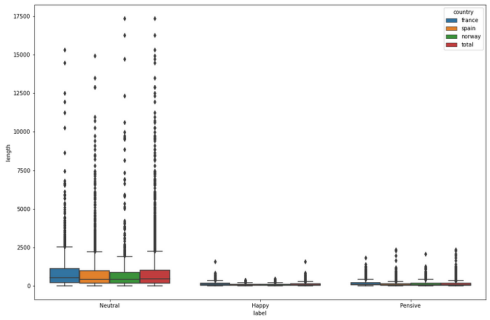

Figure 12: boxplot of the lengths per class across countries

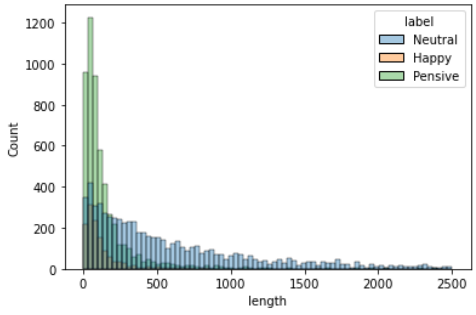

Figure 13: Histogram of lengths and discarding the extreme values bigger than 2500 frame shows that neutral is characterized by a longer segments compared to the other classes

Finally we notice segments with a small number of frames even one frame,these segments can be the result of than notation method so we consider them as noise that can affect the training, as the number of these segments are not significant, they are not taken into consideration.

	Number of segments	Number of Frames
Neutral	21	103
Happy	5	28
Pensive	31	187

Table 2: Total Number of discarded segments and frames with threshold of 8 per class

2.2 Preprocessing

The first common main step in facial recognition system is preprocessing, which is essential to extract meaningful representations because the irrelevant variations to facial expression such as different background, illuminations and head pose are common in datasets. The full prepossing process used is show on figure 14.

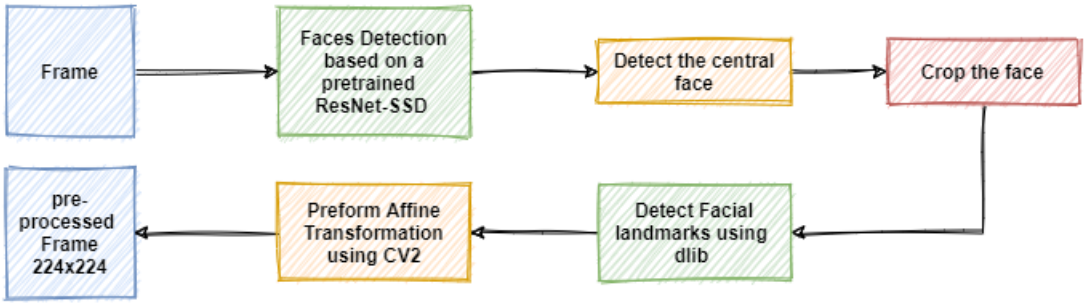

Figure 14: Frame preprocessing process

The preprocessing handles each frame individually, the first step is **Face Detection**, where each frame is passed by CV2 library through a pretrained ResNet-SSD model, all the faces detected above a certain confidence threshold are saved, in case of multiple faces in one frame, only the central face is considered. Then, the selected face is passed through CV2 dlib 68 face landmarks detector, these landmarks are then used to perform **Face alignment** using CV2 Affine Transform.

2.3 Feature Extraction

Convolutional neural networks have been extremely used in computer vision tasks and shown impressive results. The advantage of this type of networks is to extract visual features from raw images rather than handcrafted features which have less generalization

capacity and are often sensitive to irrelevant changes in the input. Many architectures have been proposed to solve the ImageNet classification problem and several improvements have been introduced. One of the most important concepts that enhanced the network's representation capacity is the so called *Inception* component used in a model called *GoogLeNet* [11], which won the ImageNet Large Scale Visual Recognition Challenge [12]. Figure 15 show a canonical Inception module.

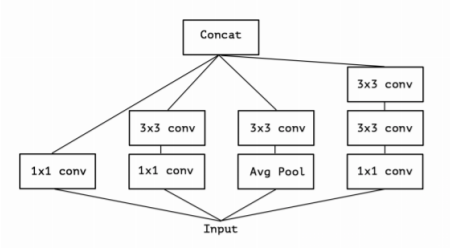

Figure 15: A canonical Inception module (Inception V3)[13]

In 2017, Francois Chollet proposed a novel architecture inspired by Inception termed *Xception*, which stands for *Extreme Inception* [13]. In this architecture, Inception modules have been replaced with depthwise separable convolutions with the same number of parameters and a gain an capacity. A depthwise separable convolution layer performs first *channel-wise* spatial convolution (a spatial convolution performed independently over each channel of an input), and then performs *pointwise* convolution (1x1 convolution, projecting the channels output by the depthwise convolution onto a new channel space). For more details about the Xception architecture, refer to that paper [13].
For the FER problem and as a first step, the static-based approach is used involving each frame independently. Thus, we use the Xception encoder which produces feature maps of size ($H \times W \times C = 7 \times 7 \times 2048$), pretrained on the ImageNet dataset. Then, a Global Average Pooling (GAP) layer is used to reduce the spatial dimensions and get a feature vector of size 2048, followed by three fully connected layers of dimension $1024, 512, 256$ respectively. Finally, a softmax layer is used to obtain the probability distribution over the emotion classes. Figure 16 describes the Xception model for frame classification. Note also that the 512-dim vector at the fully connected layer's output FC-512 is taken as an encoding of the frame to be used in the sequence modeling. Thus, the dashed box contains the so called Xception Feature Extractor.

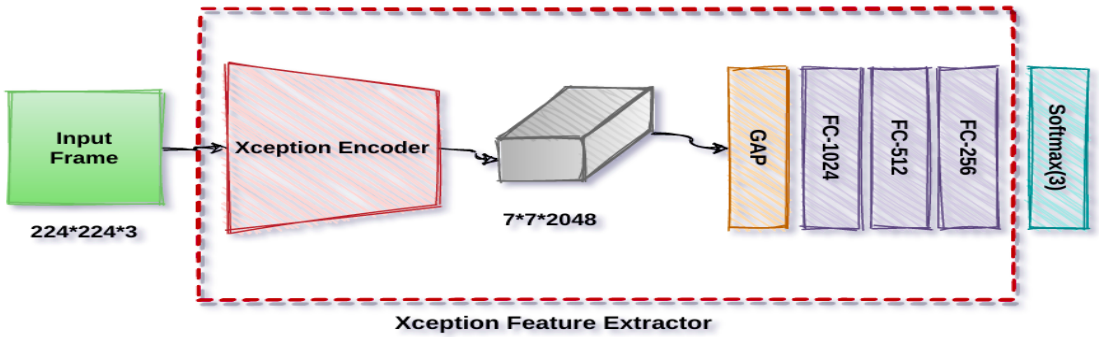

Figure 16: Xception architecture

2.4 Sequence modeling

2.4.1 LSTM cell

The standard RNN model presents some drawbacks. First, when the network is trained for long sequences, it suffers from the vanishing/exploding gradient problems, leading to convergence issues in the training procedure. Moreover, the RNN works best when each output $\hat{y}^{\langle t \rangle}$ can be estimated using *local* context. Local context refers to information that is close to the prediction's time step t. We could consider the RNN to have a short-term memory, which is not optimal when we need to predict events based on long sequence of time steps. To address these problems, we introduce more complex model called LSTM network [14]. The LSTM will be better able to remember a piece of information and keep it saved for many timesteps.
Figure 17 shows the operations of an LSTM cell, which contains four interacting blocks (or layers), each one have a well-defined purpose. These units are discussed bellow:

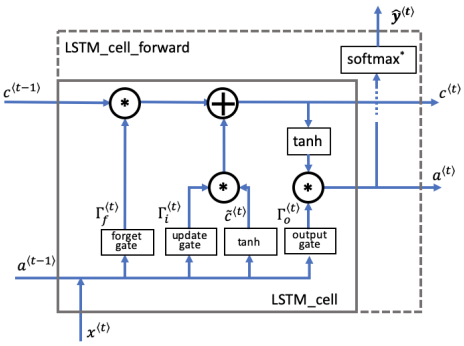

Figure 17: LSTM cell

- **Forget gate $\mathbf{\Gamma}_f$:**

$$\mathbf{\Gamma}_f^{\langle t \rangle} = \sigma(\mathbf{W}_f[\mathbf{a}^{\langle t-1 \rangle}, \mathbf{x}^{\langle t \rangle}] + \mathbf{b}_f) \tag{2.1}$$

- **Candidate value $\tilde{\mathbf{c}}^{\langle t \rangle}$:**

$$\tilde{\mathbf{c}}^{\langle t \rangle} = \tanh\left(\mathbf{W}_c[\mathbf{a}^{\langle t-1 \rangle}, \mathbf{x}^{\langle t \rangle}] + \mathbf{b}_c\right) \tag{2.2}$$

- **Update gate $\mathbf{\Gamma}_i$:**

$$\mathbf{\Gamma}_i^{\langle t \rangle} = \sigma(\mathbf{W}_i[a^{\langle t-1 \rangle}, \mathbf{x}^{\langle t \rangle}] + \mathbf{b}_i) \tag{2.3}$$

- **Cell state $\mathbf{c}^{\langle t \rangle}$:**

$$\mathbf{c}^{\langle t \rangle} = \mathbf{\Gamma}_f^{\langle t \rangle} * \mathbf{c}^{\langle t-1 \rangle} + \mathbf{\Gamma}_i^{\langle t \rangle} * \tilde{\mathbf{c}}^{\langle t \rangle} \tag{2.4}$$

- **Output gate $\mathbf{\Gamma}_o$:**

$$\mathbf{\Gamma}_o^{\langle t \rangle} = \sigma(\mathbf{W}_o[\mathbf{a}^{\langle t-1 \rangle}, \mathbf{x}^{\langle t \rangle}] + \mathbf{b}_o) \tag{2.5}$$

- **Hidden state $\mathbf{a}^{\langle t \rangle}$:**

$$\mathbf{a}^{\langle t \rangle} = \mathbf{\Gamma}_o^{\langle t \rangle} * \tanh(\mathbf{c}^{\langle t \rangle}) \tag{2.6}$$

- **Prediction $\mathbf{y}_{pred}^{\langle t \rangle}$:**

$$\mathbf{y}_{pred}^{\langle t \rangle} = \text{softmax}(\mathbf{W}_y \mathbf{a}^{\langle t \rangle} + \mathbf{b}_y) \tag{2.7}$$

2.4.2 LSTM network

As our goal is to demonstrate that temporal information is useful for the FER task, we will use LSTM networks. An LSTM neural network has the form of a chain of repeating modules of LSTM cells. Using the above equations, the LSTM model is able to preserve important information through time and understand which are the essential inputs to keep and what to discard. Many architectures have been tested for this task including LSTM-Net, DeepLSTM-Net and Bi-LSTM Net. The input of these models are a sequence of feature vectors generated by the XFE (Xception feature extractor) explained in the previous section, which is frozen during training. The reason why freezing all XFE's parameters is that raw images (frames) were not available for us, in addition to the limited time to train the model considering the computation resources that we had during the project. In other words, we used the encoding vectors directly as input to different sequence models to gain time and generate them only one time and save them on disk.
As a first architecture, we proposed the simple LSTM network composed of a chain of LSTM cells. Finally, a softmax layer is applied on the last hidden state outputting the probability distribution over the emotion classes as shown in Figure 18. During training, a *Dropout* layer is added before the softmax layer to reduce overfitting.

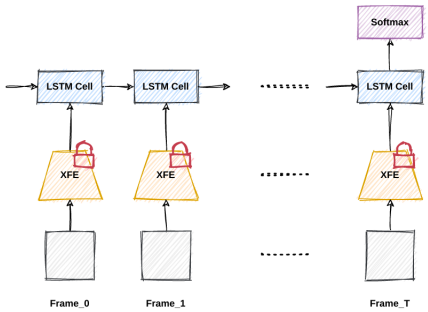

Figure 18: LSTM Network

We also proposed to use a deep LSTM network composed of two stacked LSTM layers. The softmax layer is applied on the last hidden state of the deepest layer. This would increase the representation capacity of the model. Figure 19 depicts the Deep LSTM architecture. Note that during training a *Dropout* layer is added between the two stacked LSTM layer as well as before the softmax layer.

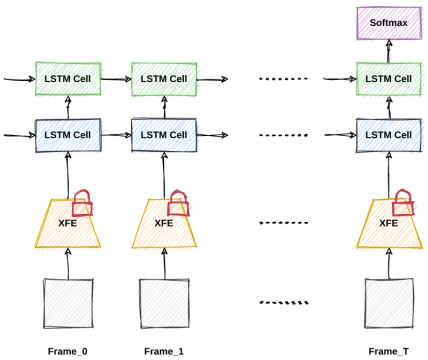

Figure 19: Deep LSTM Network

Figure 20 depicts the bidirectional LSTM that could work better as information flows from left to right as well as from right to left. As in the LSTM model, a *Dropout* layer is added before the softmax output during training.

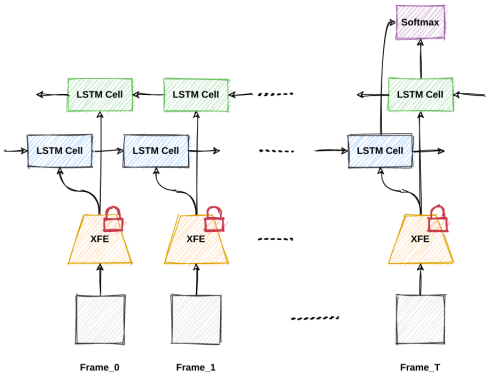

Figure 20: Bidirectional LSTM

Conclusion

In this chapter, we present the necessary elements in the machine learning pipeline before starting training models and evaluation them which will be done in the next chapter.

Chapter 3

Implementation and results

Introduction

This chapter presents how different models were implemented, trained and evaluated. We also present results and compare them. Basically, we have three approaches, the per-frame model (Xception model), the frame aggregation model and the CNN-LSTM model.

3.1 Data splits

Originally the static model (feature extractor) was trained 10 times and evaluated using 10-folds methods, but due to the time limit, we are going to follow another strategy to train our model.
we extract the features from only one model, then we split them into a train set, validation set and a test test representing 50%, 20% , 30% respectively .
After the split is done, we proceed into the division of segments in videos into sequences of 16 frames each per sequences 16 frames per sequence, and in order to address the imbalance problem we proceed we use a sliding window shown in figure 21 for the two classes happy and pensive in train and validation sets, with an overlap of 0.5 and 0.3 respectively.

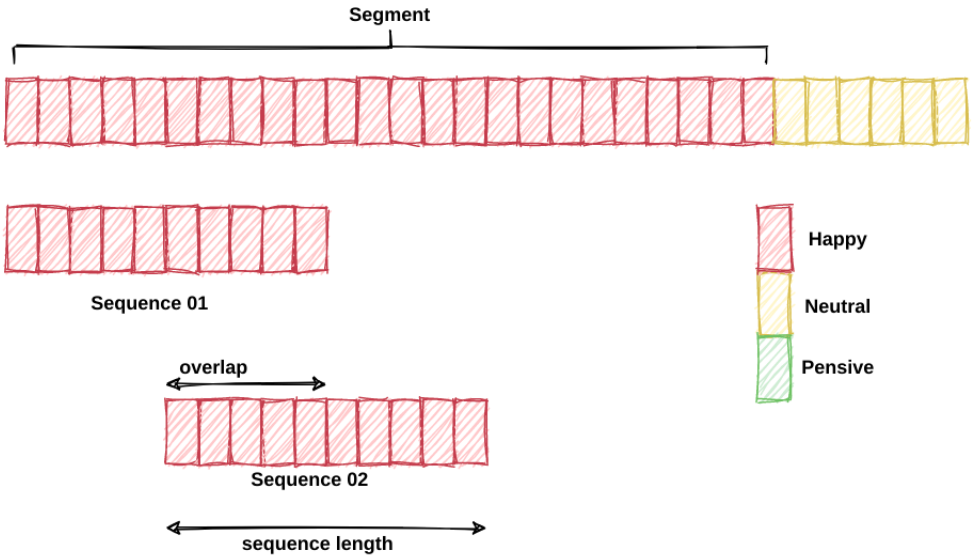

Figure 21: Sliding windows

3.2 Training

First, let's have a quick reminder of the context, the model we are going to train is going to have sequences of 16 image features extracted from a pretrained Xception as an input, these sequences are then classified as being one of the three classes : Neutral, Happy and Pensive. So basically we are going to train a many to one RNN model.

After the creation of segments, and the setting up of the dataloaders, and the model architecture, we proceed with the training phase of the model.

The main building blocks used during the training phase are weighted cross entropy loss and Adam optimizer.

For the training we fixed the sequence length to 16 frames per sequence, and used grid search tuning for the following parameters
- learning rate [0.1 , 0.03 , 0.01 , 0.003 , 0.001]
- Hidden layer size [64 , 128 , 256]
- Dropout [0.4 , 0.5 , 0.6 , 0.7 , 0.8]

the models are trained using the train dataset, the validation dataset was used for early stopping when the validation loss does not improve for 10 epochs, as well as hyperparameters tuning

Finally, comparing the results of the different models, we find the model with the parameters learning rate = 0.03 , hidden dimension = 128 and dropout = 0.4 has the best scores achieving a validation accuracy of 89%.

3.3 Model evaluation

Models were trained on train dataset, validation dataset was used for early stopping and hyperparameters tuning. Then best models are tested on test dataset.
The first model to test is the per-frame model which consists in the *Xception* model trained on independent frames. For more details see Figure 16. Due to the heavily imbalanced dataset, we will use the F1 score as a one number evaluation metric and the confusion matrix for full evaluation. F1-score, Recall and Precision are given as follows, where TP is True Positive, FN is False Negative and FP is False Positive.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.3}$$

The table 3 resumes the evaluation metrics based on the Xception model. We obtain a value of F1 measure of 48.23%.

Metric	Neutral	Happy	Pensive	Average
Precision	91.52	30.02	20.91	47.48
Recall	87.54	29.92	31.91	49.79
F1-score	89.48	29.97	25.26	48.23
Accuracy	-	-	-	81.49

Table 3: Xception evaluation

Figure 22 depicts the confusion matrix computed using the Xception model. We observe that the model is able to detect the neutral frames very well as it is the dominant class in the training set whereas happy and pensive frames are mostly confused with neutral class. The second model to test is the so called *Frame Aggregation Model*, a baseline model for

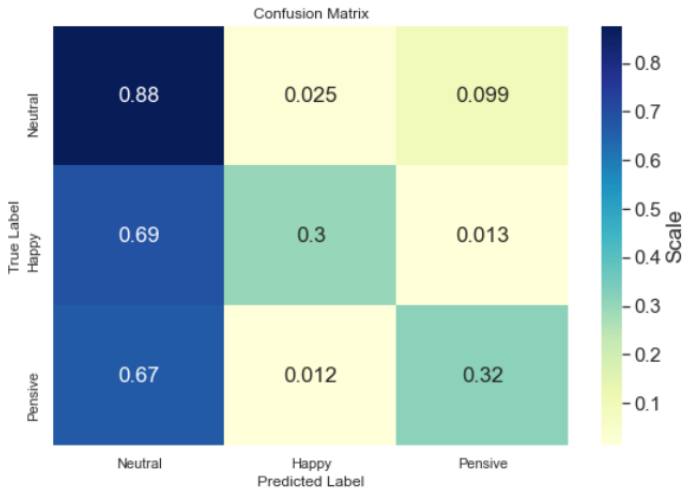

Figure 22: Xception model confusion matrix

sequence modeling which takes the average of softmax vectors generated by the Xception model applied to every frame in the sequence. Thus, this model do not need any additional training and uses directly the results of the Xception model.
Table 4 depicts the results of evaluation of the this model on test set using different metrics.We observe a little improvement of the F1 measure to a value of 50.41.

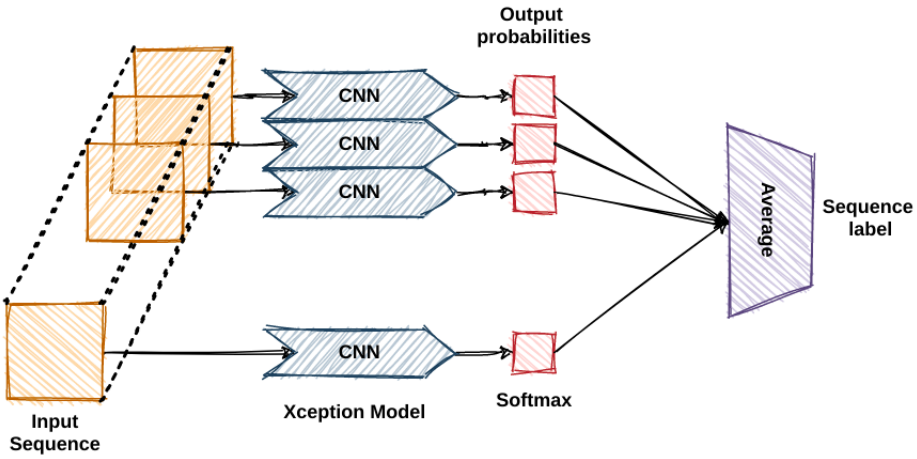

Figure 23: Frame Aggregation Model

Metric	Neutral	Happy	Pensive	Average
Precision	91.56	38.37	24.79	51.57
Recall	90.57	25.64	31.74	49.31
F1-score	91.06	30.73	27.72	50.41
Accuracy	-	-	-	84.04

Table 4: Frame Aggregation Model evaluation

Figure 24 depicts the confusion matrix for full evaluation. We don't see a significant difference with respect to the per-frame model. Rare classes (Happy and Pensive) are very confused with Neutral class.
Finally, we will evaluate the CNN-LSTM model. As we proposed three architectures, we used validation set for hyperparameters tuning, early stopping and model selection. For the best model, Table 5 depicts the different metrics used to evaluate the model. We observe a significant improvement of the F1 measure compared to the baseline model as we get a value of 67

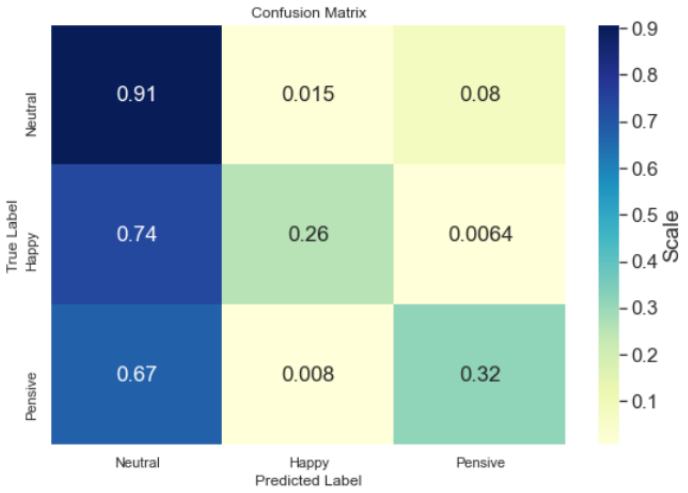

Figure 24: Frame Aggregation model confusion matrix

Metric	Neutral	Happy	Pensive	Average
Precision	97.44	30.55	40.13	56.04
Recall	83.73	73.87	82.53	80.04
F1-score	90.07	43.22	54.00	62.43
Accuracy	-	-	-	83.30

Table 5: CNN-RNN Model evaluation

For a full view of the performance of this model, we plot the confusion matrix as shown in Figure 25. Improvement over the baseline model are clearly remarked in the diagonal of the confusion matrix specially for the Happy and Pensive classes.

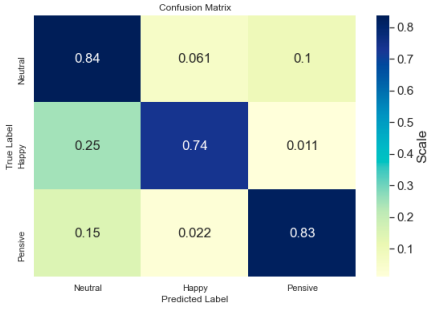

Figure 25: CNN-LSTM model confusion matrix

Conclusion

In this section, we presented the results for each approach. The imbalanced classification problem has a big impact on the results as the model tends to do very well on the dominant class (neutral class), but very bad on rare classes (happy and pensive). We demonstrated that sequence modeling using LSTM networks improved significantly the performance which encourage the reader for further fine-tuning of our models.

Conclusion

In this project, we tried to explore the use of sequences of Facial expressions to classify human basic emotions. As static methods already giving state of the art results, and knowing that emotion expression is a dynamic process, exploring the introduction of temporal information is worth looking at. First, we started by formulating the problem and introducing the basic concepts of FER systems, then we introduced the most common SOTA methods used in dynamic FER systems, followed by an analysis of the EMMPATHIC database used in the project as well as the steps in realising a complete dynamic FER system from prepossessing to feature extraction to sequence modeling, Finally, we explained the implementation as well as the results of our model compared to the Baseline model.

Bibliography

[1] B. C. Ko, "A brief review of facial emotion recognition based on visual information," *Sensors*, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/2/401/htm.

[2] S. Li and W. Deng, "Deep facial expression recognition: A survey," *IEEE Transactions on Affective Computing*, 2020. [Online]. Available: https://arxiv.org/pdf/1804.08348.pdf.

[3] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "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*, IEEE, Jun. 2010. DOI: 10.1109/cvprw.2010.5543262. [Online]. Available: https://doi.org/10.1109/cvprw.2010.5543262.

[4] S. A. Bargal, E. Barsoum, C. C. Ferrer, and C. Zhang, "Emotion recognition in the wild from videos using images," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, ser. ICMI '16, Tokyo, Japan: Association for Computing Machinery, 2016, 433–436, ISBN: 9781450345569. DOI: 10.1145/2993148.2997627. [Online]. Available: https://doi.org/10.1145/2993148.2997627.

[5] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "C3D: generic features for video analysis," *CoRR*, vol. abs/1412.0767, 2014. arXiv: 1412.0767. [Online]. Available: http://arxiv.org/abs/1412.0767.

[6] S. P. T. Reddy, S. T. Karri, S. R. Dubey, and S. Mukherjee, "Spontaneous facial micro-expression recognition using 3d spatiotemporal convolutional neural networks," *CoRR*, vol. abs/1904.01390, 2019. arXiv: 1904.01390. [Online]. Available: http://arxiv.org/abs/1904.01390.

[7] E. S. Kahou, V. Michalski, R. K. Konda, R. Memisevic, and J. C. Pal, "Recurrent neural networks for emotion recognition in video," *ICMI*, pp. 467–474, 2015.

[8] Z. Yu, G. Liu, Q. Liu, and J. Deng, "Spatio-temporal convolutional features with nested lstm for facial expression recognition," *Neurocomputing*, vol. 317, pp. 50–57, 2018, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2018.07.028. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218308634.

[9] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *CoRR*, vol. abs/1406.2199, 2014. arXiv: 1406.2199. [Online]. Available: http://arxiv.org/abs/1406.2199.

[10] D. Liang, H. Liang, Z. Yu, and Y. Zhang, "Deep convolutional BiLSTM fusion network for facial expression recognition," *The Visual Computer*, vol. 36, no. 3, pp. 499–508, Feb. 2019. DOI: 10.1007/s00371-019-01636-3. [Online]. Available: https://doi.org/10.1007/s00371-019-01636-3.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. arXiv: 1409.4842. [Online]. Available: http://arxiv.org/abs/1409.4842.

[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015. DOI: 10.1007/s11263-015-0816-y.

[13] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *CoRR*, vol. abs/1610.02357, 2016. arXiv: 1610.02357. [Online]. Available: http://arxiv.org/abs/1610.02357.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.