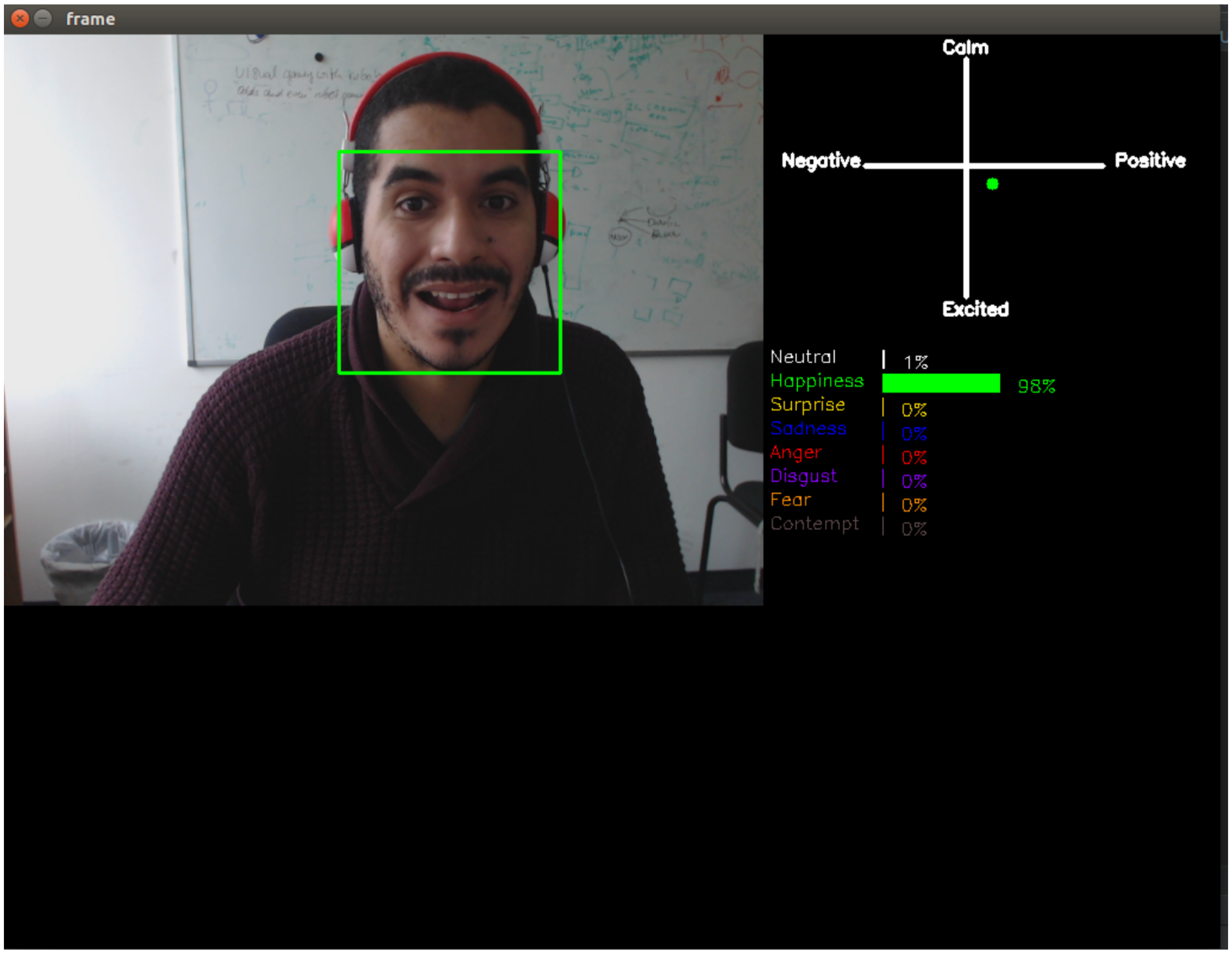
FaceChannel

Release 0.001

Pablo Barros

Oct 21, 2021

CONTENTS:



This project aims at providing a ready-to-use solution for facial expression recognition. The models available here are free to be used for personal and academic purpose.

CHAPTER ONE

QUICKSTART GUIDE

Here you will find instructions regarding how to install the library and run your first demo!

1.1 Instalation

To install the FaceChannel library, you will need python >= 3.6. The environment has a list of requirements that will be installed automatically if you run:

```
pip install facechannel
```

1.2 Facial Expression Recognition in Your Hand

FaceChannel is a python library that holds several facial expression recognition models. The main idea behind the FaceChannel is to facilitate the use of this technology by reducing the deployment effort. This is the current list of available models:

Table 1: Title

Model	Input Type	Output Type
FaceChannelV1 - Cat	(64x64x1)	[“Neutral”, “Happiness”, “Surprise”, “Sadness”, “Anger”, “Disgust”, “Fear”, “Contempt”]
FaceChannelV1 - Dim	(64x64x1)	[“Arousal”, “Valence”]
Self Affective Memory	(64x64x1)	[“Arousal”, “Valence”]

- FaceChannelV1 - Barros, P., Churamani, N., & Sciutti, A. (2020). The facechannel: A fast and furious deep neural network for facial expression recognition. SN Computer Science, 1(6), 1-10.
- Self Affective Memory - Barros, P., & Wermter, S. (2017, May). A self-organizing model for affective memory. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 31-38). IEEE.

1.3 Recognizing Facial Expression

To start the facial expression recognition is simple and painless:

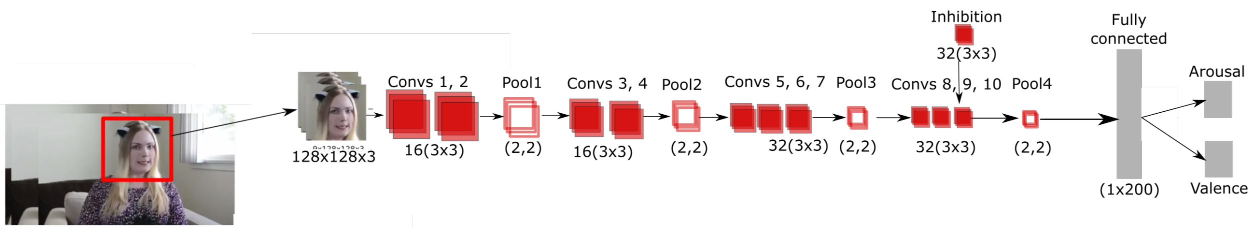
```python
"""Facial Expression Recognition"""
import cv2
from FaceChannel.FaceChannelV1.FaceChannelV1 import FaceChannelV1

faceChannelCat = FaceChannelV1("Cat", loadModel=True)

categoricalRecognition = faceChannelCat.predict(cv2.imread("image.png"))

print categoricalRecognition
```

For more examples on how to use, and to see our pre-made demos, check the examples folder.

CHAPTER

TWO

FACECHANNELV1



FaceChannelV1 is a smaller version of the FaceChannel model, with 800 thousand parameters. It was trained on the FER+ dataset, and for the dimensional version fine-tuned on the AffectNet dataset. It is available in two types: Cat, for categorical output with 8 different emotions, and Dim, for a dimensional output representing arousal and valence. FaceChannelV1 works on a frame-level, so for every input frame, it produces one output.

2.1 FaceChannel.FaceChannelV1 Model Definition

2.1.1 FaceChannel.FaceChannelV1.FaceChannelV1 module

FaceChannelV1.py

Version1 of the FaceChannel model.

`class FaceChannel.FaceChannelV1.FaceChannelV1.`**`FaceChannelV1`**(*type='Cat'*, *loadModel=True*, *numberClasses=7*)

Bases: `object`

`BATCH_SIZE = 32`
Batch size used by FaceChannelV1

`CAT_CLASS_COLOR = [(255, 255, 255), (0, 255, 0), (0, 222, 255), (255, 0, 0), (0, 0, 255), (255, 0, 144), (0, 144, 255), (75, 75, 96)]`
Color associated with each output of the pre-trained categorical model

`CAT_CLASS_ORDER = ['Neutral', 'Happiness', 'Surprise', 'Sadness', 'Anger', 'Disgust', 'Fear', 'Contempt']`
Order of the pre-trained categorical model's output

`DIM_CLASS_COLOR = [(0, 255, 0), (255, 0, 0)]`
Color associated with each output of the pre-trained dimensional model

`DIM_CLASS_ORDER = ['Arousal', 'Valence']`
Order of the pre-trained dimensional model's output

DOWNLOAD_FROM = 'https://github.com/pablovin/FaceChannel/raw/master/src/FaceChannel/FaceChannelV1/trainedNetworks.tar.xz'
URL where the model is stored

IMAGE_SIZE = (64, 64)
Image size used as input used by FaceChannelV1

buildFaceChannel()
This method returns a Keras model of the FaceChannelV1.rst feature extractor.

Returns a Keras model of the FaceChannelV1.rst feature extractor

Return type tensorflow model

getCategoricalModel(*numberClasses*)
This method returns a categorical FaceChannelV1.rst.

Returns a dimensional FaceChannelV1.rst

Return type tensorflow model

getDimensionalModel()
This method returns a dimensional FaceChannelV1.rst.

Returns a dimensional FaceChannelV1.rst

Return type tensorflow model

loadModel(*modelDirectory*)
This method returns a loaded FaceChannelV1.rst.

Parameters **modelDirectory** – The directory where the loaded model is.

Returns The loaded model as a tensorflow-keras model

Return type tensorflow model

predict(*images*, *preprocess=True*)
This method returns the prediction for one or more images.

Parameters

- **images** – The images as one or a list of ndarray.
- **preprocess** – If the image is already pre-processed or not. a pre-processed image has a format of (64,64,1).

Returns The prediction of the given image(s) as a ndarray

Return type ndarray

2.2 FaceChannel.FaceChannelV1 Image Processing Util Definition

2.2.1 FaceChannel.FaceChannelV1.imageProcessingUtil module

imageProcessingUtil.py

Image processing module used by the FaceChannelV1

class FaceChannel.FaceChannelV1.imageProcessingUtil.**imageProcessingUtil**
Bases: `object`

`currentFaceDetectionFrequency = -1`
A counter to identify which is the current frame for face detection

`detectFace`(*image*, *multiple=False*)
Detect a face using the cv2 face detector. It detects a face every "faceDetectionMaximumFrequency" frames.

Parameters

- `image` – ndarray with the image to be processed.
- `multiple` – allows the code to detect multiple faces in one single frame

Returns dets: the tuple of the position of the recognized face. using the format: startX, startY, endX, endY. A list if multiple faces are detected.

Return type ndarray

Returns face: the image of the detected face. A list if multiple feces are detected.

Return type ndarray

`faceDetectionMaximumFrequency = 10`
Search for a new face every x frames.

`property faceDetector`
get the cv2 face detector

`preProcess`(*image*, *imageSize=(64, 64)*)
Pre-process an image to make it ready to be used as input to the FaceChannelV1

Parameters

- `image` – ndarray with the image to be processed.
- `imageSize` – tuple with the final image size, default is (64x64)

Returns The pre-processed image

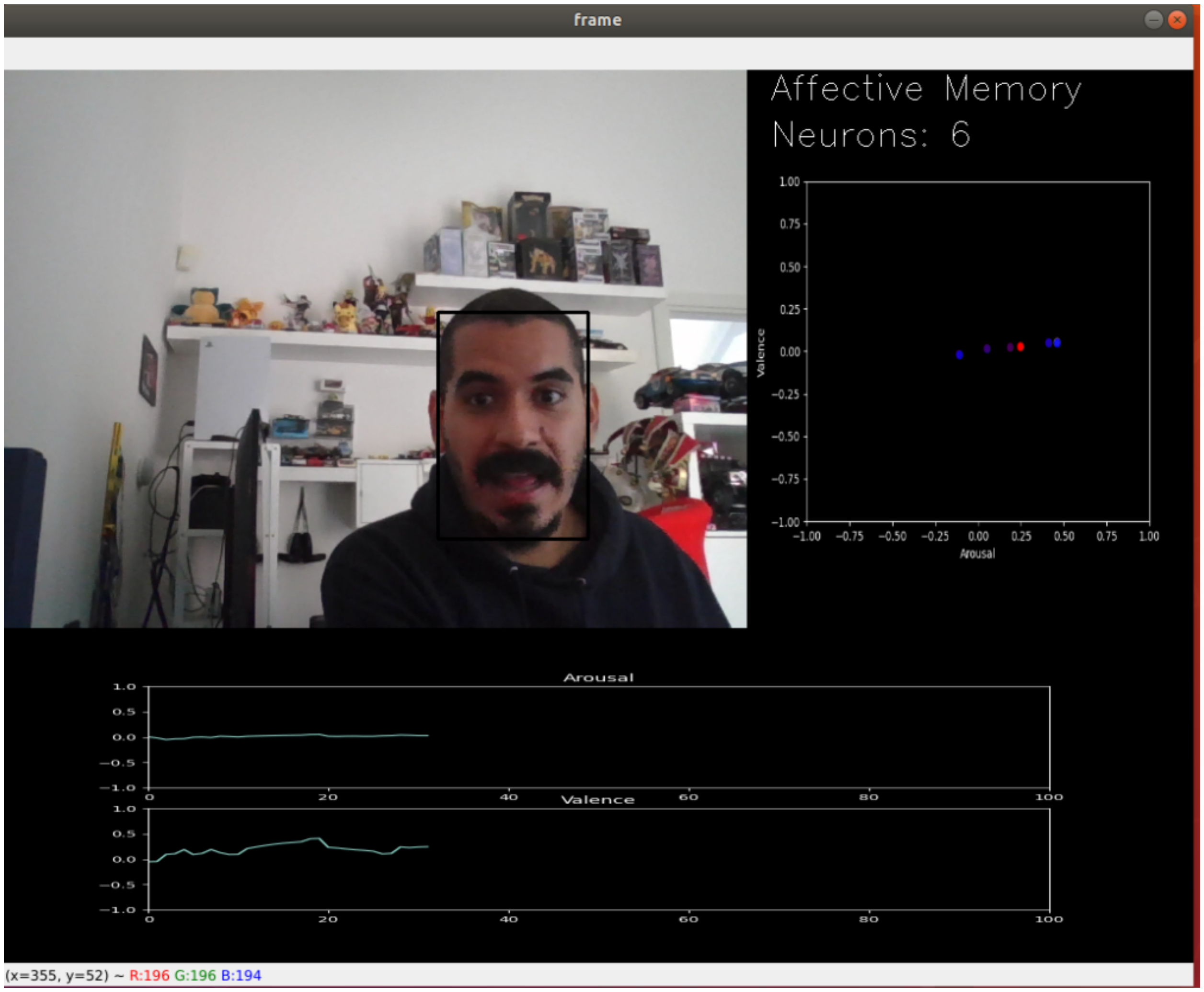
Return type ndarray

`previouslyDetectedface = None`
Identify if in the previous frame, a face was detected

CHAPTER

THREE

SELF-AFFECTIVE MEMORY



The Self-Affective Memory is a online learning model that uses the FaceChannelV1 predictions combined with a Growing-When-Required (GWR) network to produce a temporal classification of frames. It expects that each frame sent to it happens after the previously sent frame. It is able to predict arousal and valence, by reading the average of the current nodes of the GWR.

3.1 FaceChannel.SelfAffectiveMemory Model Definition

3.1.1 FaceChannel.SelfAffectiveMemory.SelfAffectiveMemory module

SelfAffectiveMemory.py

Self-Affective memory model.

class `FaceChannel.SelfAffectiveMemory.SelfAffectiveMemory.`**`SelfAffectiveMemory`**(*numberOfEpochs=5*, *insertionThreshold=0.9*, *learningRateBMU=0.35*, *learningRateNeighbors=0.76*)

Bases: `object`

`buildAffectiveMemory`(*dataTrain*)

Method that activey builds the current affective memory

Parameters **`dataTrain`** – initial training data as an ndarray.

`getNodes`()

Method that returns all the current nodes of the affective memory :return: a tuple of nodes and ages of each node. :rtype: ndarray tuple

`insertionThreshold = 0.9`

Activation threshold for node insertion

`learningRateBMU = 0.35`

Learning rate of the best-matching unit (BMU)

`learningRateNeighbors = 0.76`

Learning rate of the BMU's topological neighbors

`numberOfEpochs = 5`

Number of traning epoches for the GWR

`predict`(*images*, *preprocess=False*)

Method that predicts the current arousal and valence of a given image or set of images. as the affective memory is an online learning method, every given frame must be temporaly subsequent to the previous ones. It relies on the FaceChannelV1 for feature extraction.

Parameters

- **`images`** – The images as one or a list of ndarray.
- **`preprocess`** – If the image is already pre-processed or not. a pre-processed image has a format of (64,64,1).

Returns The prediction of the given image(s) as a ndarray

Return type ndarray

`train`(*dataPointsTrain*)

Method that trains the affective memory online :param dataTrain: initial training data as an ndarray.

CHAPTER FOUR

LICENSE

All the examples in this repository are distributed under a Non-Comercial license. If you use this environment, you have to agree with the following itens:

1. To cite our associated references in any of your publication that make any use of these examples.
2. To use the environment for research purpose only.
3. To not provide the environment to any second parties.

CHAPTER

FIVE

CONTACT

In case you have any issues, please contact:

pablo.alvesdebarros@iit.it

CHAPTER SIX

ACKNOWLEDGMENT

This environment and all its development is supported by a Starting Grant from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme. G.A. No 804388, wHiSPER.

PYTHON MODULE INDEX

INDEX