Convolutional neural network that will be able to recognize emotions in photos with human faces

•••

Dariusz Nostkiewicz
Rafał Gęgotek

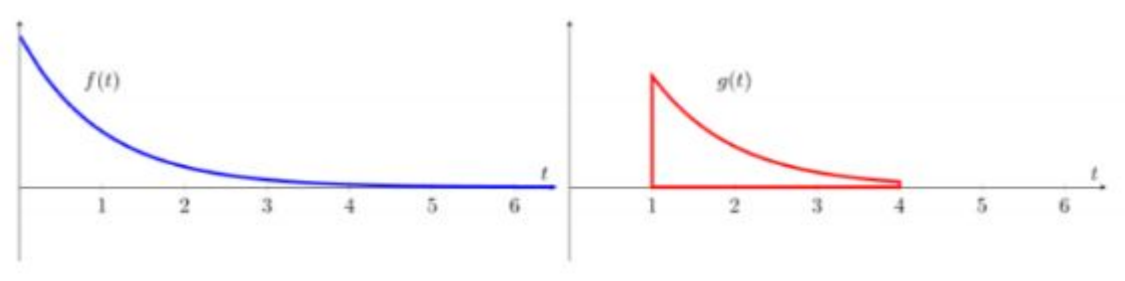
Convolution in mathematics

To understand the concept of convolutional neural networks, one must first explain what a convolution is.
It is a mathematical transformation defined for two functions (or the signals they describe) resulting in another which may be viewed as a modified version of the original functions. Usually it is a point-defined action that is a convolution product

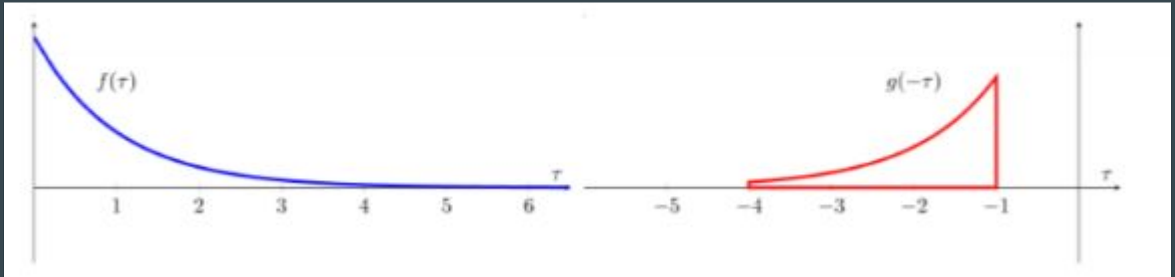
$$(f * g)(t) = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau$$

A visual representation of a convolution

- 1) Express each function in the context of an artificial variable τ.

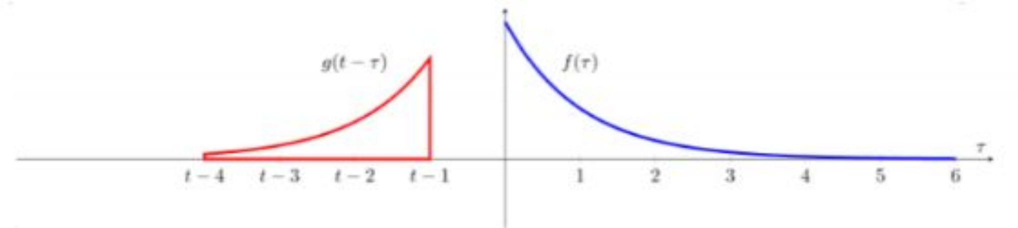


- 2) Make a reflection of one of the functions (here g (τ) -> g (-τ))

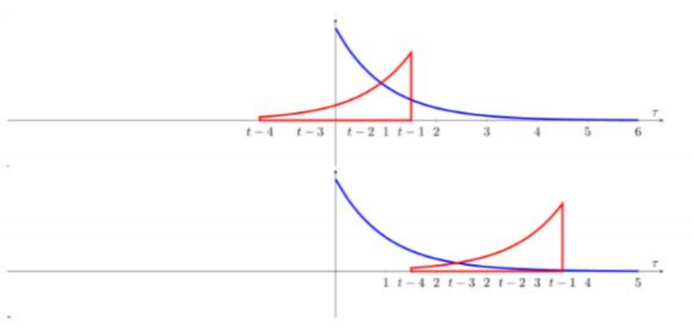


A visual representation of a convolution

- 3) Add the time shift t that allows the function g (t-τ) to move along the τ axis

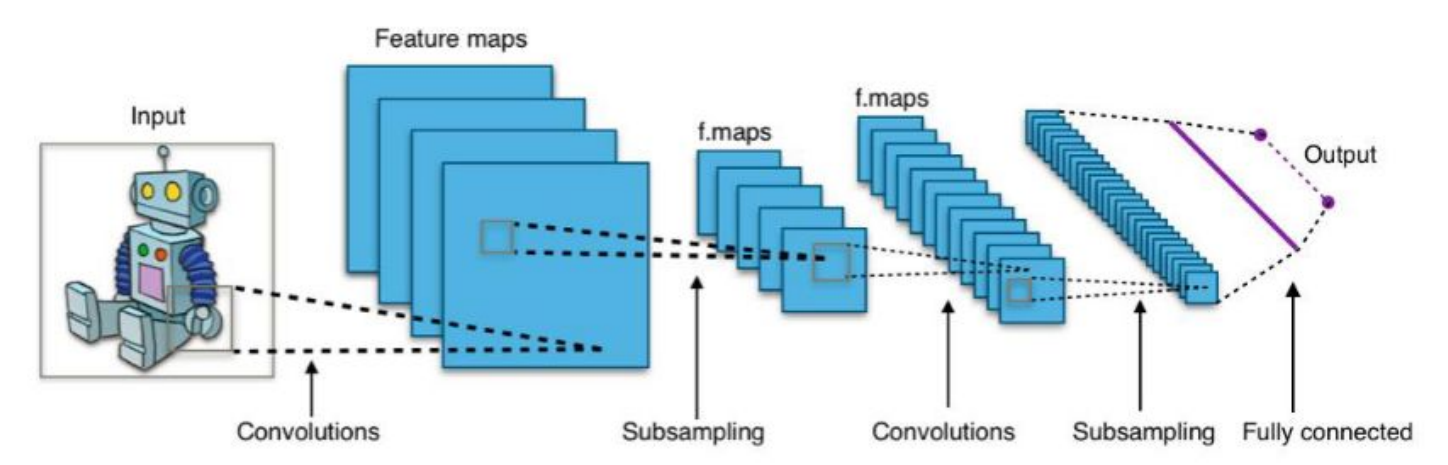


- 4) Start with t = -∞ and keep increasing t all the way to + ∞. At each point of intersection of both functions, find the integral of their product. The obtained points form a convolution of two input functions. (The convolution is not shown above).



Convolutional neural networks

A CNN network, like standard neural networks, consists of an input layer, a (many) hidden layer, and an output layer. The most important in this type of networks are the hidden layers, and they include convolutional, pooling and full layers. The graphic below illustrates this type of network



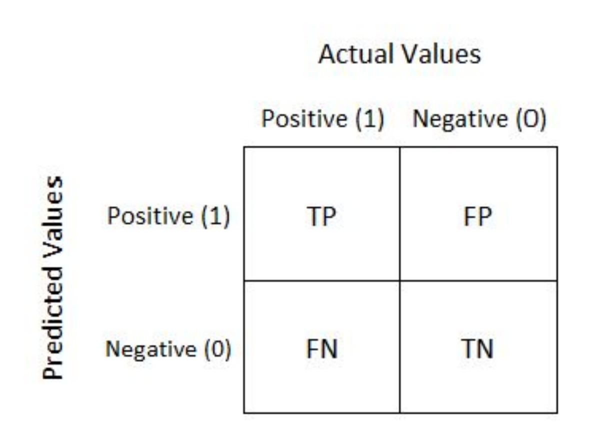
Types of neural networks

- **One-way networks** - these are networks in which there are no feedback loops, signals are sent from the input layer through hidden layers to the output layer.
- **Recursive networks** - networks with bi-directional connections between elements processors or with feedbacks, a backlink is allowed, i.e. the signal from the output can be transferred to the inputs of neurons from previous layers.
- **Mobile networks** - the interconnections between processing elements apply only to the closest neighborhood.
- **Convolutional networks** - analyze the input data and break them down into features. Features are searched for in close proximity data. The extracted features can be used to bring out the next more general ones data properties.

Validation methods

1) Confusion matrix

The error matrix is created from the intersection of the forecast class and the actually observed class, so we have 4 cases (2 for the agreement and 2 for the inconsistency of the forecast with the actual state).

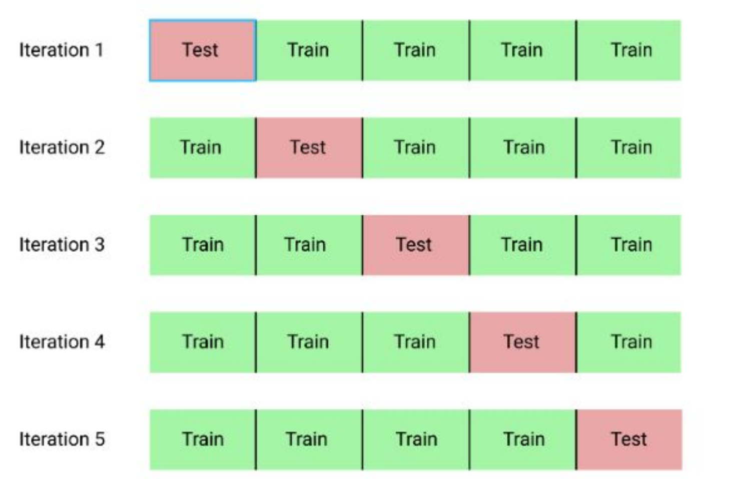


		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Validation methods

2) Cross-validation

It is a statistical method that consists in dividing a statistical sample into subsets, and then carrying out any analyzes on some of them (training set), while the others are used to confirm the reliability of its results (test set, validation set)



Iteration 1	Test	Train	Train	Train	Train
Iteration 2	Train	Test	Train	Train	Train
Iteration 3	Train	Train	Test	Train	Train
Iteration 4	Train	Train	Train	Test	Train
Iteration 5	Train	Train	Train	Train	Test

Validation methods

3) The error backpropagation algorithm works by "shifting" the error made by the network from the output layer to the input layer.

The error backpropagation training cycle consists of the following stages:

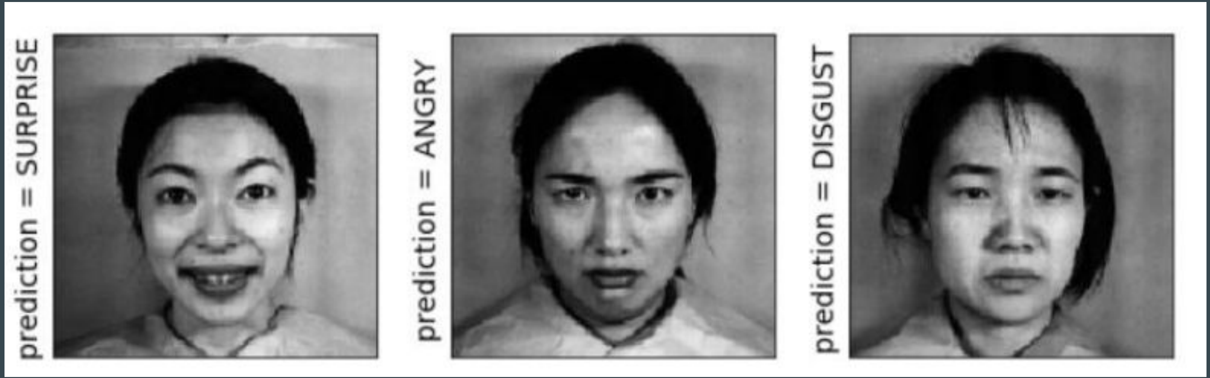
- Determination of the response of the output layer neurons and hidden layers to a given input signal.
- Determination of the error made by neurons located in the output layer and the message towards the input layer.
- Weight correction.

The Aim

Aim of the project

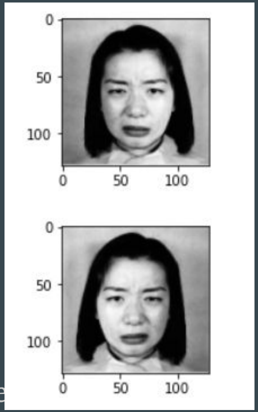
The aim of the project is to create a convolutional neural network that will be able to recognize emotions in photos with human faces. Face recognition is used in many systems, such as human-computer interactions and in security systems.

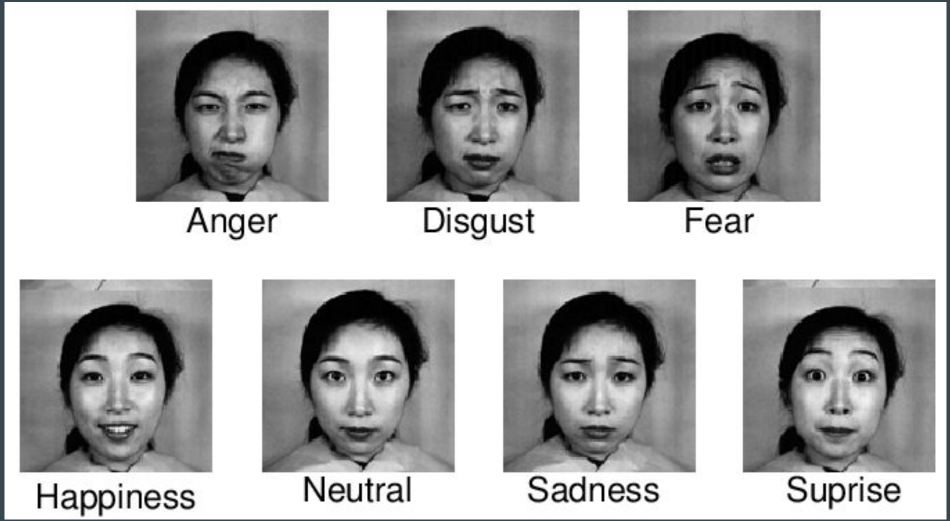
Non-verbal signals, such like facial expressions and showing emotions are important forms of interpersonal communication



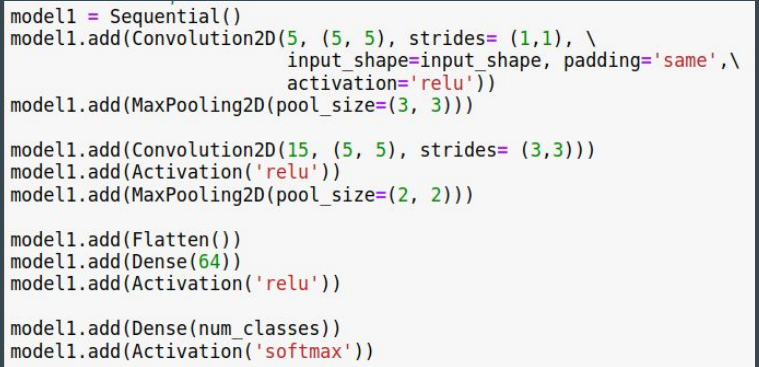
Database

- 10 Japanese female expressers
- 7 Posed Facial Expressions (6 basic facial expressions + 1 neutral)
- Several images of each expression for each expresse
- 213 images total
- Each image has averaged semantic ratings on 6 facial expressions by 60 Japanese viewers
- Resolution 256x256 pixels
- 8-bit grayscale
- Tiff format, no compression

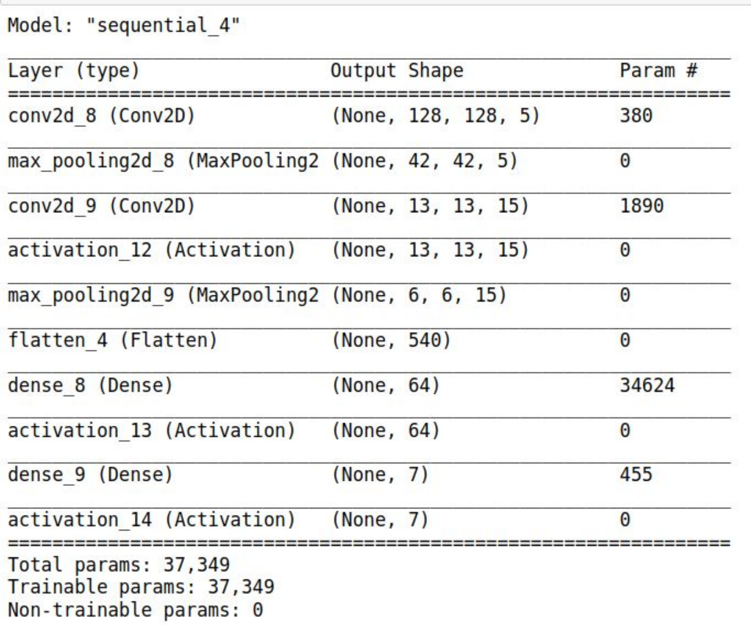




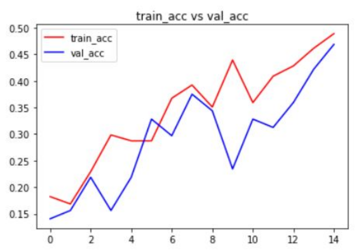
Basic Model

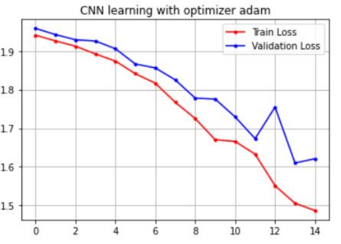


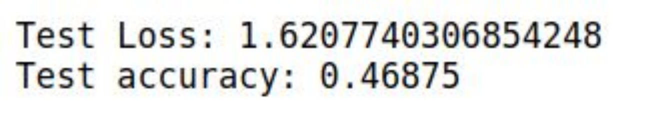
```
model1 = Sequential()
model1.add(Convolution2D(5, (5, 5), strides= (1,1), \
                         input_shape=input_shape, padding='same',\
                         activation='relu'))
model1.add(MaxPooling2D(pool_size=(3, 3)))

model1.add(Convolution2D(15, (5, 5), strides= (3,3)))
model1.add(Activation('relu'))
model1.add(MaxPooling2D(pool_size=(2, 2)))

model1.add(Flatten())
model1.add(Dense(64))
model1.add(Activation('relu'))

model1.add(Dense(num_classes))
model1.add(Activation('softmax'))
```



```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_8 (Conv2D)            (None, 128, 128, 5)       380
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 42, 42, 5)         0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 13, 13, 15)        1890
_________________________________________________________________
activation_12 (Activation)   (None, 13, 13, 15)        0
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 6, 6, 15)          0
_________________________________________________________________
flatten_4 (Flatten)          (None, 540)               0
_________________________________________________________________
dense_8 (Dense)              (None, 64)                34624
_________________________________________________________________
activation_13 (Activation)   (None, 64)                0
_________________________________________________________________
dense_9 (Dense)              (None, 7)                 455
_________________________________________________________________
activation_14 (Activation)   (None, 7)                 0
=================================================================
Total params: 37,349
Trainable params: 37,349
Non-trainable params: 0
```

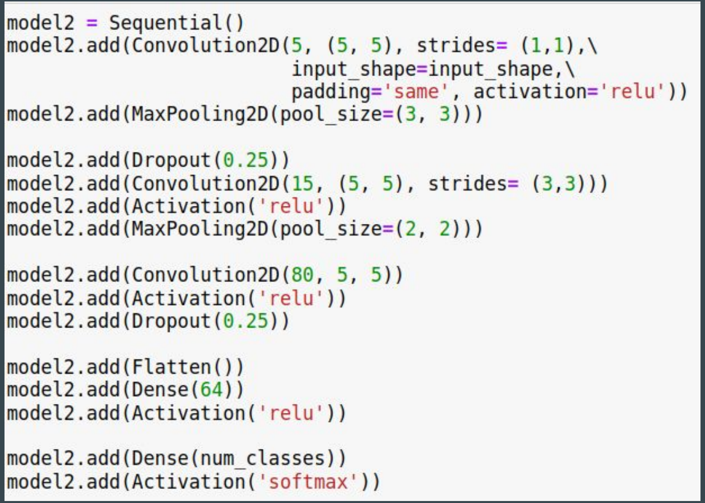




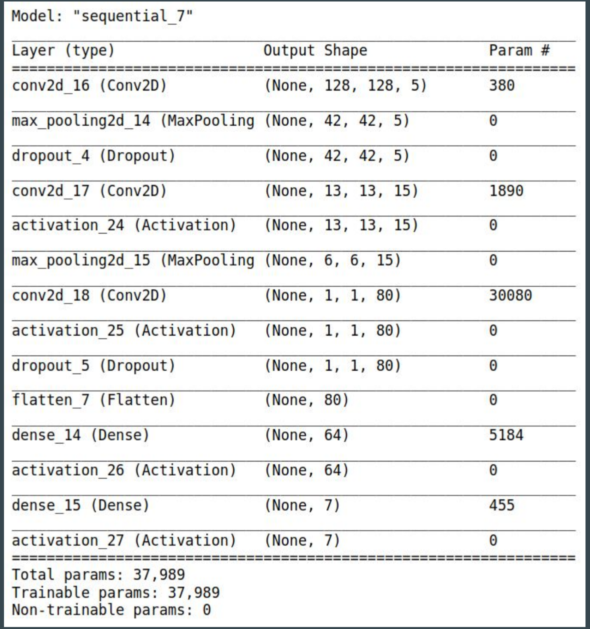


```
Test Loss: 1.6207740306854248
Test accuracy: 0.46875
```

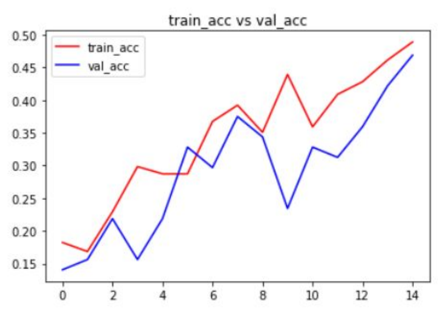
Model with dropouts

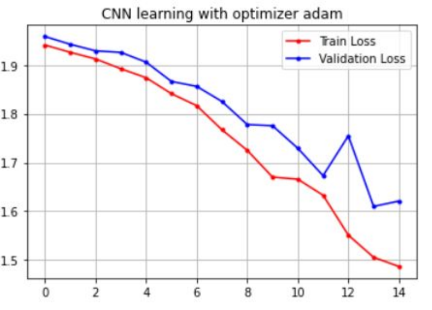


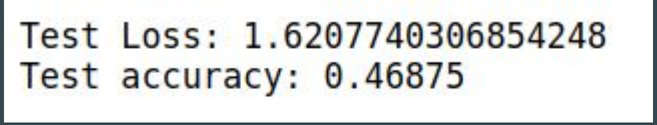
```
model2 = Sequential()
model2.add(Convolution2D(5, (5, 5), strides= (1,1),\
                         input_shape=input_shape,\
                         padding='same', activation='relu'))
model2.add(MaxPooling2D(pool_size=(3, 3)))

model2.add(Dropout(0.25))
model2.add(Convolution2D(15, (5, 5), strides= (3,3)))
model2.add(Activation('relu'))
model2.add(MaxPooling2D(pool_size=(2, 2)))

model2.add(Convolution2D(80, 5, 5))
model2.add(Activation('relu'))
model2.add(Dropout(0.25))

model2.add(Flatten())
model2.add(Dense(64))
model2.add(Activation('relu'))

model2.add(Dense(num_classes))
model2.add(Activation('softmax'))
```



```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_16 (Conv2D)           (None, 128, 128, 5)       380
_________________________________________________________________
max_pooling2d_14 (MaxPooling (None, 42, 42, 5)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 42, 42, 5)         0
_________________________________________________________________
conv2d_17 (Conv2D)           (None, 13, 13, 15)        1890
_________________________________________________________________
activation_24 (Activation)   (None, 13, 13, 15)        0
_________________________________________________________________
max_pooling2d_15 (MaxPooling (None, 6, 6, 15)          0
_________________________________________________________________
conv2d_18 (Conv2D)           (None, 1, 1, 80)          30080
_________________________________________________________________
activation_25 (Activation)   (None, 1, 1, 80)          0
_________________________________________________________________
dropout_5 (Dropout)          (None, 1, 1, 80)          0
_________________________________________________________________
flatten_7 (Flatten)          (None, 80)                0
_________________________________________________________________
dense_14 (Dense)             (None, 64)                5184
_________________________________________________________________
activation_26 (Activation)   (None, 64)                0
_________________________________________________________________
dense_15 (Dense)             (None, 7)                 455
_________________________________________________________________
activation_27 (Activation)   (None, 7)                 0
=================================================================
Total params: 37,989
Trainable params: 37,989
Non-trainable params: 0
```

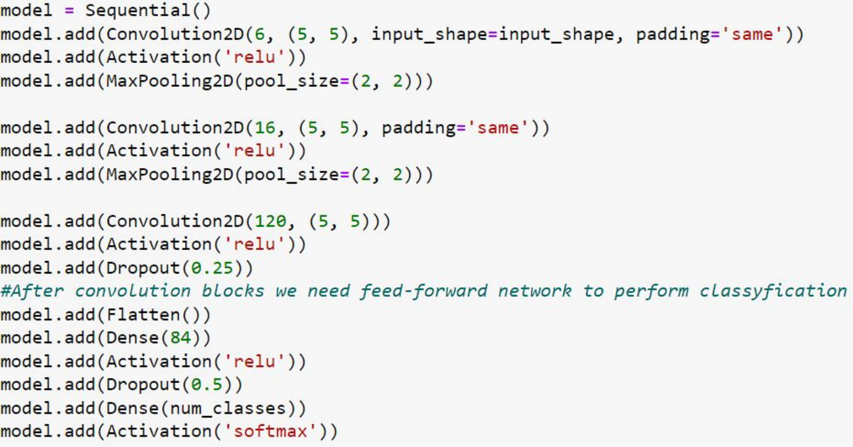




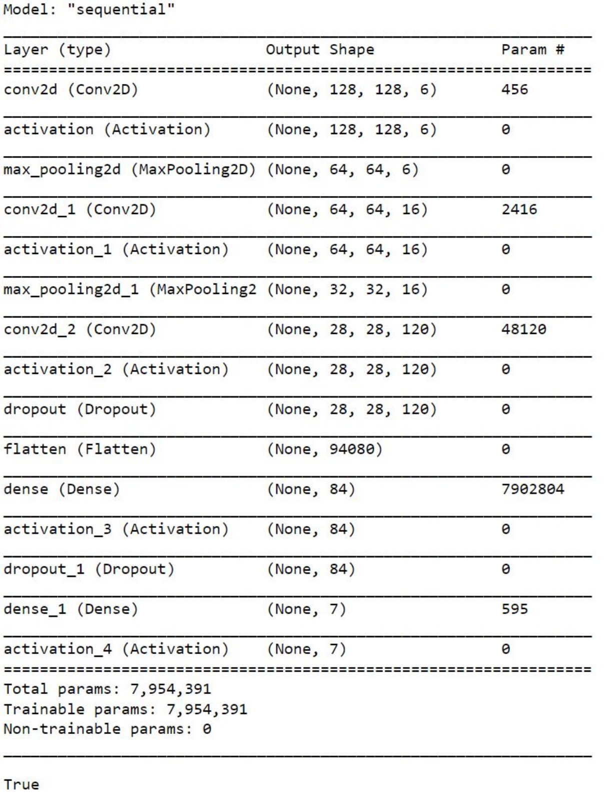


```
Test Loss: 1.6207740306854248
Test accuracy: 0.46875
```

Best model

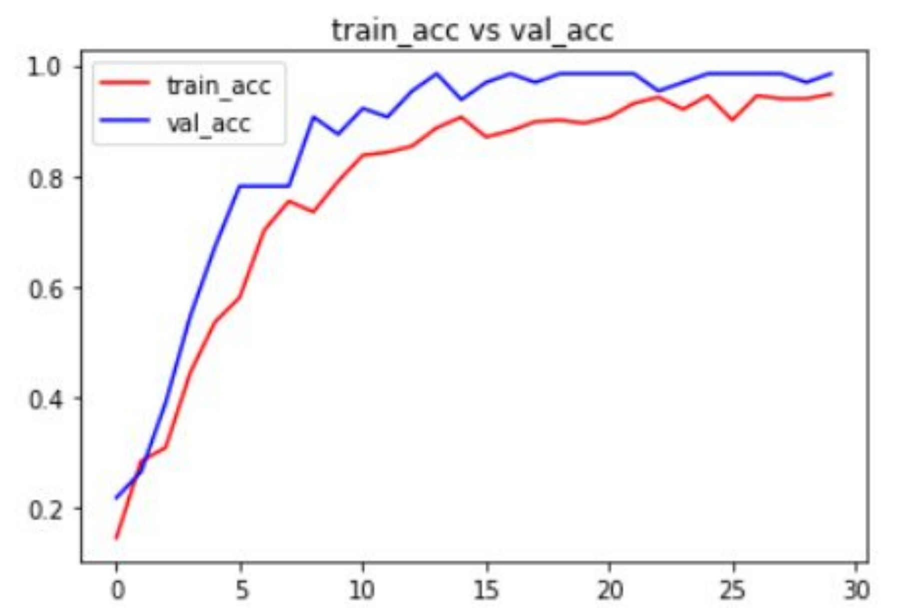


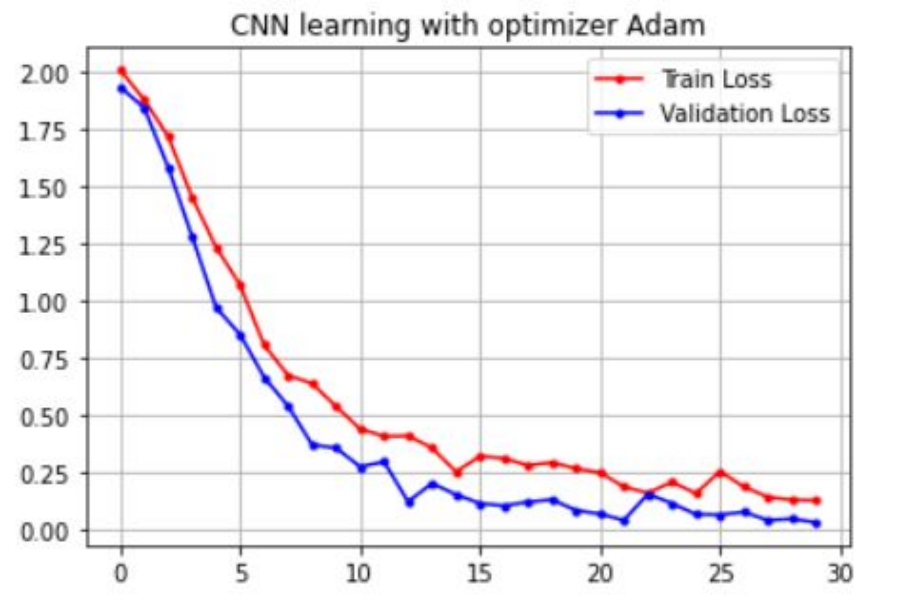
```
model = Sequential()
model.add(Convolution2D(6, (5, 5), input_shape=input_shape, padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Convolution2D(16, (5, 5), padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Convolution2D(120, (5, 5)))
model.add(Activation('relu'))
model.add(Dropout(0.25))
#After convolution blocks we need feed-forward network to perform classyfication
model.add(Flatten())
model.add(Dense(84))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes))
model.add(Activation('softmax'))
```

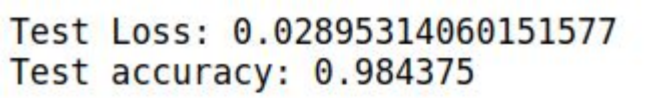


```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 128, 128, 6)       456
_________________________________________________________________
activation (Activation)      (None, 128, 128, 6)       0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 64, 64, 6)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 64, 64, 16)        2416
_________________________________________________________________
activation_1 (Activation)    (None, 64, 64, 16)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 32, 32, 16)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 28, 28, 120)       48120
_________________________________________________________________
activation_2 (Activation)    (None, 28, 28, 120)       0
_________________________________________________________________
dropout (Dropout)            (None, 28, 28, 120)       0
_________________________________________________________________
flatten (Flatten)            (None, 94080)             0
_________________________________________________________________
dense (Dense)                (None, 84)                7902804
_________________________________________________________________
activation_3 (Activation)    (None, 84)                0
_________________________________________________________________
dropout_1 (Dropout)          (None, 84)                0
_________________________________________________________________
dense_1 (Dense)              (None, 7)                 595
_________________________________________________________________
activation_4 (Activation)    (None, 7)                 0
=================================================================
Total params: 7,954,391
Trainable params: 7,954,391
Non-trainable params: 0
_________________________________________________________________

True
```

Best model (Adam)

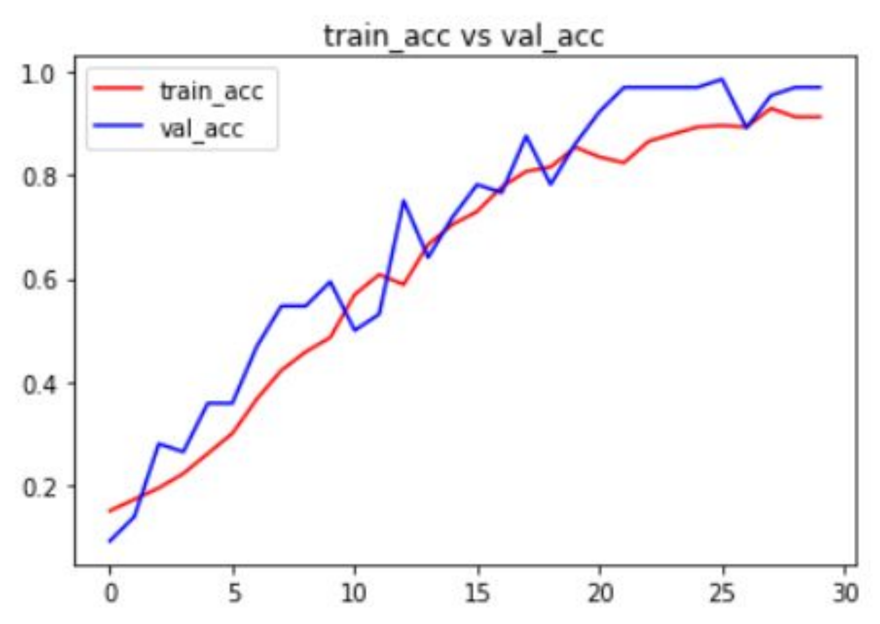


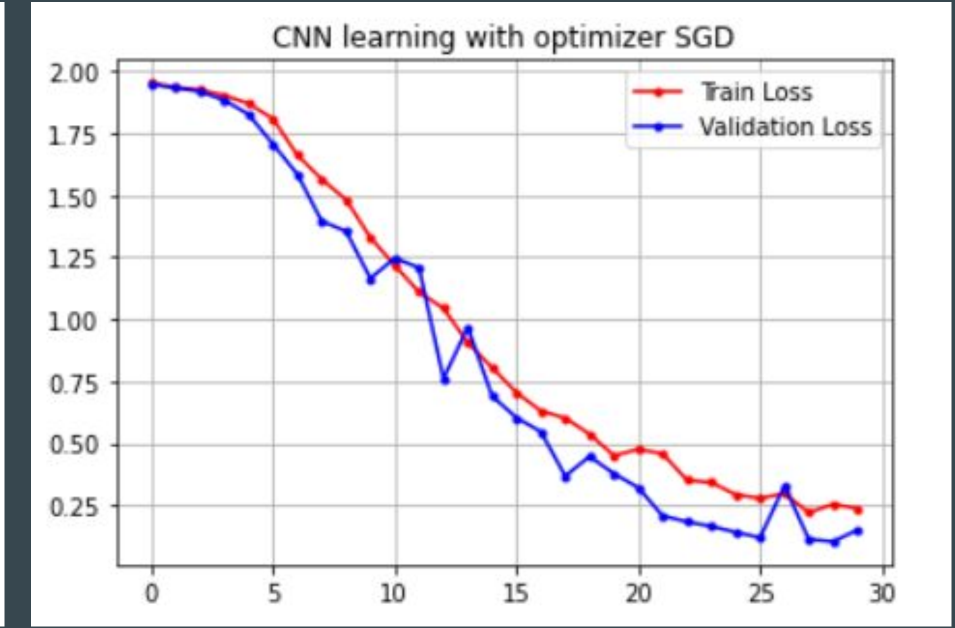


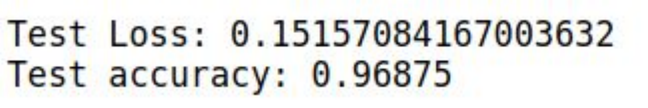


```
Test Loss: 0.02895314060151577
Test accuracy: 0.984375
```

Best model (SGD)

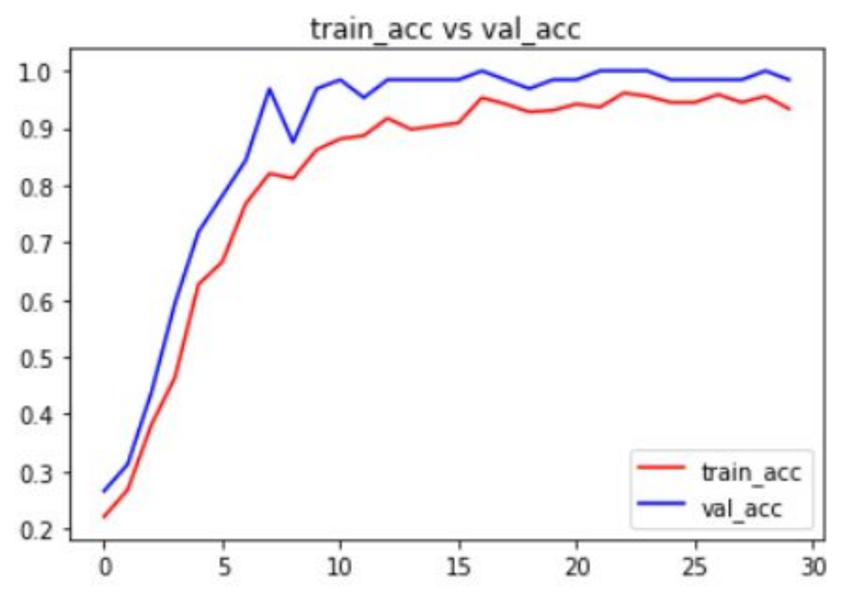


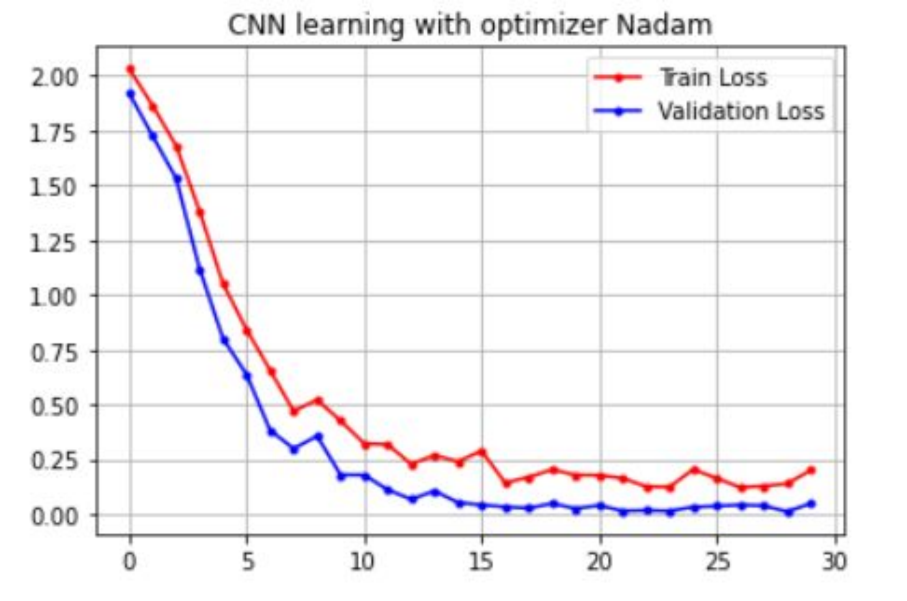


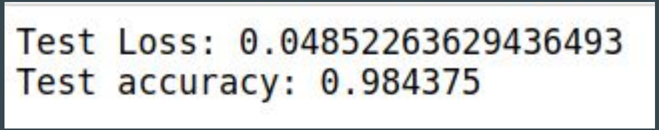


```
Test Loss: 0.15157084167003632
Test accuracy: 0.96875
```

Best model (Nadam)



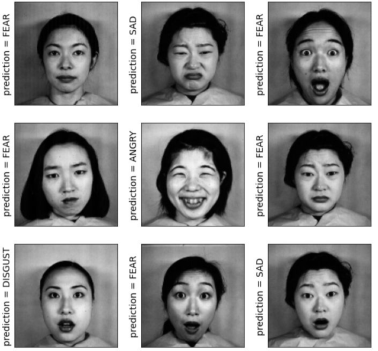


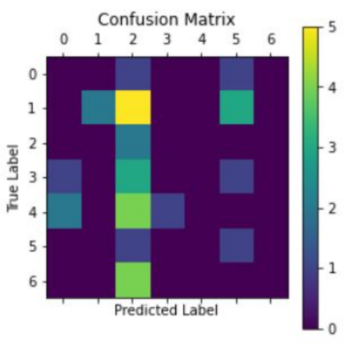


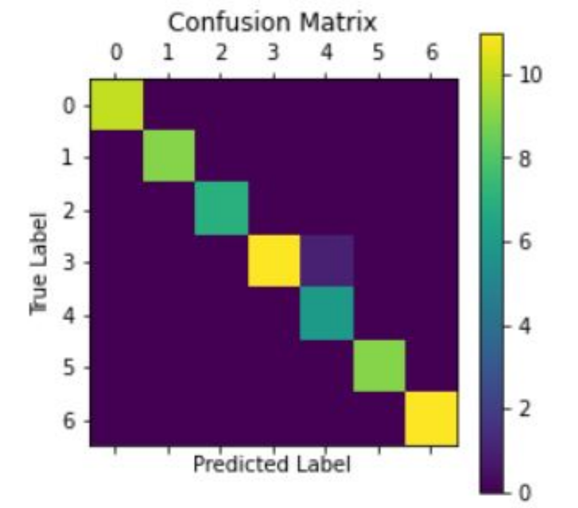
```
Test Loss: 0.04852263629436493
Test accuracy: 0.984375
```

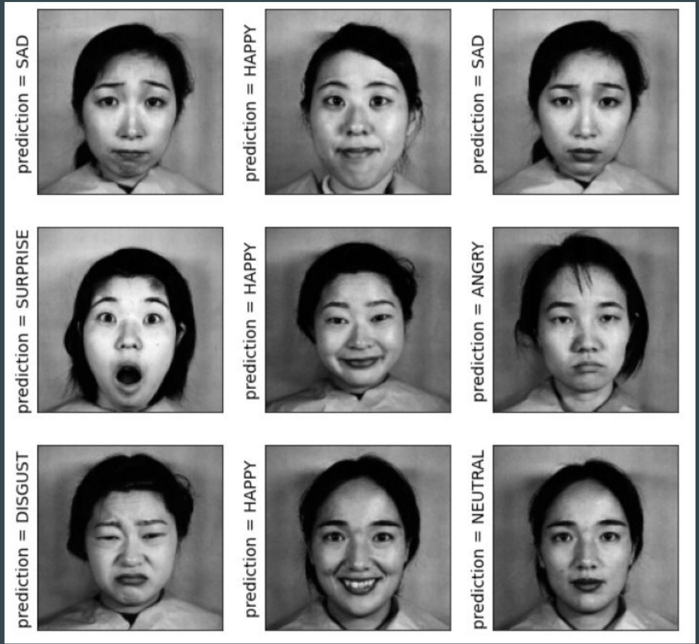
Summary

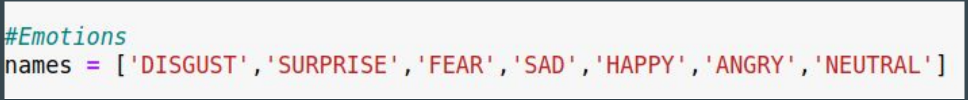
Recognizing emotions











```
#Emotions
names = ['DISGUST','SURPRISE','FEAR','SAD','HAPPY','ANGRY','NEUTRAL']
```