Recognising image emotions with the Valence and Arousal vector model using convolutional neural networks

Jaume Noguera

School of Computing
University of Portsmouth

Submitted in partial fulfilment of the requirements for the award of the degree of
Bsc (hons) Computing

April 2018

Declaration

Jaume Noguera
April 2018

Acknowledgements

I would like yo acknowledge my supervisor Dr Alex Gegov not only for all the support and motivation and support given and to succeed on the complementation of this thesis, but also to give me opportunities to conduct my work to the next level and rely on myself for other activities. I would also like to thank Dr Mihaela Cocea to give me insights and change the direction of the research.

To my mom for all the support given this years away from home and the guidance made with my academic life. No crec que hagues arribat tant lluny sense tu.

Abstract

Emotions play an important role in how we think and behave. Images are an effective tool to express and transmit emotions. In addition to the content, is been shown that colours, shapes, aesthetics and low level features play a considerable role on the emotion a picture transmits. With recent advances in computer vision, convolutional neural networks (CNN) showed a huge improvement against older methods and achieved the state-of-the-art performance in tasks such as image classification. CNN have the potential of analysing and extracting features and patterns from images. This recent years, researchers have investigated the topic and they built powerful tools to classify emotions from images into discrete categories. However, from the history of emotion classification more accurate accurate ways to classify emotions such as the valence and arousal model have been shown.

The following thesis focuses on the application of CNN to classify emotions from images, categorizing them into the valence and arousal dimensional model.

Table of contents

List of figures

List of tables

Nomenclature

Greek Symbols

δ Delta

$\sum$ Sigma

θ Theta

Other Symbols

Xception Extreme inception

Acronyms / Abbreviations

ANN Artificial neural network

CC Correlation coefficient

ReLU Rectified linear unit

CNN Convolutional neural network

conv Convolution

DIRTI DIsgust-RelaTed-Images

DNN Deep neural network

Emomardid Emotional Picture Data Base

GAPED Geneva affective picture database

IAPS International Affective Picture System

IED Image emotion dataset:

MSE Mean squared error

OASIS Open Affective Standardized Image Set

PCC Pearson correlation coefficient

RGB Red, green and blue

RMSE Root mean squared error

RNN Recurrent neural network

SAGR Sign agreement metric

SAGR Sign agreement metric

SGD Stochastic gradient descent

NAPS Nencki Affective Picture System

SMID Socio-Moral Image Database

SVM Support vector machines

Chapter 1

Project outline

This section gives an introductory overview of the solution that is going to be developed on this thesis, giving a preliminary statement of the problem, stating the research approach and a brief outline of the sections contained in the research.

1.1 Motivation

The research described in the following dissertation is motivated by the importance to understand and visualise human emotions to reduce the computing effective gap.

It is also motivated by the author's attitude to learn machine learning techniques and introduce himself to the topic.

1.2 Problem statement

In the affective computing domain emotion classification has been always a hot topic. With the advances of Artificial intelligence and Machine Learning this recent years the field of computer vision has evolved and matured with the invention of CNN demonstrated that machines can learn high level abstractions of data based on training images.

Emotions have always been a challenge in the field because they are not easily captured in words, the emotion an in images transmits depends on the viewer's perspective and can be biased depending on the individual's culture, location or beliefs. Even if some emotions are not perceived equally by everyone we are all human and by nature our life is guided by emotions. That's why there's certain situations where the knowledge refereed to an emotion of an image can be helpful.

Historically the problem has been approached using algorithms such as support vector machines (SVM) or naive bayes. This algorithms showed a considerable performance on emotion classification specifically SVM's. Years later, CNN showed an improvement respect to the older algorithms excelling in tasks such as image classification, segmentation and image extraction.

Approaches of emotion classification using CNN have shown the potential of the technology on the field achieving outstanding results. However, most of this approaches have been focusing on facial expressions to recognise emotions, not the image as a whole. Only few approaches are made on image emotion recognition and the results available are very limited because discrete emotion classification methods are used, meaning that only few classes of emotions are shown in the results.

On the literature of emotion classification we can see that classifying using discrete labels is not a very scientific approach. There's more accurate methods to categorize emotions such as the vector model also known as Valence and Arousal. Only one research paper has been found in the literature using this method to classify emotions on images but the method used to classify the emotions is not very efficient and the results shown are very restricted.

This research is going to approach image emotion classification using convolutional neural networks predicting a continuous output.

1.3 Research approach

Although approaches combining Recurrent Neural networks (RNN) and CNN have shown to improve the performance of the overall system in emotion recognition this project is going to focus on the application of CNN.

The main challenge of when designing CNN is the data. It is been shown a direct correlation between the data available and the performance of the system, specially when performing deep learning tasks. Research is going to be made on the design of the dataset used to train and evaluate the solution.

For the implementation of the CNN research is going to be carried out on existing methods comparing and evaluating them and choosing the solutions with best performance.

On the evaluation existing methods are going to be compared analysing the performance on the task and possible improvements that can be made to improve the overall error rate. Techniques such as data augmentation fine tuning or regularisation are going to be considered.

1.4 Outline

The project was divided in 3 stages. The first one was the research of relevant literature. Since the field is evolving very fast it was very problematic to keep up with the innovations of the field and the new literature. The in-depth description of the literature relevant to the project is described in chapter 2, giving a brief history on emotion classification. Chapter 3 introduces the readers to the topic of Neural networks and gives a broad insight on how they work, recent research, optimization techniques, evaluation metrics and architectures. On chapter 5 the author gives an outline of the requirements needed for the implementation of the artefacts, talking about the software used and the hardware demands. The second stage performed in this thesis is the design of the models and the dataset. They are described in chapter 6, where the reader can see how the dataset used to implement the models is built and gather insights about the models. The final stage of the research is done in chapter 7 and chapter 8 where the implementation and evaluation of the artefact is done. This was the longest process of all because it involved large amounts of time training, testing and tweaking.

Chapter 2

Literature review

This section introduces introduces the reader to the emotions and the research done in the field, giving a brief outline to the history of emotion classification, analysing the existing techniques and novel methodologies. Also critically evaluates existing solutions and available datasets with their distinctions.

2.1 Emotions

Emotions are a powerful strength in human behaviour. Strong emotions or feelings can force humans perform actions they normally not do or avoid situations they normally enjoy (Tangney and Fischer, 1995).

2.1.1 What is Emotion?

In the literature there is multiple definitions on what is an emotion but there's no consensus about what is the correct definition. Often authors define emotion as a list of feelings (anger, sadness, surprise, fear...). In the field of psychology is defined as a complex state of feeling that results in physical and psychological changes that influence thought and behaviour. Emotionality is associated with a range of psychological phenomena, including temperament, personality, mood, and motivation (Schacter et al., 2011). However, for the purposes of this thesis we will define emotion as::

> Any mental experience associated with high intensity and high hedonic content (pleasure/displeasure) (Cabanac, 2002).

2.1.2 Emotion clasification

In the early 1970s, G. B. A. Duchenne, a french physician and physiologist conducted an study on manipulating facial expression through applying galvanic electrical stimulation (Snyder et al., 2010). He wanted to demonstrate that there's different muscles responsible for each individual emotion in our expression. Charles Darwin, the well known naturalist prompted his doubts in the Duchenne model and conducted a study on the material. Darwin performed what we know as the first study in the human facial expression of emotions, his message was that emotion expressions are evolved from the past and his role was to protect us from danger, prepared us for action. He also showed that emotions have a communicative function of welfare and mankind. In his book The Expression of the emotions in man and animals(Darwin and Prodger, 1998) elaborated three principles that explain why emotions are expressed the way they are.

Discrete categories

The emotion classification techniques have been evolved since Darwin's theories. Dr Paul Ekman is a psychologist pioneer in the study of emotions and facial expressions. In 1976 through a series of studies across many cultures, ran an experiment consisting on describing a situation and showing to individuals a photography with few facial expressions. The individuals had to describe the emotion perceived and given a set of photographs, choose the facial expression of best fit. Ekman's findings showed that some perception of emotions can vary between culture or regions specially with cultures that never had contact with the outside world such as the fore tribe in New Guinea. Besides that, he showed a relationship between a particular set of emotions perceived by all human beings and gave the light to the basic 6 universal emotions (Ekman, 1992).

Anger, disgust, fear, happiness, sadness, and surprise

After Ekman's discovery, in 1980 Robert Plutchik after an intense research on the field, based on his own earlier research (Plutchik, 1980) he created a new concept known as the 'wheel of emotions'

This new concept was called wheel of emotions because it demonstrated how different emotions can blend and create new emotions. To understand the wheel we can identify that there's multiple colours, the colours represent similar emotions. The primary emotions are displayed on the first section after the central circle. If the colour intensifies the emotion strengths. The emotions that have no colour are a mix of the primary emotions surrounding them. The wheel is also designed to find the contrary emotion at the opposite side of the wheel. (Plutchik, 1991)

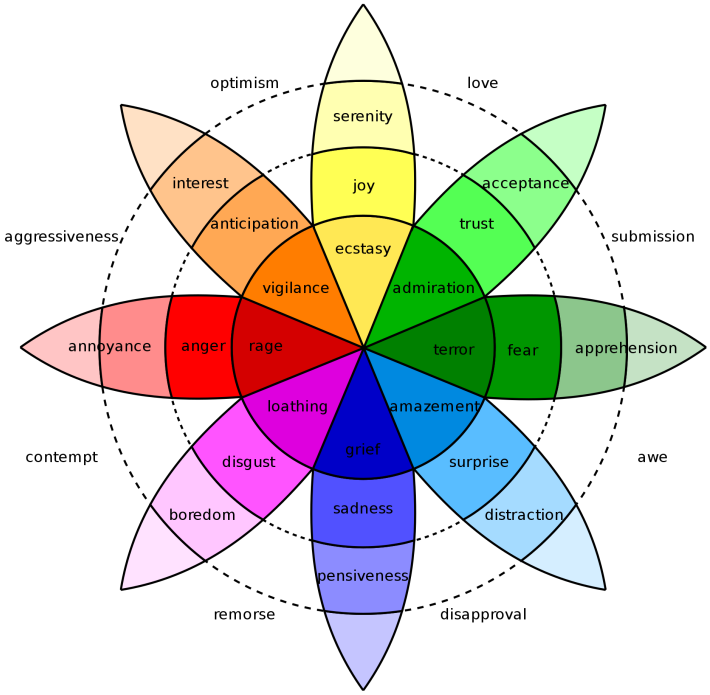


Figure 2.1: Plutchik wheel of emotions

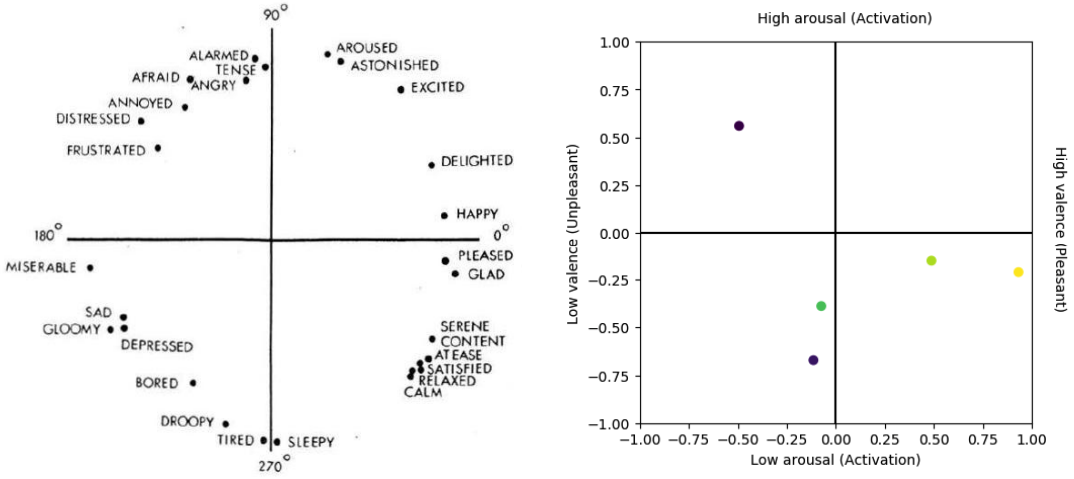
Dimensional models

On 1897 the German physician Wilhelm Wundt also known as the father of Psychology described affect (Gefühl) in German as a feeling pleasant or unpleasant (valence) arousing or subduing (arousal) and strain or relaxation, which is the intensity. Wundt considered the affect as the 6th sense (Wundt, 1874)

The dimensional models attempt to conceptualize human emotions on a 2-d or 3-d dimensional graph. Most of the implementations are based on the valence-arousal model. The dimensional models contrast classification suggests that there's only one neurophysiological core system responsible for all the human affective states. That's a great advantage compared to the one neural system per emotion proposed other theories of discrete emotion (Posner et al., 2005)

The Valence and Arousal circumplex model was first introduced in 1980 by James Russell (Russell, 1980). The model classifies emotions into a two-dimensional circular shape containing the valence arousal dimensions. Valence corresponding to the horizontal axis and arousal being the vertical axis. The centre of the circumference corresponds to neutral valence and medium level of arousal. fig. 2.2a

h



(a) Circumplex model (Russell, 1980)

(b) Vector model

Figure 2.2: Valence/Arousal models

The Valence/Arousal vector model emerged in 1992 (Bradley et al., 1992) and it was a step further. This model presumes that there's always a valence and arousal dimension and the combination of both determine the point on where a particular emotion lies. For example a negative valence is going to shift the emotion down to the bottom of the vector. This is nowadays the most used model in affective computing for developing emotion-related systems and the chosen to undertake this project.

Novel classification methods

In 2012, Hugo Lövheim proposed a new three-dimensional method to classify emotions based on the monoamines neurotransmiters serotonin, dopamine and noradrenaline. The origin of the cube corresponds to to a situation where all the three substances are low and the arrow can increase depending on the secretion of the substance. The emotions are based on the substance level. (Lövheim, 2012).

The most recent literature on emotion classification is a report where 27 distinct categories of emotion bridged by continuous gradients are shown. The researchers showed that this method can capture more dimensions in the space compared to the Valence/Arousal vector model uncovering new gradients. Overall a great new system but not used among the scientists (Cowen and Keltner, 2017)

2.1.3 Emotions on images

It is been show that the presence of certain shapes, dots or lines on an image makes a big contribution on the emotion transmits. For example, in the following study is shown that the presence of concave lines is related to a happier emotion and by contrast a convex line with a less pleasant emotion (Salgado-Montejo et al., 2017). Aesthetics play a key role on image, photographers create pictures where the subject is very sharp and the background is blurred, on the literature it is been shown in (Joshi et al., 2011) where it is affirmed that the composition, focus and sharpness plays a key role on the sentiment the image transmits. Color also plays one of the most important roles on the emotion the image, for example (NAz and Epps, 2004) showed how black and white pictures are related to negative emotions whereas yellow, green and blue seem to have a positive impact on the the image.

2.2 Existing solutions

This recent years with the advances in computer vision and neural networks, the deep learning technology showed it's potential on tasks such as image classification (Cireşan et al., 2011; Krizhevsky et al., 2012; Szegedy et al., 2015), achieving state-of-the-art performance surpassing humans.

Emotion recognitions systems have been always a hot topic on the academia, historically emotion recognition system used Support vector machines (SVM), a supervised machine learning algorithm used to classify regions on a given data (Michel and El Kaliouby, 2003; Wu et al., 2005), they showed potential and the first considerable results but very limited compared to the technology available nowadays.

2.2.1 Positive/negative sentiment analysis

Several studies showed the performance of neural networks for emotion prediction. From this studies there's approaches classifying if an image is positive or negative (Campos et al., 2015; Yanulevskaya et al., 2008; You et al., 2015). This experiments were at the time the first on applying CNN and showed the potential the technology has on the field.

2.2.2 Discrete categories

After the first applications of CNN to classify positive or negative emotions, the domain of computer vision discovered better architectures outperforming older models (Correa, 2016). Researchers started to analyse beyond the image, for example (Lu et al., 2014) build a CNN

model to recognise emotions based on aesthetic features of the image. Also approaches have been done using the principles of art theory, in (Machajdik and Hanbury, 2010) proposed a new architecture unfolding the image, doing separate analysing on features such as colour, texture, composition and content.

One of the most interesting experiments is Xu et al. (2014). In the research they used different architectures, combined with transfer learning applied to 500k images with different labels. To classify the results the wheel of emotions is used.

Other emotion recognition systems are mostly focussed on extracting the emotion based on the facial expression (Liu et al., 2013; Yu and Zhang, 2015). A wide variety of datasets are available for facial emotion recognition such as (Mollahosseini et al., 2017), a dataset with more than 1 million labeled examples with valence and arousal ratings.

The most relevant research for this project is made by (Kim et al., 2017). In this research they analysed image emotions using the valence/arousal model. They created a complex architecture based on neural networks capable of analysing the scene segmentation, objects and low level features. After this analysis is done they used a basic feed forward network to classify the values, achieving relevant results.

2.3 Datasets

Create emotional datasets is not an easy task because emotions can vary on different cultures. Besides is demonstrated that all humans can perceive the same emotions there's a cultural bias we have to consider when dealing with emotions. the following study showed that the participants likeness of categorizing the same emotion is higher when both are from the same nationality, ethical or regional group. (Elfenbein and Ambady, 2002).

The following datasets presented in the next section are carefully studied and use for the development of this thesis. To do the data gathering, most datasets used an on-line tool called Amazon's mechanical Turk. This tool is a collaborative platform where individuals have to rate/describe a given task in exchange for a monetary payment set by the requester. After that the results are evaluated by experts and the mean of all the answer is mixed for valence and arousal values.

2.3.1 Dimensional (Valence/Arousal model)

The most popular dataset used for emotion recognition is the International Affective Picture System, also known as IAPS (Lang et al., 1997). This dataset consists of 1196 colour images from multiple domains such as everyday objects and scenes. Another dataset with similar

characteristics is OASIS (Open Affective Standardized Image Set) (Kurdi et al., 2017). A dataset containing 900 colour photographs from a broad spectrum of themes. Other datasets with a large range of different categories include Emomadrid (Emotional Picture Data Base) (CEACO, 2018) with 817 labelled images. NAPS (Nencki Affective Picture System) also belongs to the general category of datasets, contains 1356 images from people, faces, animals, objects, and landscapes (Marchewka et al., 2014). One of the most recent and high-quality published dataset on emotion recognition is the Socio-Moral Image Database (SMID) (Crone et al., 2018), this dataset contains 2941 labelled images from a variety of situations.

The Geneva affective picture database (GAPED) was created with the intention to capture different stimuli. The dataset contains 730 pictures of spiders, snakes, and scenes that induce emotions related to the violation of moral and legal norms (Dan-Glauser and Scherer, 2011). In this category we can also include DIRTI, also known as DIsgust-RelaTed-Images (Haberkamp et al., 2017), with 300 disgusting images of food, animals, body products, injuries/infections, death, and hygiene.

The last dataset available of this category and the most complete one and is called IED (Image emotion dataset). This is the dataset used in (Kim et al., 2017) and consist on 10766 labelled images from a wide context, making it the biggest dataset available nowadays for image emotion recognition labelled with valence and arousal values.

Most of the datasets of this category are used to the research of this thesis. you can find the complete dataset built on section 6.1.

2.3.2 Discrete (labeled with categories)

Datasets labelled with categories are also present in the literature, for example (You et al., 2016) a dataset with 23.308 images labelled with 8 different cases designed to be trained in CNN. Other datasets containing labels are available such as (Jindal and Singh, 2015) with 1000 images with an associated label regarding the emotion of the picture. On (Machajdik and Hanbury, 2010) also built a dataset labelled with 8 discrete categories. The dataset is called art-photo and contains abstract paintings and photographs, summing a total of 1035 labelled images.

Chapter 3

Research Methodology

This section introduces to the topic of deep neural networks and the algorithms used to undertake the project, also gives a broad insight on the metrics used on this project defining the cost function and the evaluation metrics used on the implementation.

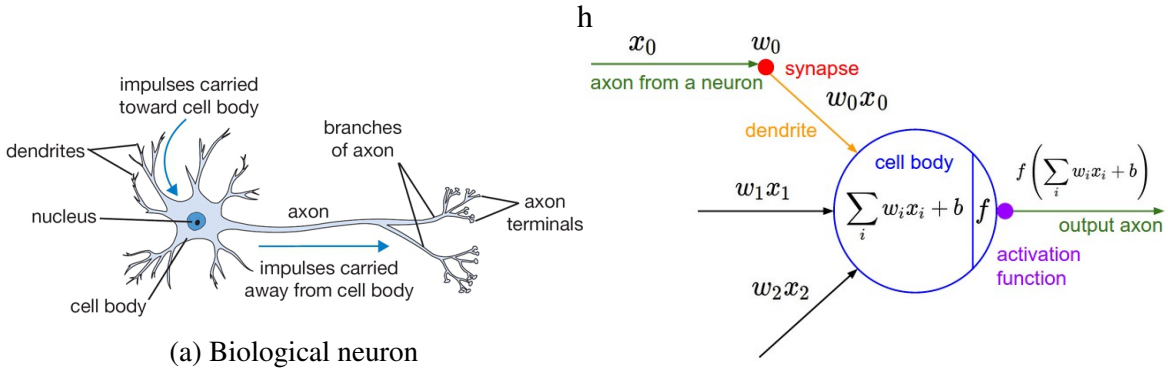
3.1 Neural Networks

Neural networks or artificial neural networks (ANN) are inspired on biological neurons and are a combination of artificial neurons stacked on layers (Hagan et al., 1996). The biological neuron receives inputs through the dendrites and outputs a signal impulse sent from their axons to the terminals where the axon branches out and synapse to other dendrites fig. 3.1a. In the artificial neurons, the input x_i (axon) interact with the weights w_i (dendrites). This weights are learnable and control the influence of the neuron on the model. If the final sum of the operation is above a certain threshold, the neuron sends a spike along it's axon. The mathematical formula for a single neuron can be interpreted as the following:

$$a_i = f(\sum_i (w_i x_i) + b_i)$$

where a_i represents the activation of the neuron, $f(x)$ is the non-linear activation function, w_i is the weight, x_i is the neuron's input and b_i is the bias related to the neuron.

We can describe neural networks as a collection of multiple artificial neurons stacked in layers. The output of a certain neuron becomes the input of another neuron in the next layer. This type of networks are called feed-forward networks. The basic structure of a feed-forward neural network consists on a input layer containing the inputs of the neural network and an output layer with the probabilities of the value to be found. The layers between the input and the output layer are called hidden layers and the values of the neurons



(a) Biological neuron

(b) Artificial neuron

Figure 3.1: Comparison between the biological neuron and the mathematical model (Karpathy, 2016)

contained in hidden layers are based on the input values. The term *deep neural networks* is used when we have a neural network with multiple hidden layers (LeCun et al., 2015).

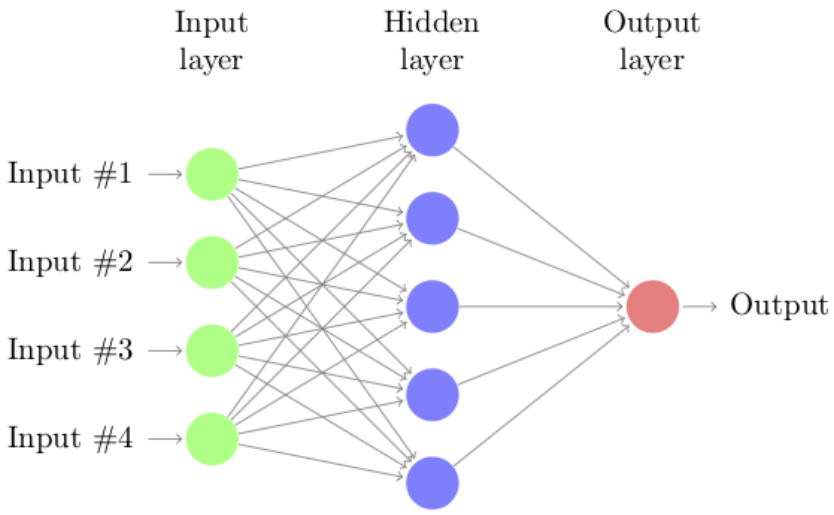


Figure 3.2: Neural network with 1 hidden layer

3.2 Convolutional neural networks

Convolution neural networks are a sub group of deep learning that focuses on computer vision and image processing. Convolution neural networks have achieved state-of-the-art performance in a broad array of high-level computer vision tasks, including image classification, object detection, image segmentation, pose estimation... They have the capability of

h

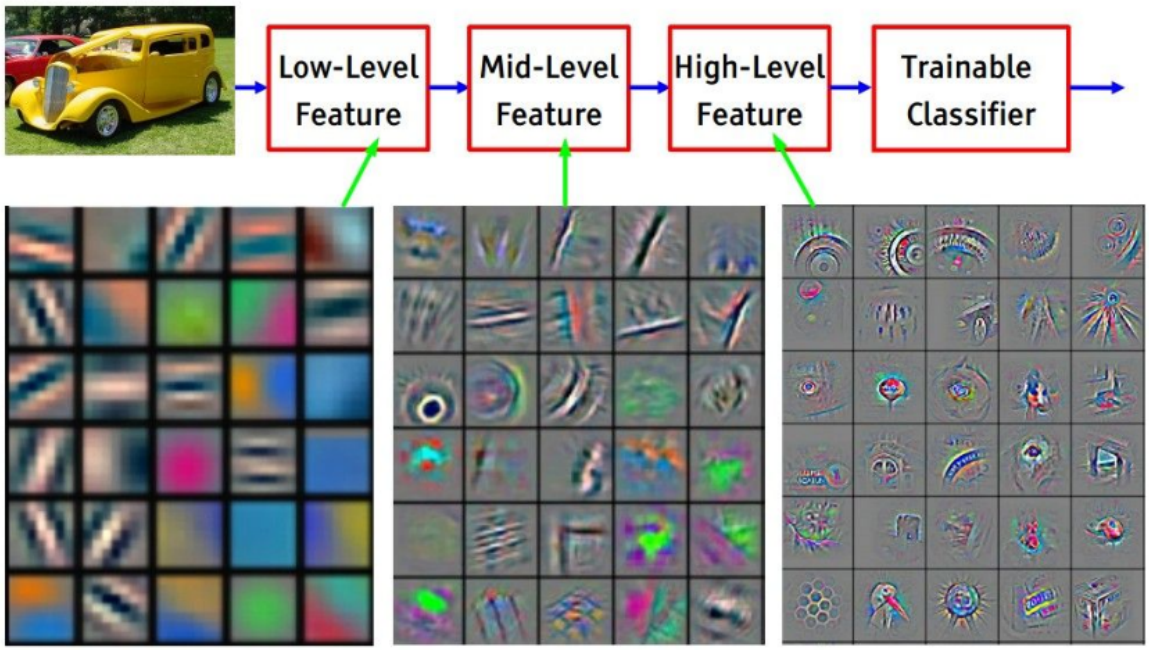


Figure 3.3: Feature Visualization CNN trained on ImageNet (Zeiler and Fergus, 2014)

extracting features and patterns from images to make predictions as we can see in the fig. 3.3 . The way they operate is very similar to standard neural networks but since are designed to work with images the input consist of an array of $width \times height \times depth$ where the width and height is the pixel size of the image and the depth is the number of channels, 1 for black and white pictures and 3 (RGB) for colour images. This thesis is going to focus on the use of convolutional neural networks for the implementation of the artefact.

3.2.1 Convolutional layers

Convolutional layers as the name suggest are the main layers used in CNN and consist on a set of learnable filters that extend through the spacial volume of the input to identify low level features on the images. For example as we can see in fig. 3.4 we have an input of an image that is $32 \times 32 \times 3$ if we apply a 3×3 filter computing dot products across the width and the height of the image volume at any position this filter will produce a new 2d activation map with the responses of the filter at every spatial position and eventually learn that some filters activate

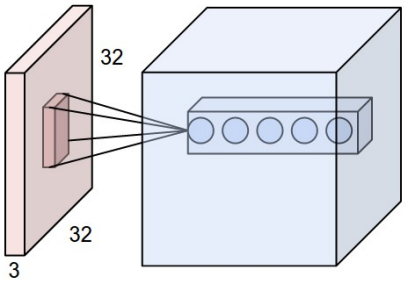


Figure 3.4: Convolutional layer filter (Karpathy, 2016)

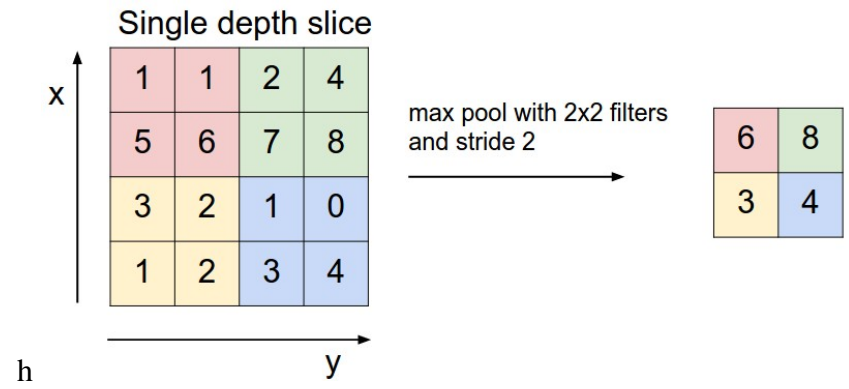


Figure 3.5: Maxpooling with a (2×2) kernel and stride of 2. (Karpathy, 2016)

when they they see some type of visual feature such as
edges, colours, objects...

When applying filters there's 3 hyper-parameters we have to consider. The first one is the depth and corresponds to the number of filters we are going to apply on each input. The second one is the stride and refers to the number of hops between filters, for example a stride of 1 will move the filter 1 pixel at a time along the input. The last hyper-parameter to consider is the padding that allows to add additional data (normally zeros) around the border of the input to match the input and the output size.

Taking the previous example but this time adding the hyper-parameters applying a 5×5 filter with depth 32, a stride of 1 and zero-padding to the $32 \times 32 \times 3$ image if we apply the following formula:

$$Output\ dimension = \frac{Input - Filter + 2 * Padding}{Strides} + 1 = \frac{32 - 5 + 2 * 0}{1} + 1 = 28$$

The result of the convolution will be $28 \times 28 \times 32$. Note that the depth (32) always adds up to the filters and is not included in the equation.

3.2.2 Pooling layers

The pooling layers, also known as down-sampling layers are normally used between conv layers and it's function is to reduce the spatial size of the previous layer to reduce the amount of parameters and make the network more efficient reducing computational resources. A common use of the pooling layers is max-pooling, showed in fig. 3.5, where the max() is taken from the filter size with a stride of 2.

h

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Figure 3.6: Batch normalization layer applied to activation x over a mini-batch.(Ioffe and Szegedy, 2015)

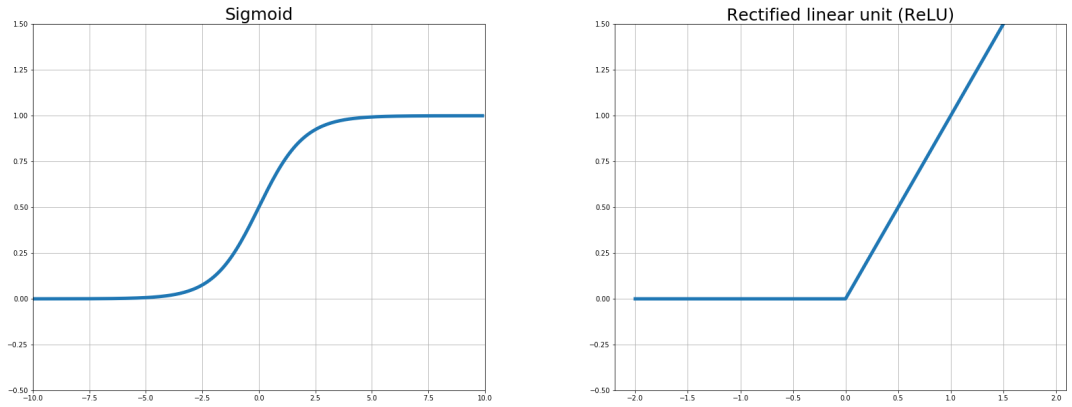
3.2.3 Batch normalization layers

The batch normalization layers are a very new and they were defined in (Ioffe and Szegedy, 2015). This layers are used in CNN between convolutional and activation(relu) layers and, as the name suggests they normalize and speed-up learning by allowing each layer to learn more data independently by itself. The way it works is by normalizing each input channel across a mini-batch by subtracting the mini-batch mean and then dividing it by the mini-batch standard deviation. After this operation is done the layer shifts it's input by a learnable offset β and scales by a learnable factor γ, fig. 3.6. This 2 parameters are the leanable parameters in the batch-norm layer and they can decide if the update is "worth" or useless

3.2.4 Activation functions

As we specified in section 3.1 the artificial neuron calculates the weight and sum plus a bias of the input and then decides if it can be activated. If we don't specify any threshold the neuron can range from $-\infty$ to ∞. Dealing with high values is usually a problem when dealing with neural networks because computers work better with smaller numbers. If we use very big values our system can be inefficient. The role of the activation function is to determine if the certain neuron is activated and if so how much.

Historically, the sigmoid function, defined as $f(x) =: 1/1+e^{-x}$ was the standard the field but years later the ReLU $f(x) = \max(0,x)$ non-linear activation function was discovered



(a) Sigmoid (b) ReLU

Figure 3.7: Activation functions

This function converts to 0 all the negative outputs and only learns the positive values. This function is more efficient than the older methods.

3.2.5 Dropout

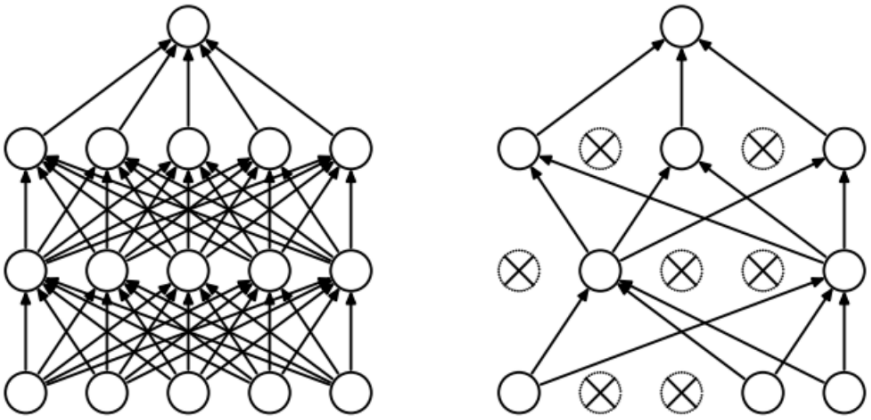
Dropout is a technique used to prevent overftting. It is very easy to understand when is seen it visually (fig. 3.8), if we include dropout, the network automatically drops units during training. This technique allows the network to perform better on testing data because prevents to adapt into training data.

3.3 Metrics

In CNN the metrics are used to train and evaluate the model. They give an insight on how the model is performing using the training data.

3.3.1 Loss function

The loss, also known as cost function of the chose for this problem is mean squared error (MSE). MSE is the most used error function to use in regression problems. This function measures the average of the squares of the errors, calculating the difference between the prediction $(\hat{y})$ and the real value (y) in order to calculate the error. The mathematical formula



h (a) Standard Neural Net (b) After applying dropout.

Figure 3.8: Dropout applied to a neural network (Srivastava et al., 2014)

for the loss function is the following:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

3.3.2 Optimizer

The optimizer role is to minimize the loss function. There's a lot of research done in multiple optimizers but the scope is too broad for the purpose of this research. The optimizer of choice is the evolution of Gradient descent, Stochastic gradient descent (SGD).

Gradient descent tries to minimize the cost function $J(\theta)$ by updating the parameters in the opposite direction of the gradient of the cost function $\nabla_\theta J(\theta)$ the following equation is updated according to a parameter called learning rate η that determines the size of the steps the function takes to reach a (local) minimum. The algorithm computes the the gradient of the cost function for the entire training dataset:

$$\theta = \theta - \eta \nabla_\theta J(\theta)$$

SGD in contrast performs an update to every training example x^i respect to it's label y^i

$$\theta = \theta - \eta \nabla_\theta J(\theta; x^i; y^i)$$

Generally SGD is calculated with respect to few training examples to avoid high variance between parameters and avoid the loss function to fluctuate between values.

3.3.3 Evaluation metrics

For the evaluation of the solution 3 metrics are implemented.

Root Mean Squared Error (RMSE)

RMSE is the most common evaluation metric when evaluating a continuous output. It's role is to calculate the difference between the predicted value ($\hat{y}$ and the real label y to give a predicted standard deviation value. The mathematical intuition is described as the following:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

Pearson correlation coefficient (PCC)

Even RSME is the standard evaluation metric, it can be very affected to outliers and don't consider the covariance of the data. The Pearson correlation coefficient (ρ) tries to overcome this problem, measuring the linear correlation between 2 variables $(\hat{y}, y)$. The output of the function is a value between -1 and 1, where 1 is total positive (means a good fit), 0 means no direct correlation and -1 means very negative correlation. The mathematical formula is formed like the following.

$$\rho = \frac{\text{cov}(\hat{y}, y)}{\sigma_{\hat{y}}\sigma_y}$$

Sign agreement metric (SAGR)

Correctly predicting the signs is essential in emotion-prediction systems. In the dataset used for this project, the value of valence and arousal values fall within the interval of [1-9] where for valence 1 is low (negative) and 9 is high (positive) and, referring to the arousal 1 is low (passive) and 9 high(active). SAGR, defined in (Nicolaou et al., 2011), is used to evaluate the sign agreement of the valence and arousal predictions. SAGR is defined as:

$$SAGR = \frac{1}{n}\sum_{i=1}^{n}\delta(sign(\hat{y}_i), sign(y_i)) \tag{3.1}$$

where δ is the Kronecker delta function, defined as:

$$\delta(\hat{y}, y) = \begin{cases} 1, & \hat{y} = y \\ 0, & \hat{y} \neq y \end{cases} \tag{3.2}$$

Chapter 4

Time Management

This section gives a brief description on how the author achieved the sucess on this project. Gives an overview of the techniques used to overcome

4.1 Project Management

This process was a challenge for the author of the thesis because before the beginning of the start date he didn't had any knowledge about the topic. Applying planning and project management skill was crucial for this regard. Before the start of the project a detailed plan with all the variables was made. The detailed plan can be found here fig. 4.

The initial plan has been modified along the process. At the beginning the author tried to re-frame the project plan but after several weeks, the initial plan was discarded due to the work-load of other units and the optimism of the plan itself. The initial plan was re-framed and author decided to apply other strategies to accomplish the objectives created, applying an adaptive project management framework based on the workload of other units.

This approach consisted on boiling down the final goals into smaller sub-goals and gradually putting all the pieces together. For example this approach was used while writing software: Creating the script to run the models was a chunk:

1. Create the script
    (a) Data processing
        i. Read data
        ii. Resize the images
    (b) Create the model
        i. Create layers

A. Apply filters

B. Apply activation function

This method resulted to be more efficient than the previous classic planning approach. Also the author applied the learn-by-doing approach. At the beginning, he was obsessed to learn the theory behind the problem taking on-line MOCC'S and reading books about the topic. But he didn't understand clearly the intuitions behind the material until he started hands-on experimenting about the development of the solution.

4.2 Research skills

Keeping up with the research on this field was a challenge. The author had some basic research skills before the project started but they improved exponentially during the implementation of this thesis.

The author tried to apply the most novel techniques available on the field, using the latest tools available. At the beginning of the project, when the deployment of the artefact started, Tensorflow (a framework used to run code) was on version (v1.0). Months later, close to the end of the deadline, Tensorflow is already on the version (v1,8). Eight versions have changed in a period of 6 months. This was a challenge because progressively new features were being added and old functions stopped working. It required a lot of adaptation and problem-solving skills.

Most of the intuitions given in this thesis come from well-known papers that in some way revolutionized the field of computer vision. The author read this papers alongside the implementation to apply the most updated techniques. Also he got insight and inspiration to write the research methodology based on this regard.

Chapter 5

Requirements

5.1 Hardware

The most important requirement when dealing with CNN is the computing power. For the implementation of this project the author of the project decided to use the Google Cloud platform for the implementation of the models. The machine resides in europe-west1-d zone (Belgium) and is used to to train the models. The hardware has been carefully thought to keep the costs at the minimum and have the best performance available and is made. The system is from the following specifications:

- $1 \times$ NVIDIA Tesla P100 GPU
- $4 \times$ 2.2 GHz Intel Xeon E5 v4 (Broadwell) CPU's
- 15Gb Ram
- 50gb SSD hard drive
- Ubuntu server 16.04 as the main operating system

Training Deep neural netwoks have a high computational cost, for example on if we take a the experiments ran for this project the images are resized to 150×150 pixels. If we take this input and pass it to a VGG16 model, results on a 14.7 million parameters that need to be updated during the training process. CPU's is designed to make small computations very fast, for example dividing few numbers in a CPU is very fast. However CPU's struggle when dealing with large amounts of data, for example the VGG16 model used above. When multiplying matrices of thousands of numbers GPU's are more efficient because are designed to work and render 3d structures on parallel, that means that they have high amounts of

memory and computational power available, this makes them perfect to train deep learning models.

As we concluded, the GPU is the most important hardware part for training DNN. The GPU used to undertake the project is the Nvidia P100, the fastest available on Google cloud with 16GB of internal memory. To better use all the resources available on the GPU the CUDA toolkit and the cuDNN (cuda Deep Neural Network) library have been installed. The cuDNN library provides a extra performance when training neural networks because it tunes the GPU according to the common operations used to develop deep learning models.

5.2 Software

The language of choice used for the implementation of the code is Python. The framework is a high level object-oriented programming language released on 1991 for the Dutch computer scientist Guido van Rossum for better code simplicity and readability. One of the main features of Python is the syntax, it uses white spaces for indentation rather than brackets used on most programming languages, features an automatic memory management and a dynamic type system. Python is the main programming language that runs in top of this project.

The interface used to interact with the server and run the code is Jupyter notebook (Pérez and Granger, 2007) (Formerly known as IPython Notebooks) is an interactive tool that allows users to run code in the browser. As said before, the server used to develop the software for this thesis is hosted on the Google cloud platform and is based in Belgium. The Jupyter interface allowed the author to run the code straight from the web browser from everywhere. In top of Python and Jupyter, few libraries were used to handle scientific operations:

- Keras (Chollet et al., 2015): Used to the main development of the models. Keras is a high level API running on top of Tensorflow and it allows to do a very simple prototyping and implementation of models used in the latest research available. It is very easy to use and have a great written documentation available.

- Tensorflow (Abadi et al., 2015): Initially used for the development before the author decided to switch to Keras. The Keras API runs on top of the Tensorflow developement framework. Tensorflow is an open source library mainly used for machine learning applications, is developed and mantained by Google and the community.

- Numpy (Jones et al., 2001–): The reference python scientific computing package, employed along with Keras to perform linear algebra transformations and numerical operations.

- Scikit-learn (Pedregosa et al., 2011): Also known as sklearn is an open source machine learning Python API build on top of numPy with some useful functions for data manipulation used for this thesis.
- Open-CV (Open Source Computer Vision Library) (Bradski, 2000): As the name suggests is an open source Python computer vision library used in this research for image manipulation.
- Pandas (McKinney, 2011): A python library developed for data analysis tasks, it is used to read and manipulate CSV files from disk to the interface.
- Matplotlib: Library inside the SciPy (Jones et al., 2001–) package. It is used to visually show insights about the data generated by the models.

Chapter 6

Design

This section provides an overview of the experimental setup. The first part describes how the dataset has been built and the second part gives a brief overview of the models used for the deployment of the experiment.

6.1 Dataset

On a supervised learning problem the dataset is the most important piece. In deep learning is demonstrated that the accuracy of the model varies depending on the amount of data we have, on more training data available, better accuracy the model is going to have (Goodfellow et al., 2016; Valdenegro-Toro, 2017). The problem with training models with a small datasets is that the algorithm is not capable of learning all the features the image, as a result makes the model more prone to overfitting.

Following the previous rule and the research on datasets stated in the literature review a new dataset called jndataset has been created. Jndataset is the biggest available dataset for emotion recognition labeled with valence and arousal values consisting on 17810 images. The dataset is a mix of most of the datasets available on the literature. All the datasets used to create jndataset along with the number of examples can be seen in table 6.1. To have more information about the datasets, all of them are carefully described in section 2.3.

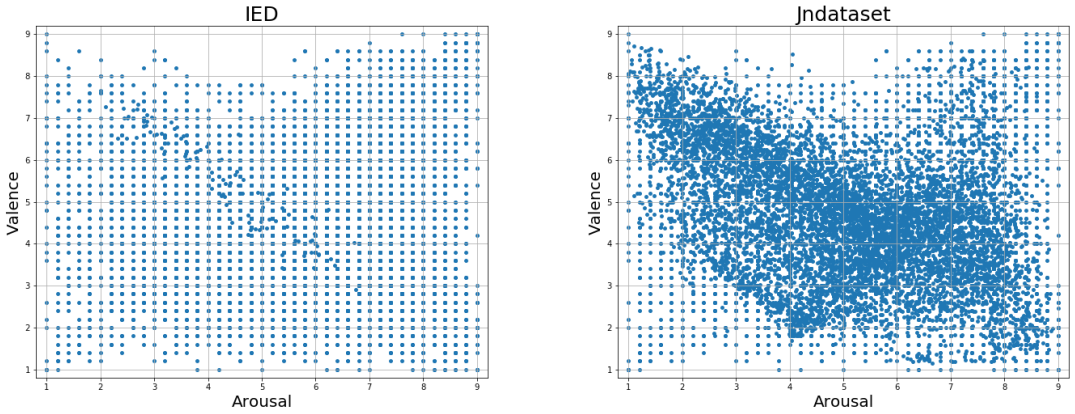
As we can see in the table table 6.1 not all the values were labeled with the same range, jndataset employs the standard rule of valence and arousal between 1-9 The formula used to rescale the inputs is:

$$(((OldValue - OldMin) \times (NewMax - NewMin)) \div (OldMax - OldMin)) + NewMin$$

Jndataset is the most complete dataset available nowadays. If we plot a scatted distribution of the valence and arousal values, we can see that jndataset occupies a broader spectrum

Dataset	Range	Number of examples
IED	$1-9$	10766
DIRTI	$1-9$	300
Emomadrid	$-2-2$	817
GAPED	$0-100$	730
SMID	$1-5$	2941
OASIS	$1-9$	900
NAPS	$1-9$	1356

Table 6.1: Dataset used to create jndataset



(a) IED

(b) jndataset

Figure 6.1: Emotion distribution comparison between IED and jndataset

compared to IED, the biggest existing solution until the date fig. 6.1. This will allow the system to be trained on more examples and be more accurate.

On fig. 6.2 we can see the total distribution of valence and arousal values. Comparing them side by side can be seen that the valence values are slightly higher than the arousal values. This can be caused by a slight classification bias made by the people who rated the datasets, because individuals have a tendency to share positive images rather than negative. The Arousal values are more normalized across the space but it can be seen that few examples with extreme values of arousal are missing.

6.2 Models

4 models are used to test the performance of the system. All the models have 2 layers in common. The first layer and the last one. The first layer consists on an input layer

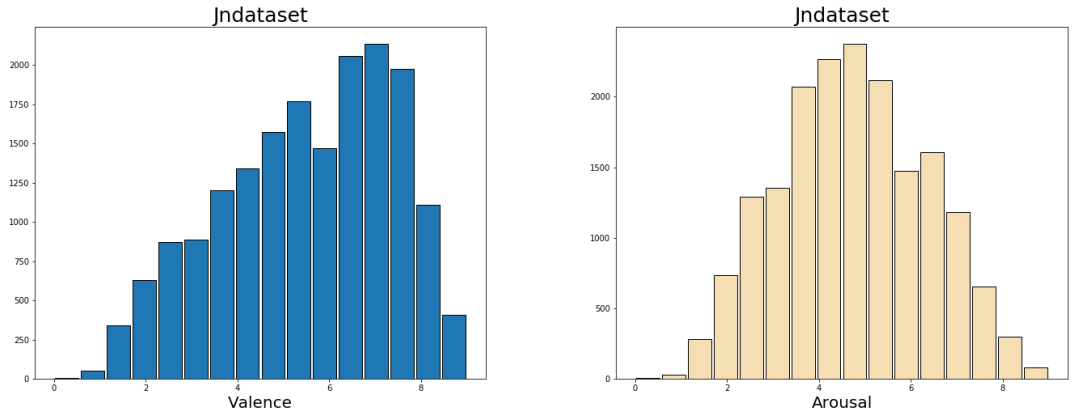


Figure 6.2: Histogram of valence and arousal values in the jndataset

of $batchsize \times 150 \times 150 \times 3$ and the output layer is a dense layer of shape $batchsize \times 2$ containing the continuous output values of valence and arousal.

6.2.1 Emosepar

The novel model introduced in this thesis is called Emosepar. It is established with this name because is designed especially for emotion recognition and employs separable convolutions introduced in section 6.2.4.

This model consist on a simple, basic and efficient architecture made of separable convolutional layers. The architecture is formed from an input block:

$$convolution(32,3,3) + batchNormalization + ReLU + maxpooling(3,3)$$

followed by 3 blocks like the following shown in fig. 6.3. This blocks consist on a

$$SeparableConvolution + batchNormalization + ReLU + SeparableConvolution + batchNormalization$$

All the block layer employ a filter size of (3×3) for the separable convolution layers, the difference is between the depth that is multiplied by 2 in every block, the first block has 128, the second one 256 and finally 512 for the last block. All the block layers interconnect with a Max pooling layer of (3×3) filter size. The last step of the network, before the output lands on the final dense layer is a global average pooling to flatten the output. Once we have the pixels flatten, Dropout is applied with a rate of 0.5.

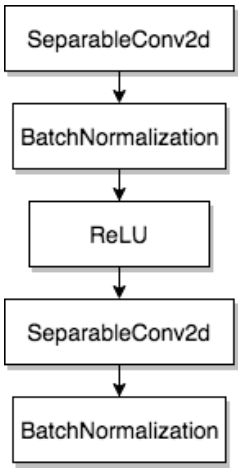


Figure 6.3: Emosepar block

6.2.2 VGG16

The VGG16 model was introduced by K. Simonyan and A. Zisserman from the Visual Geometry Group (VGG) at the University of Oxford (Simonyan and Zisserman, 2014) and it outperformed all the other models at the time. The architecture can be seen in fig. 6.4 and consist on 13 convolution layers along with (3×3) filters and 3 fully connected layers. The convolution layers on the network are divided in blocks, and between each block a max pooling of (2×2) is applied. This network is chosen because is a bridge between very complicated architectures like inception and very simple CNN.

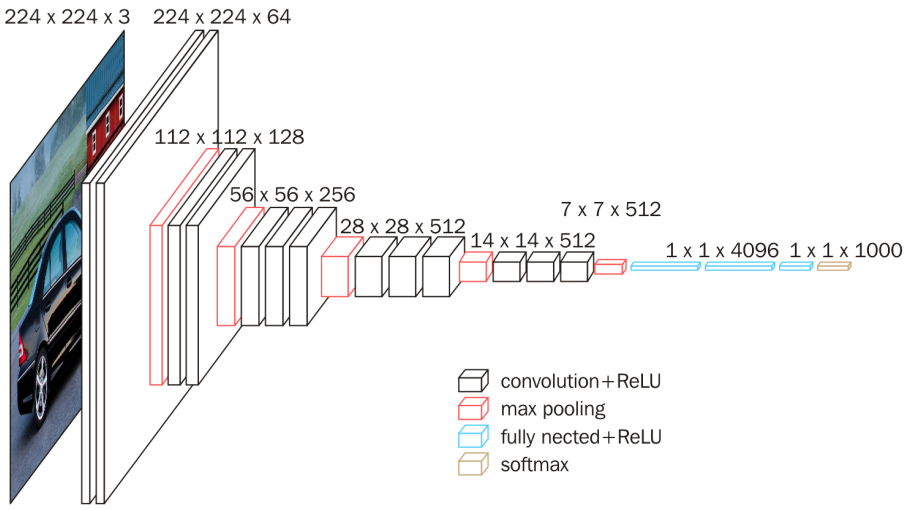


Figure 6.4: Example of a VGG16 architecture

6.2.3 Inception v3

Incepction networks were introduced by Google in 2014 (Szegedy et al., 2015) with the idea of overcoming the limitations of traditional convolutional neural networks, reducing computational resources for accurate image classification. Classic convolutional layers can only learn from a filter applied into a 3D space, with 2 spatial dimensions (width and height) and a channel dimension, the idea behind inception layers was to unify multiple filters in a single layer, making possible to learn more levels of abstraction. As we can see in fig. 6.5 a single inception layer uses (1×1), (3×3) and (5×5) convolutions along with a (3×3) max pooling. Convolution operations are computationally expensive. That's why the authors of the architecture suggested (1×1) convolutions before the convolutional operations, to reduce the dimensionality of the output and improve the overall performance.

The Version 3 of inception was introduced in (Szegedy et al., 2016) and it is a more optimized version on the original Inception, The author showed that convolutions with filters larger than 3×3 are inefficient. They removed the (5×5) filters and they used batch normalization to improve the efficiency of the network removing the need of having fully connected layers at the end of the network, making a more optimized implementation overall.

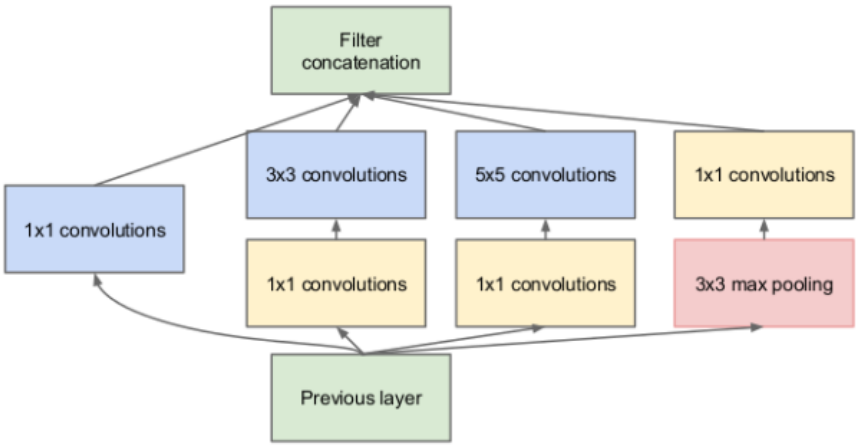


Figure 6.5: Inception layer (Szegedy et al., 2015)

6.2.4 Xception

Xception (Extreme inception) is an architecture introduced by François Chollet (Chollet, 2016) and it consist on a novel implementation of the inception layers. The hypothesis behind the inception module is that "cross-channel correlations and spatial correlations are sufficiently decoupled that it is preferable not to map them jointly." What the author wanted to

state with the previous sentence is that when performing a convolution to an image the filters is applied through the volume of the image, for example when performing $64(3 \times 3 \times 3)$ filters to an image $(150 \times 150 \times 3)$ this results on 64 features maps of (148×148) and thus have relevant information for our task and provide the inputs for the next filter to do another convolution. On Xception instead of applying a filter to the spatial volume of the input it maps the spatial correlation for each input channel separately, applying a (1×1) convolution filter to the input and then employ a separate filter to each one of the outputs done by the first convolution (fig. 6.6). If we take the previous example, having an input image of $(150 \times 150 \times 3)$, first a $64(1 \times 1 \times 3)$ convolution is applied resulting on a $64(150 \times 150)$ feature vector, after the feature vector is created, a (3×3) filter is applied individually to each of the 64 feature vectors created previously. This new layers introduced are called Separable Convolutions.

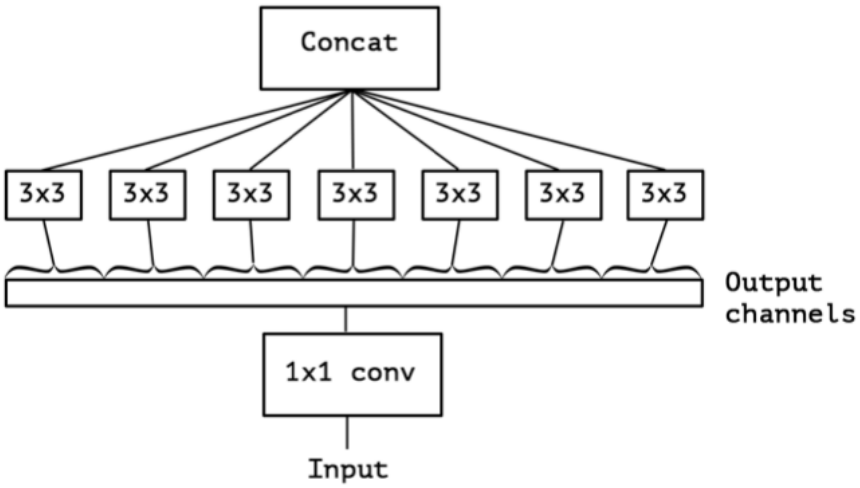


Figure 6.6: Separable Convolution layer (Chollet, 2016)

Chapter 7

Implementation

The experiment has been set-up in the google cloud server described in chapter 6 and Jupyter notebook was used to run the experiments.

7.1 Data pre-processing

For the implementation the same process is done for the implementation of the models. All the images are stored in the disk of the machine, the first task we do is load the images of the dataset along with the labels, shuffling all the examples at the end of the process after loading the data loaded from the file system. Referring to the image size the following study showed a relationship between the model performance and the image resolution(Valdenegro-Toro, 2017). For the purpose of this research and the computational resources available, all the images are resized to 150×150 pixels and are appended with the correspondent valence and arousal label along the procedure. To make the process easier and train the models with the same data order after the data is created, the array containing all the images along with the corresponding values is saved to disk.

After the data is loaded to the computer, the dataset is splitted between training and validation set table 7.1. The validation data consist on 10% of the whole examples available on the dataset. from the 90% left of the dataset, 25% is used as a testing data and the remaining 75% is used for training. All the training and testing data has been randomized with the same random state, meaning that the same randomised data is used on all implementations. The training data is the data where the model is being trained on. Testing data is where loss function is calculated during training, this are examples that the optimizer has never seen. Applying the loss that way avoids overfitting on the training data, resulting on a better performance overall. The validation test set is used only to evaluate the model performance,

```
def create_data():
    values = pd.read_csv(PATH_LABELS)
    training_data = []
    index = 0
    for img in tqdm(sorted(os.listdir(PATH_IMG))):
        label = values.values[index]
        path = os.path.join(PATH_IMG,img)
        img = cv2.imread(path)
        img = cv2.resize(img, (IMG_HEIGHT,IMG_WIDTH))
        training_data.append([img,label])
        index=index+1
    shuffle(training_data)
    np.save('data_150.npy', training_data)
    return training_data

def preprocessData(data):
    data = np.array(data)
    arr1 = np.array([i[0] for i in data]).reshape(-1,IMG_WIDTH,IMG_HEIGHT,3)
    arr2 = np.array([i[1] for i in data]).reshape(-1,2)
    return arr1, arr2
```

Listing 1: Data preprocessing

it is data that has never seen in training and gives an approximate estimate on how our model is performing.

Set	Range
Train	12021
Test	4008
Validation	1781

Table 7.1: Range used for training/testing and evaluation

7.2 Training

The training objective is to learn from the examples, that's why after the models are compiled, the training function is called. On all implementations a batch size of 64 is used. The number of epochs used on each implementation varied depending on the learning curve of the system, the emosepar model had the slowest learning curve of all the implementation, possibly because used droppout that added noise to the network, resulting on higher loss.

7.2.1 Loss and optimizer

Introduced in section 3.3, MSE and SGD are the choices made for the loss and the optimizer respectively.

SGD is configured with a learning rate of 0.00001, decay of 0.0000001, momentum of 0.9 and nesterov. The weight decay is a hyperparameter thats specifies how much the learning rate decays on each operation, gives good results because on each iteration the loss is reduced, meaning that the learning rate of the optimizer should also be smaller to converge to the local minimum. Momentum and nesterov are 2 additional regularization strategies to converge faster to the local minimum, avoiding errors on the gradient.

```
sgd = keras.optimizers.SGD(lr=0.00001, decay=1e-6, momentum=0.9, nesterov=True)
```

7.3 Data augmentation

One of the main drawbacks when dealing with a supervised learning is that we need a lot of data to train the model. To test the model the author decided to use this techniques to see how they compare with the standard model. Results are shown here table 8.2

Data augmentation consists on creating artificial data based on the available data in our dataset. To compare it with the normal implementation random rotations, shifts, zoom and flips are made on the training examples while training. The data is generated automatically using the following function:

```
from keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(
    rotation_range=80,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True,
    fill_mode='nearest')
```

Chapter 8

Testing and evaluation

This chapter puts together all the research done, evaluates the solution and gives insight about the results achieved.

The training phase was monitored and it went as expected, in fig. 1 we can see how the training loss decayed alongside the testing until the point of maximum learning is reached.

8.1 Model performance comparison

In table table 8.1 we can see the results achieved on each model. Surprisingly the model that achieved most overall performance is Emosepar, scoring the lowest RSME and the biggest SAGR. The model design functioned as expected, dropout made a big difference on the overall performance. This results can be a signal that the dataset is not big enough to learn to be trained on large models and converges better on a small dataset like Emosepar. The inception v3 model showed an incredible correlation on the validation set, meaning that has an incredible covariance respect to the data.

The worst model that performed on the task is VGG16 and Xception achieved the best performance on training but failed on the validation. That may be a symptom of over-fitting on the training data.

8.2 Evaluation with data augmentation

As we described, data augmentation had the potential of improving the overall performance of the model. The model that achieved the best performance on the first task was evaluated generating augmented data explained in section 7.3. The results show no improvement using the technique, all the relevant metrics are worse.

Model	Training			Testing			Validation		
	RMSE	CC	SAGR	RMSE	CC	SAGR	RMSE	CC	SAGR
Emosepar	1.65	0.86	0.61	**1.63**	0.83	**0.62**	**1.66**	0.84	**0.61**
VGG16	1.75	0.90	0.58	1.75	0.91	0.56	1.78	0.92	0.57
Inception v3	1.72	**0.91**	0.58	1.76	**0.93**	0.58	1.80	**0.95**	0.56
Xception	**1.64**	0.84	**0.62**	1.67	0.87	0.60	1.68	0.87	0.59

Table 8.1: Table with the final metrics

Data augmentation	Training			Validation		
	RMSE	CC	SAGR	RMSE	CC	SAGR
Emosepar	1.65	0.86	0.61	1.66	0.84	0.61
Emosepar + Data augmentation	1.63	0.83	0.62	1.69	0.84	0.60

Table 8.2: Comparison of Emosepar with data augmentation

8.3 Evaluation of the valence and arousal values

Results are evaluated individually on the Emosepar model. The individual metrics show that a better performance on the evaluation of the Arousal in favour of the valence, especially on the validation. This can be due the less dynamic range of values, as we saw in the dataset histogram fig. 6.2, the network learned that more positive values are on the valence distribution of the jndataset.

Emosepar	Training			Testing			Validation		
	RMSE	CC	SAGR	RMSE	CC	SAGR	RMSE	CC	SAGR
Valence	1.69	0.81	0.65	1.69	0.83	0.67	1.72	0.82	0.65
Arousal	1.55	0.83	0.67	1.51	0.88	0.61	1.55	0.89	0.60

Table 8.3: Table with the final metrics

Chapter 9

Conclusions and future work

This chapter concludes the thesis, explaining the conclusions on the project and future work that can be done to improve the metrics.

9.1 Conclusions

This thesis showed the potential of CNN applied to emotion recognition evaluating a continuous output. The models evaluated showed a significant performance, achieving good results overall.

This research also showed the performance achieved on the Emosepar model and the potential that have for future implementations on emotion recognition systems.

9.2 Future work

The system has a long way to be improved. Simple tweaks can be done to improve the performance of the system.

9.2.1 Data normalization

It is been shown that normalizing the data have a big impact when dealing with neural networks. The results stated in this thesis showed a big RMSE on all the models. RSME is very sensitive to outliers, if we normalize the valence and arousal values on the distribution of our dataset, for example re-scaling all the values between 0 and 1 the data will be closer and the system will have more potential to improve.

9.2.2 More Data

The model showed good performance on the emosepar model. This model used moderate values of dropout. Dropout is used when small training data is available. Even that the author built the biggest dataset available for emotion recognition using the valence and arousal vector model, it can be seen in the distribution of the dataset that some gaps still need to be filled. The author thinks that more data can substantially improve the performance of the system.

9.2.3 Transfer learning

(Razavian et al., 2014) showed that we can use transfer learning to achieve state-of-the-art models with moderate amounts of data. Transfer learning consist on re-training only the last layer of an existing in (Valdenegro-Toro, 2017) they create various models with transfer learning and without and demonstrated that using transfer learning resulted on better accuracy on the test set, even with models with limited amounts of data.

9.2.4 Regularization

The use of any kind of regularization can improve the performance. Normalizing the weights of the network to make the model less sensitive to outliers.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y. and Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL https://www.tensorflow.org/, software available from tensorflow.org.

Bradley, M. M., Greenwald, M. K., Petry, M. C. and Lang, P. J., 1992. Remembering pictures: Pleasure and arousal in memory. *Journal of experimental psychology: Learning, Memory, and Cognition*, 18 (2), 379.

Bradski, G., 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Cabanac, M., 2002. What is emotion? *Behavioural processes*, 60 (2), 69–83.

Campos, V., Salvador, A., Giro-i Nieto, X. and Jou, B., 2015. Diving deep into sentiment: Understanding fine-tuned cnns for visual sentiment prediction. *Proceedings of the 1st International Workshop on Affect & Sentiment in Multimedia*, ACM, 57–62.

CEACO, 2018. Emomadrid emotional pictures database. URL http://www.uam.es/CEACO/EmoMadrid.

Chollet, F., 2016. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*.

Chollet, F. et al., 2015. Keras. https://keras.io.

Cireşan, D. C., Meier, U., Masci, J., Gambardella, L. M. and Schmidhuber, J., 2011. High-performance neural networks for visual object classification. *arXiv preprint arXiv:1102.0183*.

Correa, E., 2016. Emotion Recognition using Deep Convolutional Neural Networks.

Cowen, A. S. and Keltner, D., 2017. Self-report captures 27 distinct categories of emotion bridged by continuous gradients. *Proceedings of the National Academy of Sciences*, 114 (38), E7900–E7909. URL http://www.pnas.org/content/114/38/E7900.

Crone, D. L., Bode, S., Murawski, C. and Laham, S. M., 2018. The socio-moral image database (smid): A novel stimulus set for the study of social, moral and affective processes. *PloS one*, 13 (1), e0190954.

Dan-Glauser, E. S. and Scherer, K. R., 2011. The geneva affective picture database (gaped): a new 730-picture database focusing on valence and normative significance. *Behavior research methods*, 43 (2), 468.

Darwin, C. and Prodger, P., 1998. *The expression of the emotions in man and animals*. Oxford University Press, USA.

Ekman, P., 1992. An argument for basic emotions. *Cognition & emotion*, 6 (3-4), 169–200.

Elfenbein, H. A. and Ambady, N., 2002. On the universality and cultural specificity of emotion recognition: a meta-analysis. *Psychological bulletin*, 128 (2), 203.

Goodfellow, I., Bengio, Y. and Courville, A., 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Haberkamp, A., Glombiewski, J. A., Schmidt, F. and Barke, A., 2017. The disgust-related-images (dirti) database: Validation of a novel standardized set of disgust pictures. *Behaviour research and therapy*, 89, 86–94.

Hagan, M. T., Demuth, H. B., Beale, M. H. et al., 1996. *Neural network design*, volume 20. Pws Pub. Boston.

Ioffe, S. and Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Jindal, S. and Singh, S., 2015. Manipal Image Sentiment Analysis Dataset. URL https://figshare.com/articles/Image_Sentiment_Analysis_using_Deep_Convolutional_Neural_Networks_with_Domain_Specific_Finetuning/1496534.

Jones, E., Oliphant, T., Peterson, P. et al., 2001–. SciPy: Open source scientific tools for Python. URL http://www.scipy.org/, [Online; accessed <today>].

Joshi, D., Datta, R., Fedorovskaya, E., Luong, Q.-T., Wang, J. Z., Li, J. and Luo, J., 2011. Aesthetics and emotions in images. *IEEE Signal Processing Magazine*, 28 (5), 94–115.

Karpathy, A., 2016. Cs231n convolutional neural networks for visual recognition. URL http://cs231n.github.io/.

Kim, H.-R., Kim, S. J. and Lee, I.-K., 2017. Building emotional machines: Recognizing image emotions through deep neural networks. *arXiv preprint arXiv:1705.07543*.

Krizhevsky, A., Sutskever, I. and Hinton, G. E., 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 1097–1105.

Kurdi, B., Lozano, S. and Banaji, M. R., 2017. Introducing the open affective standardized image set (oasis). *Behavior research methods*, 49 (2), 457–470.

Lang, P. J., Bradley, M. M. and Cuthbert, B. N., 1997. International affective picture system (iaps): Technical manual and affective ratings. *NIMH Center for the Study of Emotion and Attention*, 39–58.

LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. *nature*, 521 (7553), 436.

Liu, M., Li, S., Shan, S. and Chen, X., 2013. Au-aware deep networks for facial expression recognition. *Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on*, IEEE, 1–6.

Lövheim, H., 2012. A new three-dimensional model for emotions and monoamine neurotransmitters. *Medical hypotheses*, 78 (2), 341–348.

Lu, X., Lin, Z., Jin, H., Yang, J. and Wang, J. Z., 2014. Rapid: Rating pictorial aesthetics using deep learning. *Proceedings of the 22nd ACM international conference on Multimedia*, ACM, 457–466.

Machajdik, J. and Hanbury, A., 2010. Affective image classification using features inspired by psychology and art theory. *Proceedings of the 18th ACM international conference on Multimedia*, ACM, 83–92.

Marchewka, A., Żurawski, Ł., Jednoróg, K. and Grabowska, A., 2014. The nencki affective picture system (naps): Introduction to a novel, standardized, wide-range, high-quality, realistic picture database. *Behavior research methods*, 46 (2), 596–610.

McKinney, W., 2011. pandas: a foundational python library for data analysis and statistics.

Michel, P. and El Kaliouby, R., 2003. Real time facial expression recognition in video using support vector machines. *Proceedings of the 5th international conference on Multimodal interfaces*, ACM, 258–264.

Mollahosseini, A., Hasani, B. and Mahoor, M. H., 2017. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *arXiv preprint arXiv:1708.03985*.

NAz, K. and Epps, H., 2004. Relationship between color and emotion: A study of college students. *College Student J*, 38 (3), 396.

Nicolaou, M. A., Gunes, H. and Pantic, M., 2011. Continuous prediction of spontaneous affect from multiple cues and modalities in valence-arousal space. *IEEE Transactions on Affective Computing*, 2 (2), 92–105.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

Pérez, F. and Granger, B. E., 2007. IPython: a system for interactive scientific computing. *Computing in Science and Engineering*, 9 (3), 21–29. URL http://ipython.org.

Plutchik, R., 1980. *Emotion: A psychoevolutionary synthesis*. Harpercollins College Division.

Plutchik, R., 1991. *The emotions*. University Press of America.

Posner, J., Russell, J. A. and Peterson, B. S., 2005. The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology. *Development and psychopathology*, 17 (3), 715–734.

Razavian, A. S., Azizpour, H., Sullivan, J. and Carlsson, S., 2014. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382. URL http://arxiv.org/abs/1403.6382.

Russell, J. A., 1980. A circumplex model of affect. *Journal of personality and social psychology*, 39 (6), 1161.

Salgado-Montejo, A., Salgado, C. J., Alvarado, J. and Spence, C., 2017. Simple lines and shapes are associated with, and communicate, distinct emotions. *Cognition and Emotion*, 31 (3), 511–525.

Schacter, D., Gilbert, D. T. and Wegner, D. M., 2011. *Psychology (2nd Edition)*. New York: Worth. URL http://www.amazon.com/Psychology-Daniel-L-Schacter/dp/1429237198/ref=sr_1_1?s=books&ie=UTF8&qid=1313937150&sr=1-1.

Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Snyder, P. J., Kaufman, R., Harrison, J. and Maruff, P., 2010. Charles darwin's emotional expression "experiment" and his contribution to modern neuropharmacology. *Journal of the History of the Neurosciences*, 19 (2), 158–170. URL https://doi.org/10.1080/09647040903506679, pMID: 20446159.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15 (1), 1929–1958.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A. et al., 2015. Going deeper with convolutions. Cvpr.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z., 2016. Rethinking the inception architecture for computer vision. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.

Tangney, J. P. E. and Fischer, K. W., 1995. Self-conscious emotions: The psychology of shame, guilt, embarrassment, and pride. *The idea for this volume grew out of 2 pivotal conferences. The 1st conference, on emotion and cognition in development, was held in Winter Park, CO, Sum 1985. The 2nd conference, on shame and other self-conscious emotions, was held in Asilomar, CA, Dec 1988.*, New York, NY, US: Guilford Press.

Valdenegro-Toro, M., 2017. Best practices in convolutional networks for forward-looking sonar image recognition. *CoRR*, abs/1709.02601. URL http://arxiv.org/abs/1709.02601.

Wu, Q., Zhou, C. and Wang, C., 2005. Content-based affective image classification and retrieval using support vector machines. *International Conference on Affective Computing and Intelligent Interaction*, Springer, 239–247.

Wundt, W. M., 1874. *Grundzüge de physiologischen Psychologie*, volume 1. W. Engelman.

Xu, C., Cetintas, S., Lee, K.-C. and Li, L.-J., 2014. Visual sentiment prediction with deep convolutional neural networks. *arXiv preprint arXiv:1411.5731*.

Yanulevskaya, V., van Gemert, J. C., Roth, K., Herbold, A.-K., Sebe, N. and Geusebroek, J.-M., 2008. Emotional valence categorization using holistic image features. *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, IEEE, 101–104.

You, Q., Luo, J., Jin, H. and Yang, J., 2015. Robust image sentiment analysis using progressively trained and domain transferred deep networks. *AAAI*, 381–388.

You, Q., Luo, J., Jin, H. and Yang, J., 2016. Building a large scale dataset for image emotion recognition: The fine print and the benchmark. *AAAI*, 308–314.

Yu, Z. and Zhang, C., 2015. Image based static facial expression recognition with multiple deep network learning. *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ACM, 435–442.

Zeiler, M. D. and Fergus, R., 2014. Visualizing and understanding convolutional networks. *European conference on computer vision*, Springer, 818–833.

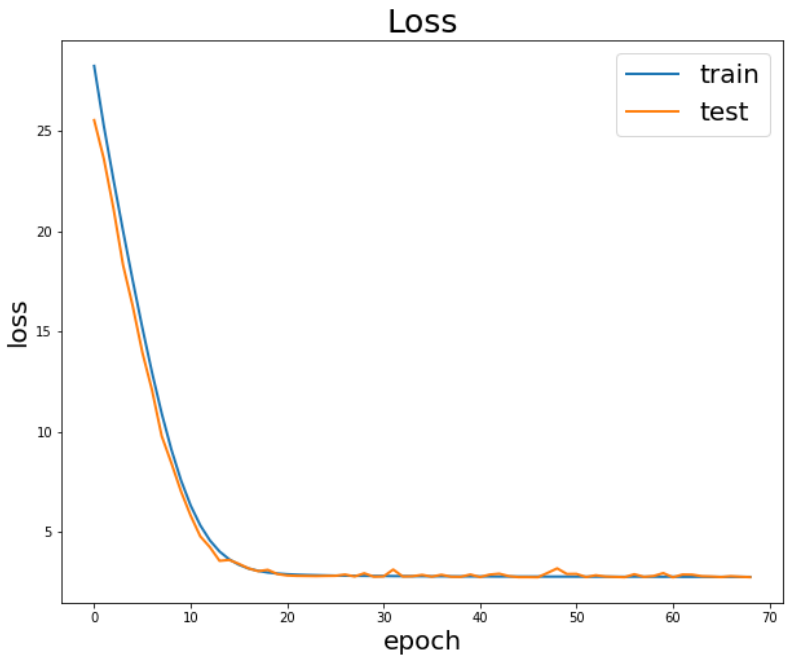


Figure 1: Loss function on the Emosepar model

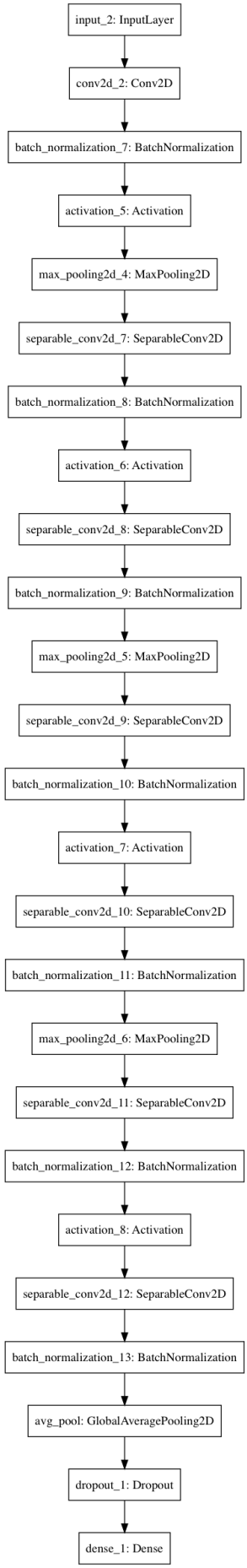


Figure 2: Emosepar full model

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	(None, 150, 150, 3)	0
conv2d_2 (Conv2D)	(None, 148, 148, 32)	864
batch_normalization_7 (Batch	(None, 148, 148, 32)	128
activation_5 (Activation)	(None, 148, 148, 32)	0
max_pooling2d_4 (MaxPooling2	(None, 73, 73, 32)	0
separable_conv2d_7 (Separabl	(None, 71, 71, 128)	4512
batch_normalization_8 (Batch	(None, 71, 71, 128)	512
activation_6 (Activation)	(None, 71, 71, 128)	0
separable_conv2d_8 (Separabl	(None, 69, 69, 128)	17664
batch_normalization_9 (Batch	(None, 69, 69, 128)	512
max_pooling2d_5 (MaxPooling2	(None, 34, 34, 128)	0
separable_conv2d_9 (Separabl	(None, 32, 32, 256)	34176
batch_normalization_10 (Batc	(None, 32, 32, 256)	1024
activation_7 (Activation)	(None, 32, 32, 256)	0
separable_conv2d_10 (Separab	(None, 30, 30, 256)	68096
batch_normalization_11 (Batc	(None, 30, 30, 256)	1024
max_pooling2d_6 (MaxPooling2	(None, 14, 14, 256)	0
separable_conv2d_11 (Separab	(None, 12, 12, 512)	133888
batch_normalization_12 (Batc	(None, 12, 12, 512)	2048
activation_8 (Activation)	(None, 12, 12, 512)	0
separable_conv2d_12 (Separab	(None, 10, 10, 512)	267264
batch_normalization_13 (Batc	(None, 10, 10, 512)	2048
avg_pool (GlobalAveragePooli	(None, 512)	0
dropout_1 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 2)	1026

Total params: 534,786
Trainable params: 531,138
Non-trainable params: 3,648

Figure 3: Emosepar Architecture hierarchy

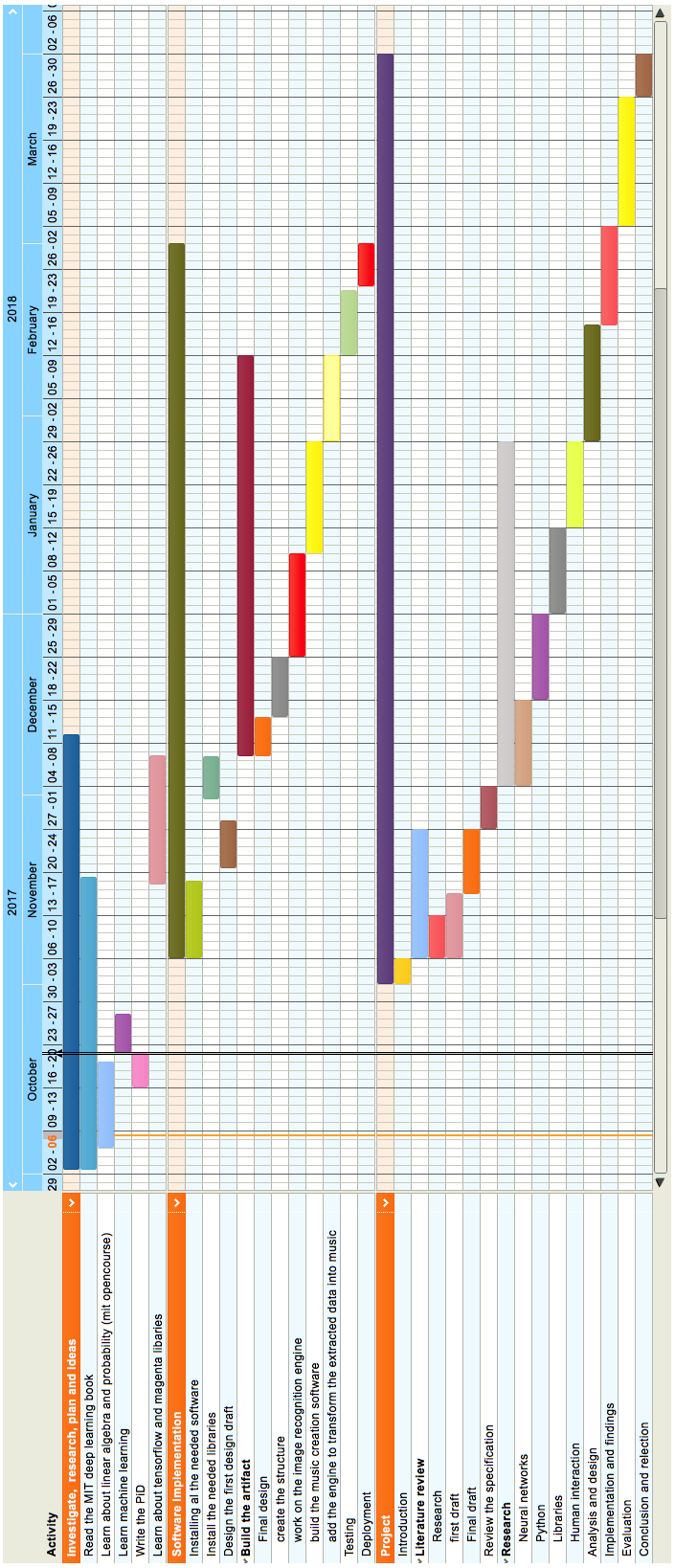


Figure 4: Initial project plan