Universidad Politécnica de Puebla

Programa Académico de Posgrado

Automatic detection of drivers with fatigue using deep learning

Thesis to obtain the grade

Maestría en Ingenería en Sistemas y Cómputo Inteligente

Presenta:

JORDI JAROMIL CRUZ MEDRANO

Director: Dr. Jorge de la Calleja
Co-Director: Dr. Hugo Jair Escalante

Juan C. Bonilla, Puebla, Mexico, Mayo 2017.

El presente trabajo fue realizado en el Laboratorio de Investigación y Posgrado del Departamento de cómputo de la Universidad Politécnica de Puebla, ubicada en Tercer carril del Ejido "Serrano" S/N, San Mateo Cuanalá, Municipio Juan C. Bonilla, Puebla // CP 72640. Apoyo del CONACYT, Beca No. 704756, Programa de Maestría perteneciente al Programa Nacional de Posgrados de Calidad (PNPC-CONACYT)

Universidad Politécnica de Puebla
Maestría en Ingeniería en Sistemas y Cómputo Inteligente

Automatic detection of drivers with fatigue using deep learning

Thesis realized by:

JORDI JAROMIL CRUZ MEDRANO

Approved by ... April 22, 2017.

Profesor	**(Signature)**
Dr. Jorge de la Calleja	....
Dr. Hugo Jair Escalante	....
Dra.Ma.Auxilio Medina	....
Dr. Antonio Benitez	....

Juan C. Bonilla, Puebla, Mexico, Mayo 2017.

Universidad Politécnica de Puebla
Maestría en Ingeniería en Sistemas y
Cómputo Inteligente

(Signature)

. . . .
Jordi Jaromil Cruz
Medrano

Abstract

This document presents a prototype for the identification of fatigue in individuals through the recognition of yawns, closed eyes and inclined head by using deep learning. This with the objective to detect the fatigue that is considered dangerous for any task that requires the use of heavy machinery. For the construction of the prototype we followed the following steps: obtaining the set of images, preprocessing, extraction of characteristics using a pre-trained network giving us as a result the high-level feature extraction, next we did a experiment with weka for determine what machine learning algorithm use for training, after of training we built the prototype. The experimental results indicate that support vector machines algorithm obtains the best results with 98.3% of accuracy in average , with a system capable of correctly detecting the three fatigue manifestations.

Keywords

Machine learning, Deep learning, Convolutional networks, Fatigue recognition, Intelligent transportation systems.

Acknowledgements

To my family Teresa Medrano, Daniel Cruz, Zairi Cruz because I owe it all to you. Many thanks for everything, for all the sacrifices for me and my sister. I'm who I am all thanks to you.

With a special mention to Elisa Marquez , Humberto de la Cruz , Isabel Angeles, Hector Hernandez, Tania Hernandez, Dhalia Hernandez, Mayra Hernandez, Rosita Ocampo, Lourdes Zaldivar and my family in general. Thank you for being my support and believing in me.

I am also grateful to the following university staff: Dr. de la Calleja, Dra. Auxilio, Dr. Benitez and Dr. Hugo Escalante for their unfailing support and assistance for this project thesis.

And finally, last but by no means least, also to all my classmates . . . it was great sharing a laboratory, dreams, homework and more with all of you during the last four years.

Jordi Jaromil Cruz Medrano

Contents

List of Figures

List of Tables

I

Research problem

Chapter 1: Research problem

1.1 Introduction

Diverse studies have revealed that fatigued driving can be more dangerous than driving while intoxicated. This influences the ability to make appropriate decisions as well as performance and time to react to various setbacks that can arise in the moment of driving. A fatigued driver is a risk for himself and for other users of roads. Fatigue produces a significant increase in the number and range of errors in driving, with decreased attention and level of care for the vehicle. There is a large percentage of accidents caused by tired drivers, which leads to monetary losses, injuries and even death. According to the European Transport Safety Council (ETSC), fatigue at the wheel causes almost 20% of accidents in commercial transport vehicles [6].
A viable strategy for solving this problem is the development of systems that monitor, in real time, the physical state of the drivers. This strategy is not new, and several alternative systems have already been developed. Some of them have been depleted by various inconveniences such as: being too invasive for the driver, being imprecise by not considering external factors such as the state of the road, the vehicle, and even the ability of the driver. In the strategies previously proposed, one of the most promising approaches is related to the use of machine learning and computer vision. These two disciplines as a whole allow for the development of systems that can detect, in real time, what was previously recorded.
This research aims to develop a computer vision system that allows for the recognition of drivers when they show signs of fatigue. For the development of this system, we will use deep learning, one of the most recent approaches in the area of machine learning that has obtained very good results in various applications for the recognition of objects in general.
Experimental results shows that that Deep learning significantly increase accuracy compared with older machine learning models. We have also our implementation in Matlab.

1.2 Objectives

The objectives of this research are as follows:

1.2.1 General Objectives

To develop a software prototype for detecting drivers with fatigue by using deep learning.

1.2.2 Specific Objectives

- To characterize images showing signs of people with fatigue such as yawning, inclined head and closed eyes.

- To develop models to recognize people that are yawning, inclined head and closed eyes.
- To build a prototype to manage the recognition system.

1.3 Justification

Many people are unaware of the risks of fatigued driving, whether mild, moderate or severe. "Some researchers consider that over 40% of accidents are closely related to fatigue [6]." Among the groups most at risk of causing an accident due to being in a state of fatigue are: young drivers, night shift workers, public transport drivers and commercial drivers. "1 in 5 traffic accidents occurs because of sleep and fatigue, and this is among the top five causes of casualties [6]." Due to the above, it is important to develop systems in general that allow the detection of conductors that show signs of fatigue, such as closing the eyes periodically and for prolonged periods (not blinking), tilting the head forward and yawning. With the development and implementation of these types of systems, we could prevent a high percentage of accidents, avoiding not only economic losses, but also human lives. Using computer vision systems trained with machine learning algorithms, we can obtain more robust and complete prototypes using recent approaches that have delivered very good results, such as deep learning.

1.4 Overview

The rest of the thesis is organized as follows: Chapter 2 presents the background for this research work as machine learning area and fatigue condition. Chapter 3 explains the methodology used. We describe the tasks of related work divided into two parts: the first one is the necessary task for obtain the models and the other one the task for building the prototype. Chapter 4 presents the results and finally, Chapter 5 has the conclusions contributions and future work.

II

Theoretical Framework

Chapter 2: Theoretical Framework

In this chapter we provide an overview of sources we have explored while researching. The approached research problem, also we show how our research fits within field of study.

2.1 Fatigue

The Highway Traffic Safety Administration, estimates that 100,000 accidents occurred because of tired or distracted people. Out of these accidents, 1,500 people died, 71,000 people were wounded, and 12.5 millions dollars were lost in damages [6].

Fatigue could be described as the loss of energy such as physical or mental, and is a common complaint. We need to remember that it is a symptom and not a disease and is different than drowsiness, which is the necessity to sleep. Mental fatigue is a feeling of decreased cognitive performance. It can manifest as somnolence or lethargy. In any case, this could be dangerous when performing tasks that require constant concentration, such as driving [6].

While it can vary in each person, people with fatigue have some of the following complaints: Distraction and loss of concentration, Loss of sensory functions and perception, Slowness in making decisions and Automatic movements.

Fatigue symptoms may include: Yawning, Inclined head, Nodding off, Closed eyes, Cramps and Headache.

In this thesis, we focus in the following symptoms: yawning, inclined head and closed eyes.(Figure 2.1).

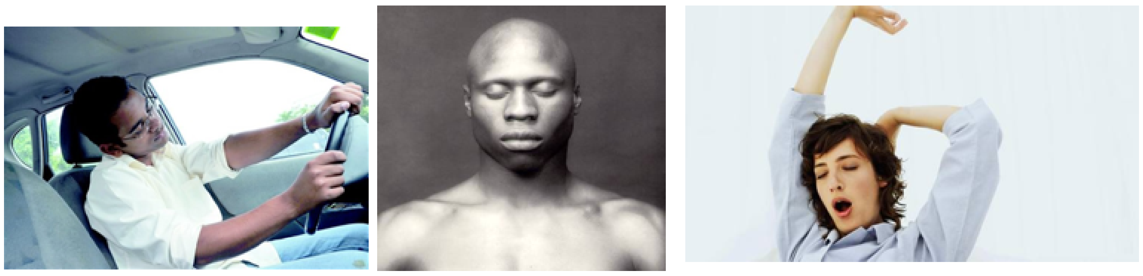


Figure 2.1: *Example of the symptoms inclined head, closed eyes and yawning shown in that order.*

2.2 Machine Learning

Machine learning is a subfield of Artificial Intelligence (AI) which gives computers the ability to improve their experience and make predictions or decisions on data[1, 7]. Generally machine learning is divided into three main variants:

- **Supervised Learning**
  Input data has a known label and results. A model is prepared through a training process in which we give to the algorithm the labels necessary for witch it must be able to make predictions. Example algorithms include Logistic Regression and the Back Propagation Neural Network[8].

- **Unsupervised Learning**
  Input data has no known label and does not have a known result. A model is prepared through a training process in which we don't give to the algorithm the labels necessary for must be able to make predictions. Example algorithms include: the a priori algorithm and k-Means[8].

- **Semi-supervised Learning**
  Input data is a mixture of labeled and unlabeled data. Example algorithms include Logistic Regression and the Back Propagation Neural Network.[8]

Depending on the selected variant, there are many kinds of machine learning algorithms grouped by similarity [8].

The learning process (Figure 2.2), is as follows [1]:

- **Training set** is a set of images, texts, sounds, documents, data or information that could be classified.
- **Feature extraction** involves reducing the information necessary for describing a large set of information to obtain a new vectors set.
- **Labels** depend on the style selected (whether the vectors are labeled or not).
- **Machine learning algorithm** is to receive the input data in a vector form and use statistical analysis to predict an output value within an acceptable range.
- **Predictive model** uses statistics measures to predict results based on a previous training. If we compare new data with the predicted model, this will be capable of determining what it is.

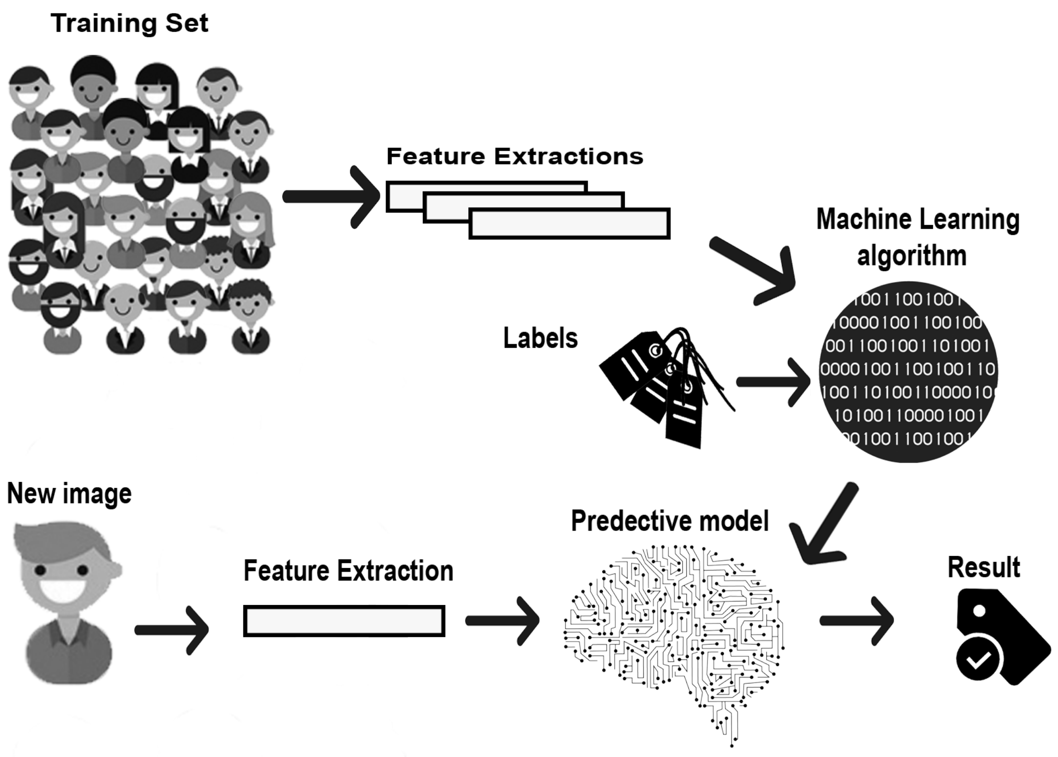


Figure 2.2: *Example of machine learning recognizing faces. [1]*

2.2.1 Deep Learning

Deep Learning is a new focus in machine learning inspired by the architecture and function of artificial neural networks. It promises general, powerful, and fast machine learning, moving us one step closer to AI. Deep learning performance just keeps getting better as you feed it more data compared with older algorithms and making machine learning algorithms much better and even easier to use. Deep learning consists of several levels, which vary depending on the amount of data used for the training (Figure 2.3) [9, 10].

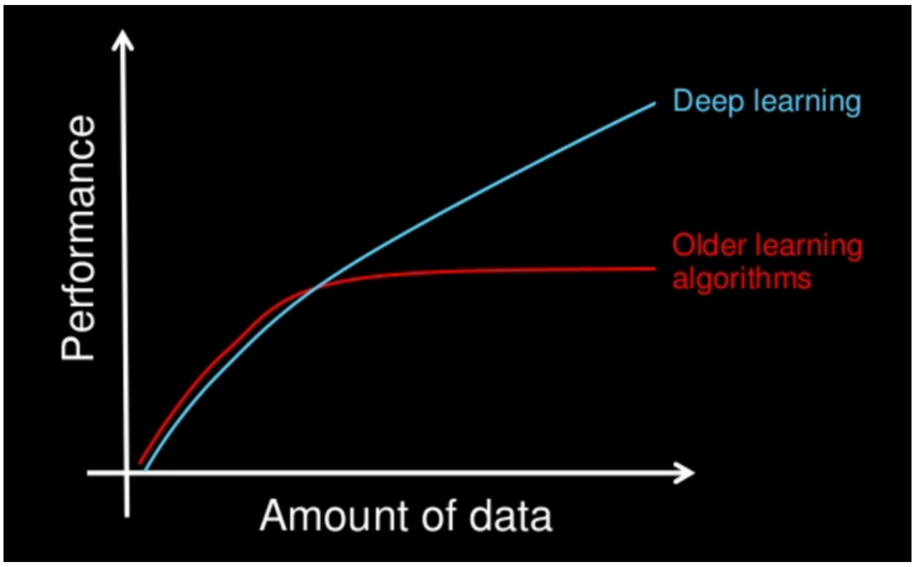


Figure 2.3: *How do data science techniques scale with amount of data? [2]*

Deep learning try to make full use the unknown structure in the input distribution in order to discover good representations. Feature extraction is an important factor for any machine learning process, but is even more critical in deep learning because the process of feature extraction uses high level features (Figure 2.4) [10].
Giving way to new methods of training computers, deep learning needs a lot of information from training data. There are deep pure networks where the feature extraction goes hand in hand with the system training [10, 11]. Models that have been incorporated from a low-level feature to a high-level are also considered deep networks [11, 12] . Deep learning at learning feature hierarchies with high level features formed by the composition of lower level features. This allow to learn complex functions mapping the input to the output directly from data.

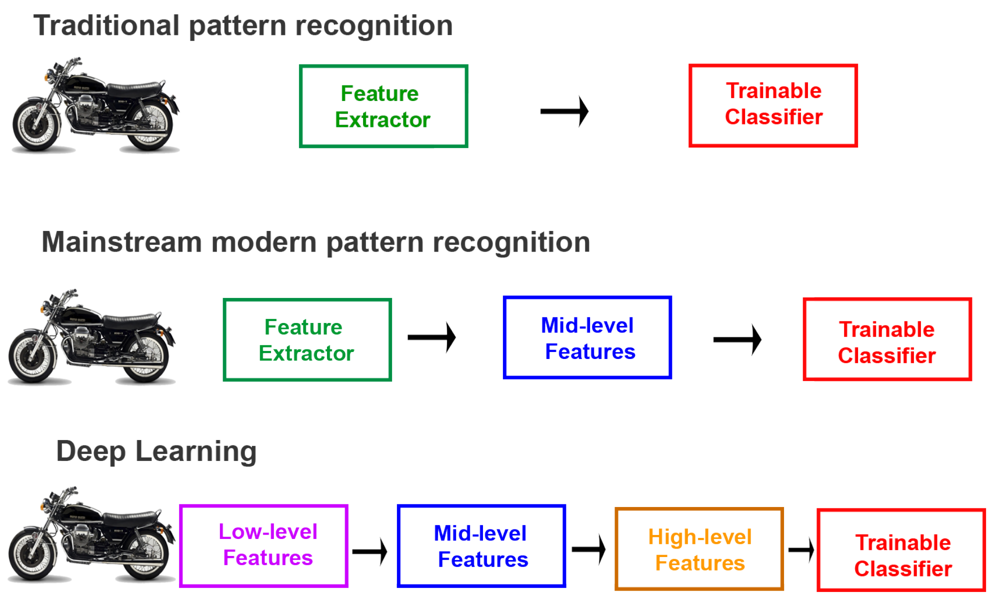


Figure 2.4: *Deep Learning Hierarchical Representations when we shows how in deep learning the feature are in low, mid and high level to works*

2.3 Artificial Neural Networks

Artificial neural networks are inspired by the structure and performance of biological neural networks. These contain layers of simple computing nodes that operate as non-linear summing devices(X). These nodes are interconnected by strong connections lines, and the weights are adjusted when data is presented to the network during the training process (Figure 2.5). Successful training could result in a network that performs tasks such as predictions (Y). Many applications of artificial neural networks have been reported in computer engineering literature, and applications in medicine are growing [13, 14].

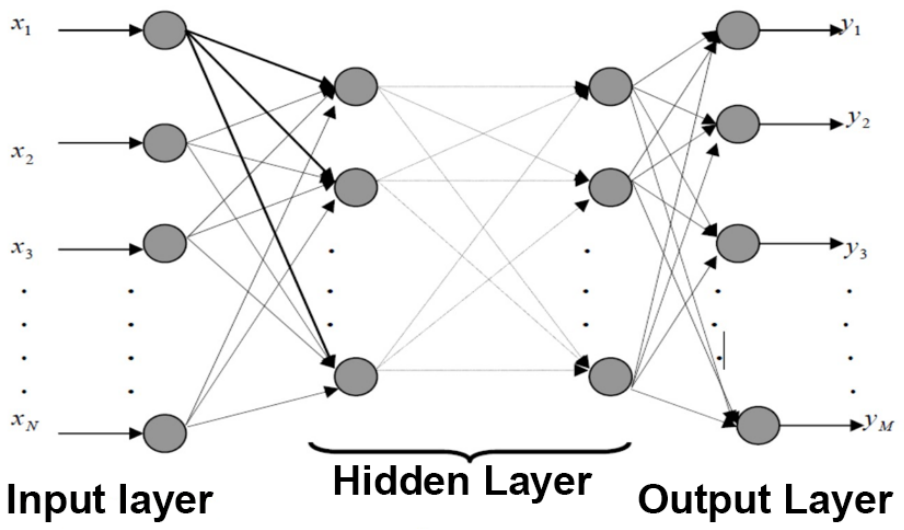
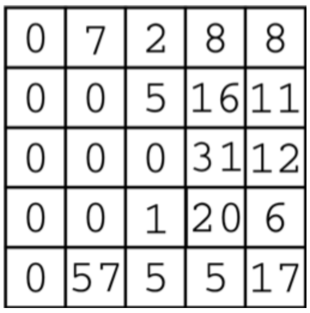


Figure 2.5: *Typical structure of an artificial neural network*

2.3.1 Convolutional Networks

Convolutional Networks are more sophisticated structures used to solve difficult image pattern recognition, with their precise and simple architecture. They offer a simplified method for pattern recognition problems and future extraction. Convolutional neural networks use three basic ideas: local receptive fields, shared weights, and pooling [15, 5].

- **Local receptive fields**. Consider an image of nxn in this case the image is of size of *5 x 5* (Figure2.6).In the algorithm, the operations will be performed in steps within a small region of *x* size. In the example, we use an area in order to obtain a new, smaller matrix (Figure 2.7).



0	7	2	8	8
0	0	5	16	11
0	0	0	31	12
0	0	1	20	6
0	57	5	5	17

Figure 2.6: *Example of the resulting matrix of an image in convolutional neural networks*

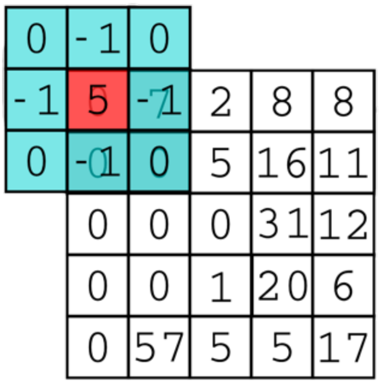


Figure 2.7: *Example for obtaining a value that represents a fragment in the new matrix*

- **Shared weights and biases**. The network structure described so far can detect just a single kind of localized feature. For this process we need more than one feature map [15, 5]. Each feature map is defined by a set of shared weights, and a single shared bias. The result are 3 different kinds of features (Figure 2.8). We show just 3 feature maps in this example. However, in practice, convolutional networks may use more feature maps [15, 5, 16].

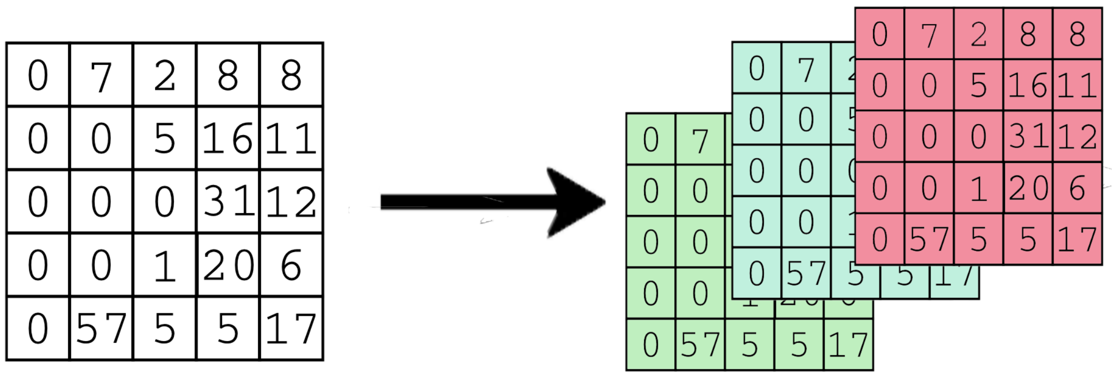


Figure 2.8: *Example of how shared weights are used to obtain three new matrices with different abstraction characteristics.*

- **Pooling layers**. What the pooling layers do is simplify information. This takes each feature map output from the convolutional layer and prepares a condensed feature map. The result is a matrix with half of its size(Figure 2.9). If we put all the steps mentioned together we create a complete convolutional network (Figure 2.10)[15, 5, 16].

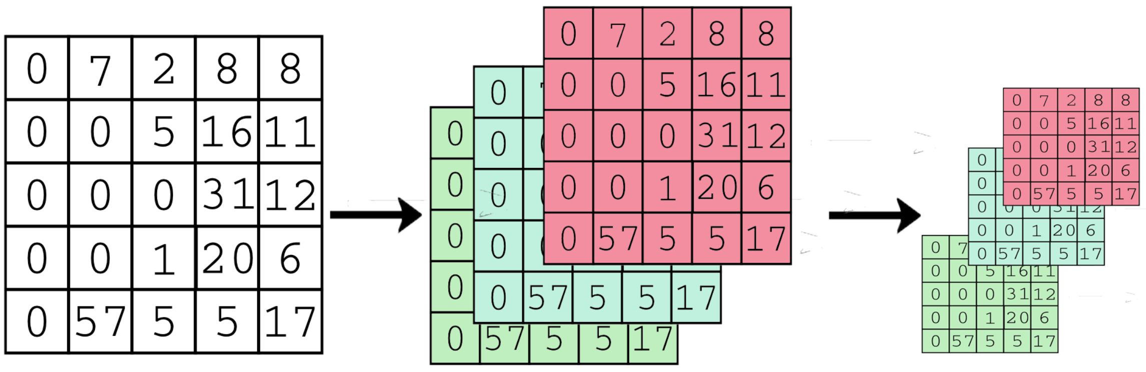


Figure 2.9: *Example for obtaining a value that represents a fragment in the new matrix.*

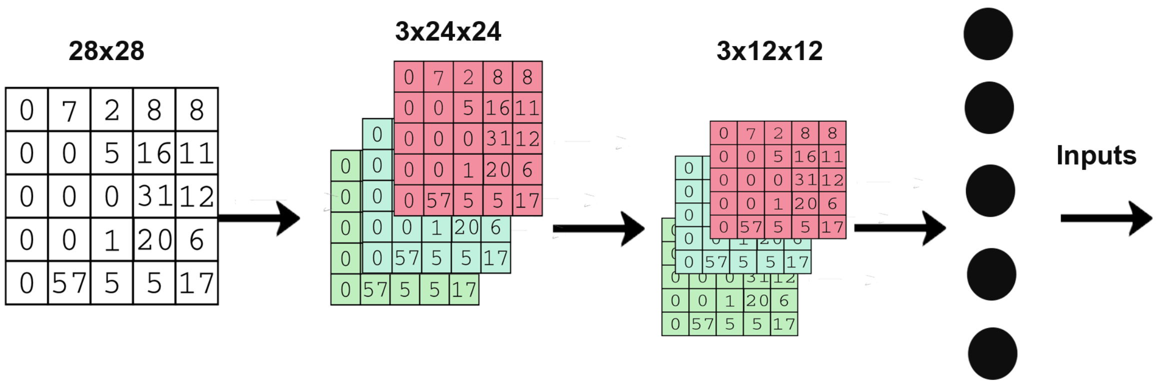


Figure 2.10: *Example of a complete convolutional network.*

2.4 Support Vector Machines

Support Vector Machines is a classifier that learns an optimally separating hyperplane. In other words, the algorithm outputs an optimal hyperplane which categorizes new examples (Figure 2.11) [3].
SVM algorithm works looking for the hyperplane that gives the largest minimum distance

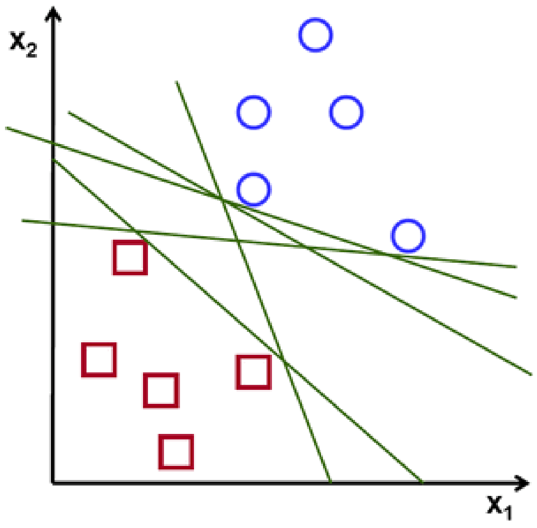


Figure 2.11: *Set of points separated by straight line. It's using support vector machines with different kernel*

to the training examples named Margin. Therefore, the optimal hyperplane maximizes the margin of the training data (Figure 2.12) [3].

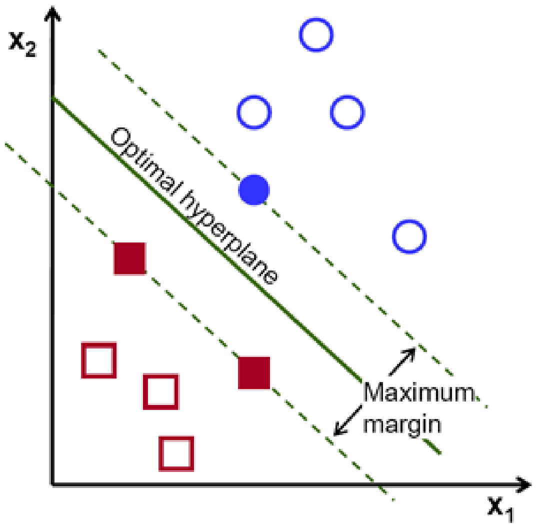


Figure 2.12: *The optimal hyperplane determinate by maximum margin*

The other kernels are: Linear Kernel, Polynomial Kernel, Radial Kernel. This gives to SVM the opportunity to classify with hyperplane more nonlinear even in a circle form (Figure 2.13) [3].

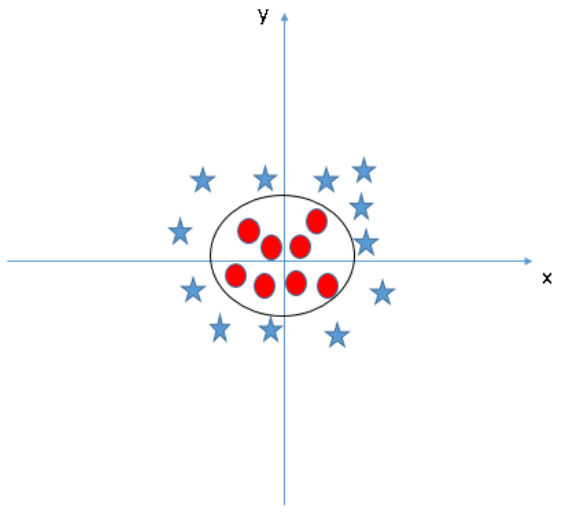


Figure 2.13: *Example of a Polynomial kernel where the red points are correctly classify by the optimal hyperplane in circle form [3].*

2.5 Computer Vision

Computer vision is a subfield of Artificial Intelligence that gives computers the ability to see. It has to take into account the hardware and software limitations. This uses techniques, algorithms and theories to accomplish a visual understanding [17, 18]. Computer vision is made up of different elements such as image pre-processing, feature detection and description, image registration, Depth calculation, object recognition, motion analysis and heuristics [17, 18].
The explanation of these elements are the following:

- **Image pre-processing** includes tasks such as noise reduction, normalizing color and gamma values among others.
- **Feature detection and description** identifies important regions in the image that can be shrunk instead of the full-sized image, reducing computational complexity.
- **Image registration** aligns multiple images for simplifying pixels comparisons
- **Depth calculation** improves the performance of algorithms that require an understanding of three-dimensional space such as 3D model reconstruction.
- **Object recognition** identifies groups of pixels or features that could represent objects.
- **Motion analysis** extracts from multiple frames of video in order to predict the trajectory of objects.
- **Heuristics** facilitate decision-making with previous information in a split-second.

2.5.1 Feature Extraction

Feature extraction applies functions that look for attributes that represent and describe the object of interest. It is the first step for pattern recognition and could be used to assign the object a determined class[19, 20].
Feature detection transforms an image from a large set of pixels into a reduced representation as a feature vector (Figure 2.14) [19, 20, 21].

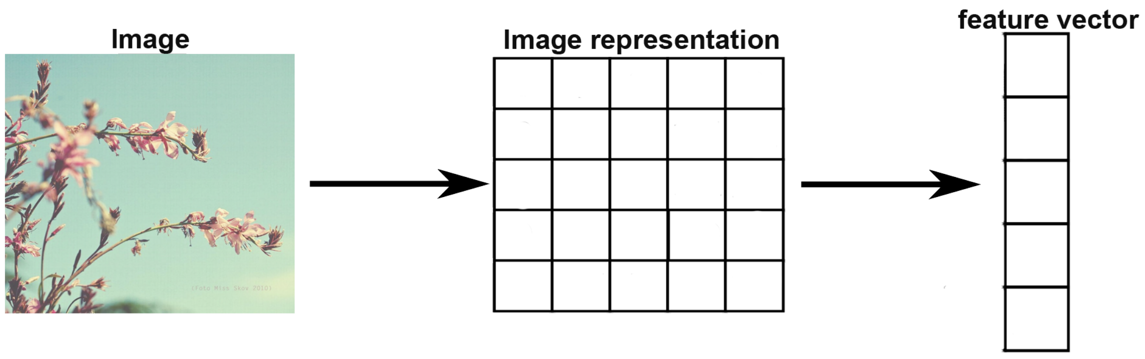


Figure 2.14: *Typical computer vision processing.*

Face detection is growing deeper and deeper into the feature extraction area. Many algorithms exist to detect faces but haar-like features has been highlighted for its velocity and efficiency in face image recognition[19, 20].

2.5.1.1 Haar-like features

Haar-like features are rectangular regions that are placed in a specific location. Next, these features sums up the pixel intensities in each region and calculate the difference between them. The difference is then used to categorize subsections of an image by looking for something specific. It is important to remember that this algorithm works by appearance [4, 21]. In human faces, commonly, the eye area is darker than the areas on the cheeks. Haar-like features for face detection work in rectangular areas above and between the eye and cheek regions [4].

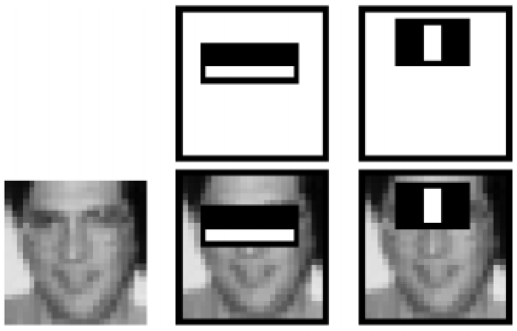


Figure 2.15: *Example of Haar-like Features areas [4].*

2.6 Literature Review

A literature review is the result of looking for scholarly articles relevant to a particular issue, area of research, or theory, and provides a description, summary, and critical evaluation of these works in relation to the research problem that we are approaching.

In 2002, Singh Sarbjit and Papanikolopoulos Nikolaosporpose [22] detected people with fatigue by finding and tracking the location of the pupil. They used a gray scale model to determine whether the eye was open or closed. The result was a model that apparently works well. However, there were some problems. Under these circumstances, the system was able to detect prolonged eye blinks 95% of the time, and it produced occasional false alarms.

In 2007, Xiao Fan, Bao-Cai Yin and Yan-Feng [23] Sun proposed to locate and track a driver's mouth movement using a camera to detect people yawning. They used Gabor for feature extraction and Latent Dirichlet Allocation for training the model. They obtained an average recognition rate of 95%.

In 2008, Saradadevi Mandalapu and Bajaj Preeti [24] proposed a method to locate and track drivers' mouths using Viola-Jones and the Support Vector Machines. They had as a result 81% accuracy.

In 2009, Hu Shu Han and Zheng Gangtie [25] tried to perform a model to detect drowsiness. It employed the Support Vector Machines. They got 86.67% accuracy on average during the trials to detect when the subject was 'sleepy'.

In 2011, Zhao Chunlin, Zheng Chongxun, Zhao Min, Tu Yaling and Liu Jianping [26] explored the possibility to detect fatigue using electroencephalographic signals (EEG) as data and the Support Vector Machines for training. They obtained 81.64% of accuracy in their investigation.

In 2011, Coetzer Reinier C. and Hancke Gerhard P. [27] developed a monitoring system, centered around the tracking of drivers' eyes. It was using the following techniques: Artificial Neural Networks (ANN), Support Vector Machines (SVM) and Adaptive Boosting (AdaBoost). They obtained 95.5% accuracy with SVM, 97.7% accuracy with AdaBoost and 96.1% accuracy in their results.

In 2014, Jiao Yingying, Peng Yong, Lu Bao-Liangc, Chen Xiaoping, Chen Shanguang and Wang Chunhui [28] explored a way to detect fatigue using Slow Eye Movement (SEM) data.It used the Support Vector Machines and the Graph Regularized Extreme Learning Machine for training the models. They had as a result of 93% accuracy on average.

Table 2.1 shows a comparison between the approaches presented in this review of the state of the art. From this Table, we can observe that the highest accuracy was obtained by Coetzer, with 97.7%.

Year	Principal author	Method	Acurracy
2002	Singh Sarbjit	RGB Detector	95%
2007	Xiao Fan	Gabor & LDA	95%
2008	Saradadevi Mandalapu	Viola Jones & SVMx	81%
2009	Hu Shu	SVM	86.67%
2011	Zhao Chunlin	EEG	81.64%
2011	Coetzer Reinier C.	ANN, SVM & Adaboost	97.7%
2014	Jiao Yingying	SEM	93%

Table 2.1: *Comparative table of related works.*

In the literature, we found as a limitations of our work the followings: build of dataset , selection of machine learning algorithm. The method proposed is trying to avoid the limitations presented by literature.

III

Methodology

Chapter 3: Methodology

In this chapter we describe the rationale for the application of specific procedures and techniques used to identify, select, and analyze information applied to understanding the research problem.

3.1 Introduction

The methodology used in this work is divided into two parts. The first one has to do with it predicted models, that consist to obtain a dataset, a pre-trained network, the feature extraction of the images, experiments with *weka* and finally the training. And, the second part that consist in specifications and requirements to implement the prototype as connect the camera with the software and the programming part (figure 3.1).

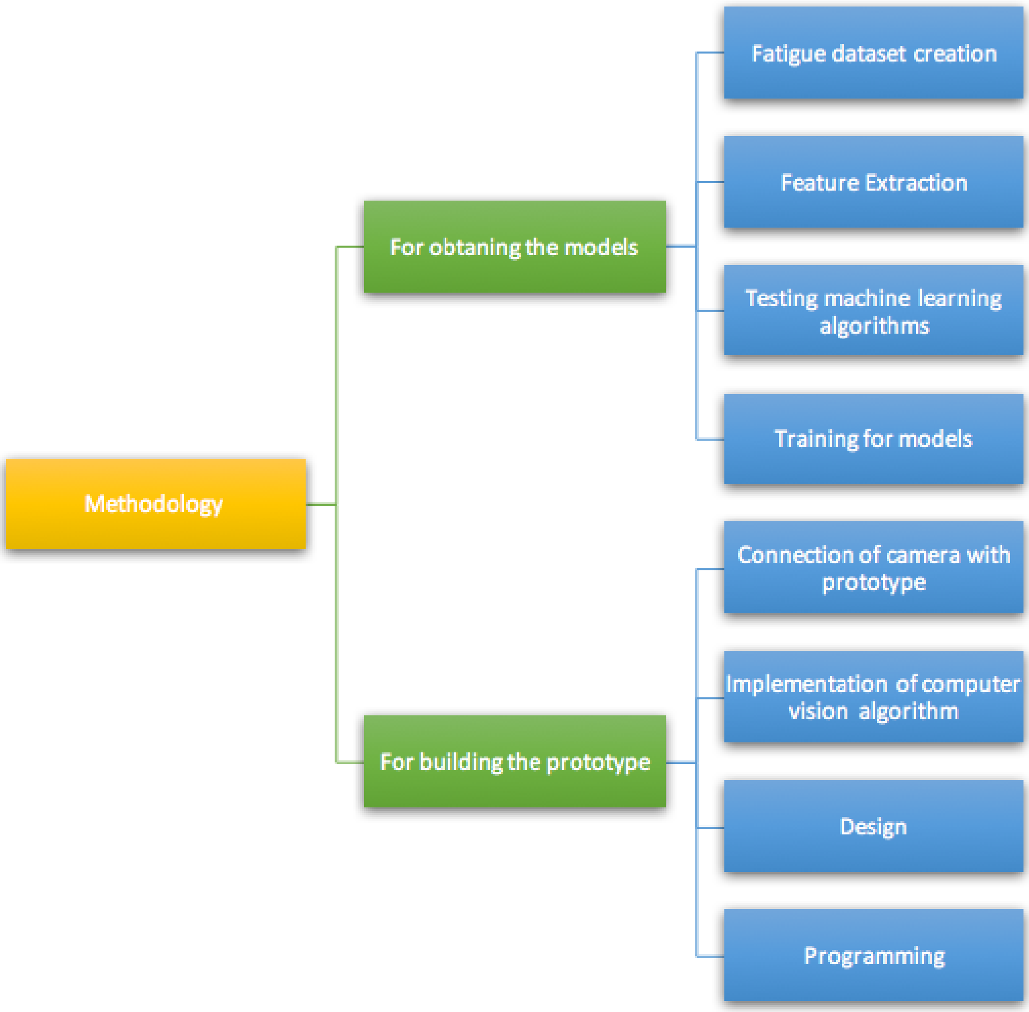

Methodology
For obtaning the models
Fatigue dataset creation
Feature Extraction
Testing machine learning algorithms
Training for models
For building the prototype
Connection of camera with prototype
Implementation of computer vision algorithm
Design
Programming

Figure 3.1: *Diagram that represent both methodologies used in this work.*

3.2 Methodology for obtaining the models

3.2.1 Fatigue dataset creation

The first stage consists on building a dataset of images for the approached problem. To build a database of people expressing fatigue, a web searching was performed from different websites such as *we heart it, Pinterest, Photobucket and Google* [29]. From these images, we manually selected the best candidates. Finally, it is worth to mention that because these images are taken from the web, they are varied in the position of the person, their age, lighting, distance, sharpness and other factors that make this dataset more complete.

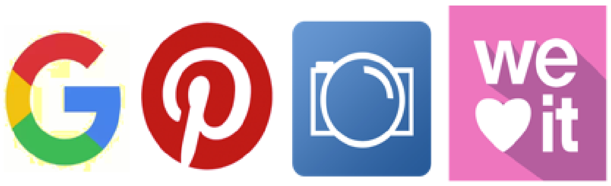


Figure 3.2: *Logotypes of the pages used for the research images*

3.2.2 Feature Extraction

The annotated images are labeled with their specific expression. The network architecture we used was a deep neural network from MatConvNet [30]. MatConvNet is a MATLAB toolbox useful for implementing Convolutional Neural Networks (CNNs) for computer vision applications. It is easy to use, efficient, simple and available for use in different projects. The network architecture we used in these experiments was Vgg-Face.

Vgg-Face consists of sixteen convolutional layers, some of which are followed by haar-like for preprocessing images looking for faces in it, next continue with three max-pooling layers, and six fully- connected layers. The network components are pre-trained with the well-labeled data of famous people. (Figure 3.3).

For the experiment we eliminate the last layer (the label layer) of the convolutional network, in order to obtain the high-level features of the images. Finally, we copied the vectors obtained from the convolutional network to create a file that contains all the vectors of each category (inclined head, yawning and closed eyes).

3.2.3 Testing machine learning algorithms

In this stage we test several machine learning algorithms with the goal of determine how the dataset works. This stage was carry out using *Weka*[1].

First we created the *csv* file necessary to work in *weka* where we put in the file the feature extraction obtained of last step. Next, we put their corresponding label in the

[1] "Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes"[31]

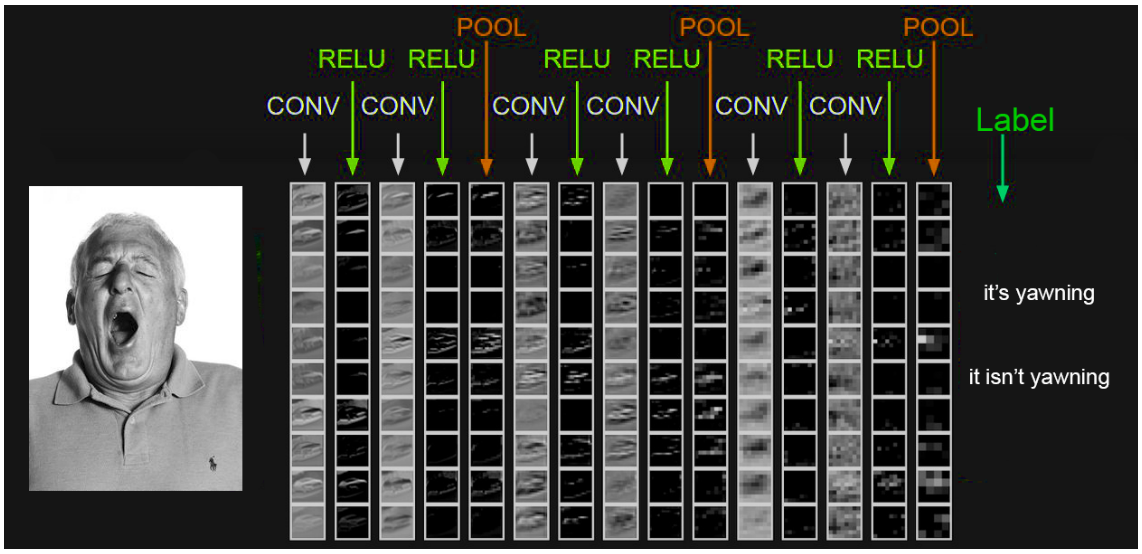


Figure 3.3: *Example of Vgg-Face architecture [5].*

finally part in the file.

Next, we tested the following algorithms: naive Bayes, support vector machines, random forest and liblinear. These algorithms were selected from a previous experiment, where all the algorithms in *Weka* were tested.

3.2.4 Training of models

For training, ten percent of the images was selected as a test set and the rest was used as training and validation sets. All models were evaluated against a well-labeled test set. The training was done in *MATLAB*[2] following the methodology presented in (Figure 3.4); and used SVM functions and the default parameters with linear kernel and 10-fold cross-validation. Finally, we obtained the three models: one for yawning, one for closed eyes and one for inclined head.

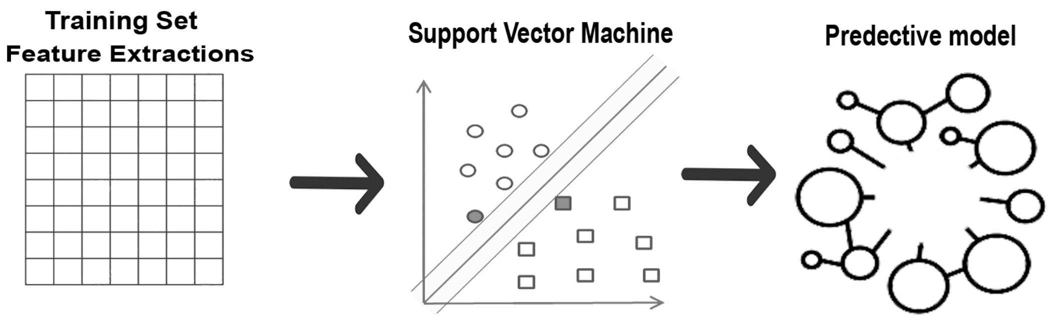


Figure 3.4: *Supervised learning process used.*

[2]"MATLAB is a software Self-styled as multi-paradigm numerical computing environment and fourth-generation programming language. A proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces"[32].

3.3 Methodology for building the prototype

The prototype development was done in Matab, because in this tool it was implemented the pre-trained network, which is necessary for feature extraction. The stages for the development of the prototype are: connection of camera with prototype, implementation of computer vision algorithm, design and programming.

3.3.1 Connect camera with prototype

In order to establish the connection of the camera with the prototype, it was necessary to install different tools and drivers in *Matab*(Figure 3.5). These, we developed some code to select a camera, in the case that several cameras are available in the computer, and installing the corresponding driver.

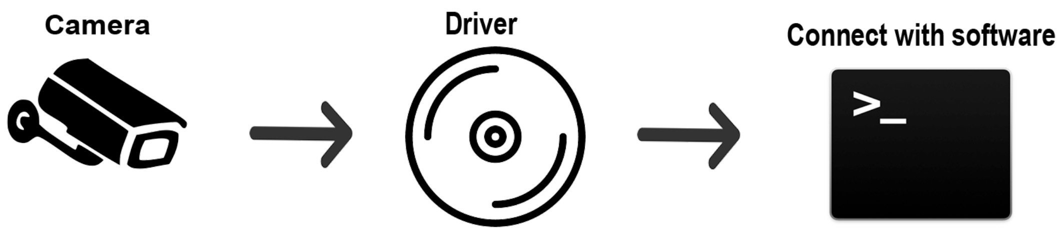


Figure 3.5: *Process used to connect the camera to the software.*

3.3.2 Implementation of computer vision algorithm

In order to implement a computer vision algorithm in the prototype, it was necessary to combine the different functions obtained from previous steps such as: connection of the camera, extraction of features and models. The process is the following: First we connected camera with MATLAB, next from camera we extract a image(frame), this image continue with feature extraction to obtain the feature vector and finally, this vector is proceeded in order to obtain a result(Figure 3.6).

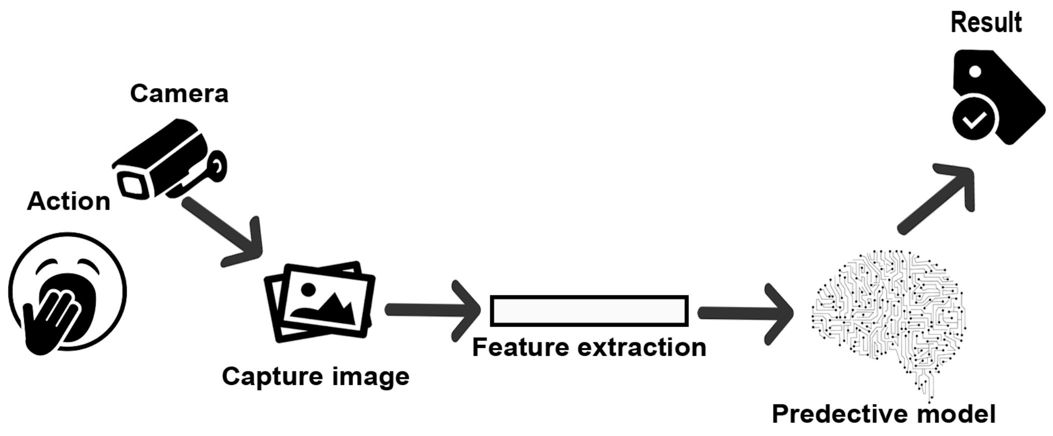


Figure 3.6: *Process used of computer vision algorithm.*

3.3.3 Design

In order to meet the system requirements, a driver survey was conducted. We did questions to determine and selected requirements of the design for the prototype. And help us to know that the factors that we selected previously were the correct one. Next, with the information obtained, we could did a analysis and a purpose of a design for the prototype.

IV

Results

Chapter 4: Experimental Results

In this chapter we report the findings of our study based upon the methodologies to obtain the prototype.

4.1 Models performance

4.1.1 Dataset

The dataset that we created contains 98 images of people yawning, 90 images of people with closed eyes, 96 images of people with inclined heads and 100 images of people with different expressions. The latter set of images was used as negative images (Table 4.1)The same did it for the images used for testing. Figure 4.1 shows examples of images that contains the dataset obtained from web pages. We want to highlight that the system was tested with a different data set of images that we used for training. Examples of these images are shown in figures 4.6, 4.7,4.8,4.9 and 4.10.

Expression of fatigue	Number of images
Yawning	98
Inclined head	90
Closed eyes	96
Random images	100

Table 4.1: *Number of annotated images in each category.*

Expression of fatigue	Number of images
Yawning	50
Inclined head	50
Closed eyes	25
Random images	50

Table 4.2: *Number of annotated images in each category used for testing.*

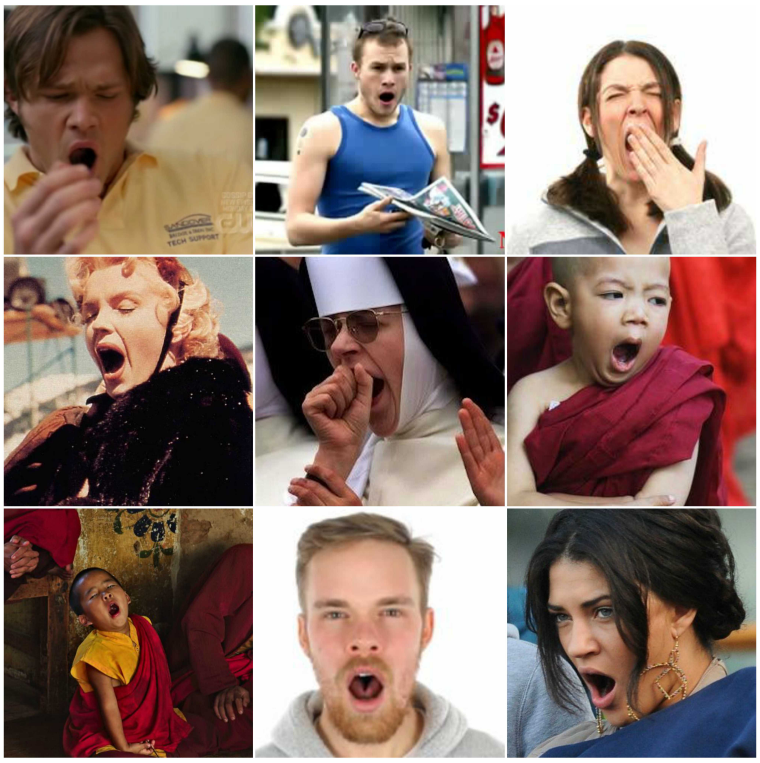


Figure 4.1: *Examples of images that conform the yawning dataset*

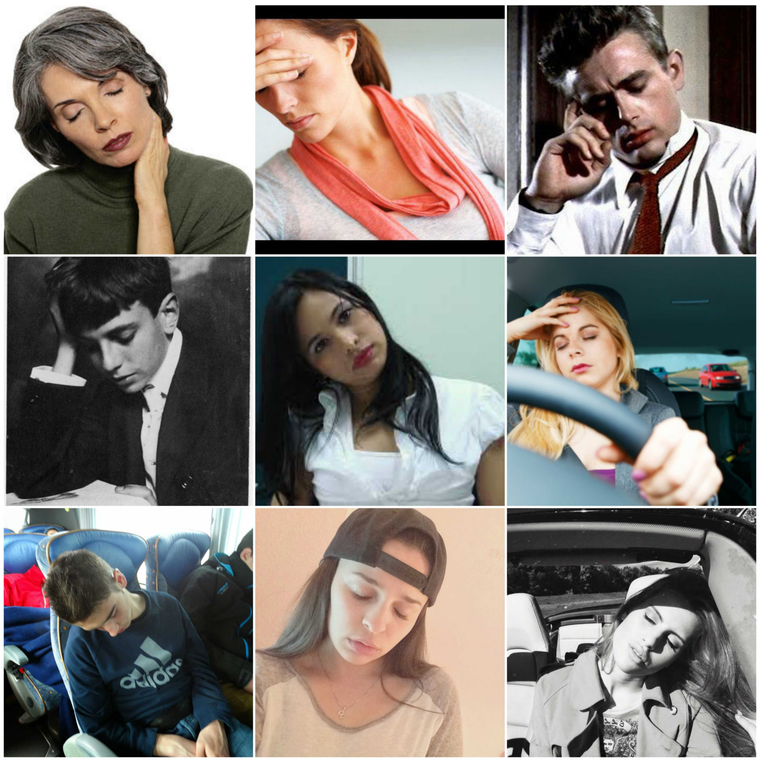


Figure 4.2: *Examples of images that conform the inclined head dataset*

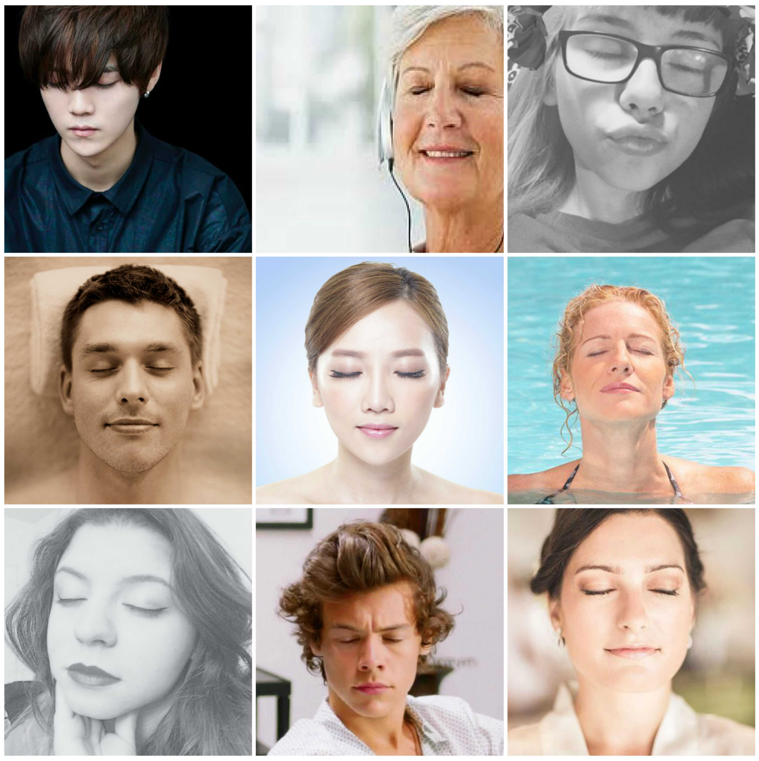


Figure 4.3: *Examples of images that conform the closed eyes dataset*

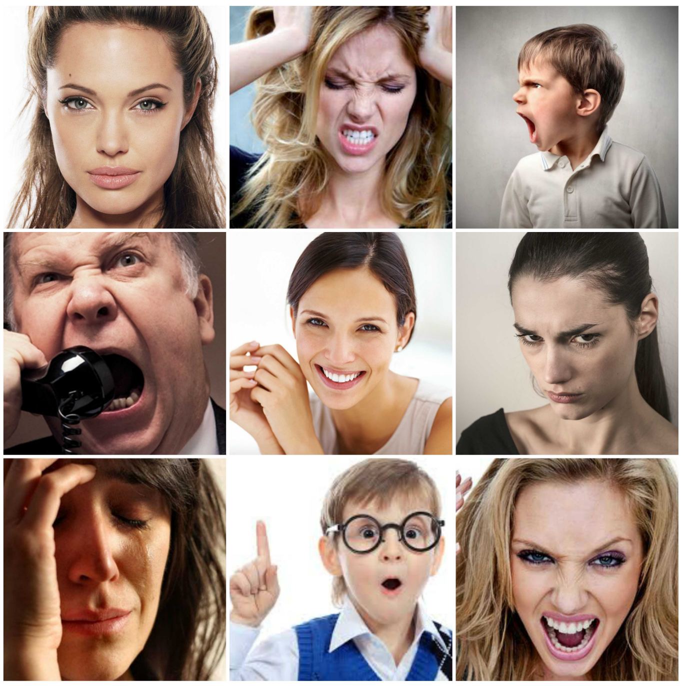


Figure 4.4: *Examples of images that conform the negative dataset*

4.1.2 Results

The experimentation in *weka*, was made with the following algorithms: lib-linear, support vector machine, random forest and Naive Bayes.Tables 4.3, 4.4 and 4.5 show the accuracies for yawning, closed eyes and inclined head, respectively. As we can observe from these tables, the best average accuracy for yawning is 83.56% with SVM; for closed eyes is 76.85% using RF; while for inclined head is 88.0% with Liblinear. However, SVM obtained the best accuracy on average for the three symptoms, with 82.05%, as we can see in Table4.5.

Seed	NB	SV	RF	LL
1	81.02	84.22	80.33	82.91
5	83.10	83.12	80.33	83.04
10	83.27	83.20	79.99	83.21
15	83.05	83.36	79.48	82.22
20	82.34	83.87	81.21	82.65
Average	82.55	83.56	80.27	82.81

Table 4.3: *Accuracy for yawning using naive Bayes (NB), support vector machines (SVM), random forest (RF) and liblinear (LL).*

Seed	NB	SV	RF	LL
1	51.48	74.57	76.99	74.94
5	51.36	78.29	76.37	77.96
10	52.81	75.50	77.09	75.91
15	51.91	76.65	76.54	75.74
20	51.79	75.74	77.23	76.02
Average	51.87	75.66	76.85	76.12

Table 4.4: *Accuracy for closed eyes using naive Bayes (NB), support vector machines (SVM), random forest (RF) and liblinear (LL).*

Seed	NB	SV	RF	LL
1	85.80	86.69	82.23	87.91
5	85.16	86.60	82.93	86.65
10	85.14	87.10	84.76	87.76
15	86.11	87.05	82.85	88.95
20	86.27	87.24	84.06	88.72
Average	85.70	86.94	83.37	88.0

Table 4.5: *Accuracy for inclined head using naive Bayes (NB), support vector machines (SVM), random forest (RF) and liblinear (LL).*

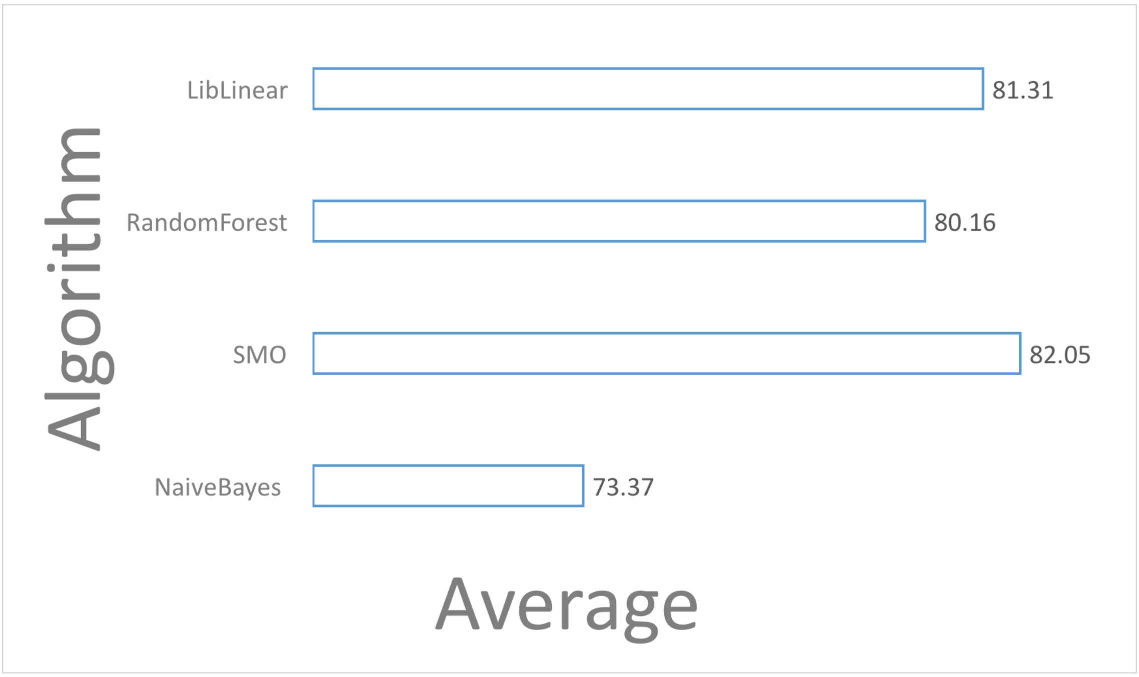


Figure 4.5: *Average accuracy for the three symptoms of fatigue: yawning, closed eyes and inclined head.*

Next,we trained a model with support vector machine implemented in matlab. The results from this training are shown in table where present the average obtained with Cross- validation loss that it is return to us loss obtained by cross-validated in the model. For every fold, this method computes classification loss for in-fold observations using a model trained on out-of-fold observations.t 4.6.

Symptom	Average	Cross-validation loss	F-Measure
Yawning	99.50%	0.2073	1
Inclined head	100%	0.0798	1
Closed eyes	100%	0.2526	1

Table 4.6: *Accuracy obtained for training the models.*

To assess the effectiveness of our proposed deep learning model for fatigue factors, we conducted extensive experiments evaluating the performance of the models. In these experiments, we used different images from the ones we used for training obtained in same form than we obtained the dataset . In Figure 4.6, we show images that were correctly classified as “not yawning”. While in Figure 4.7, we show images that were correctly classified as “yawning”.

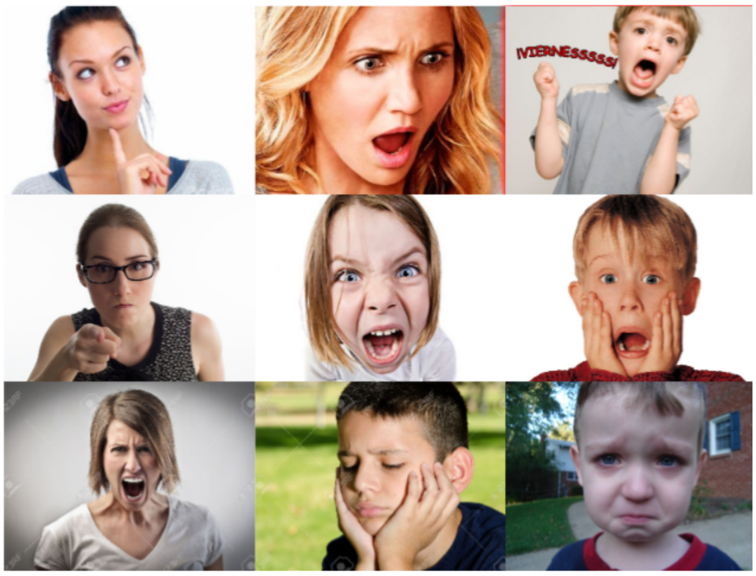


Figure 4.6: *Examples of images for testing the model with persons that are not yawning*

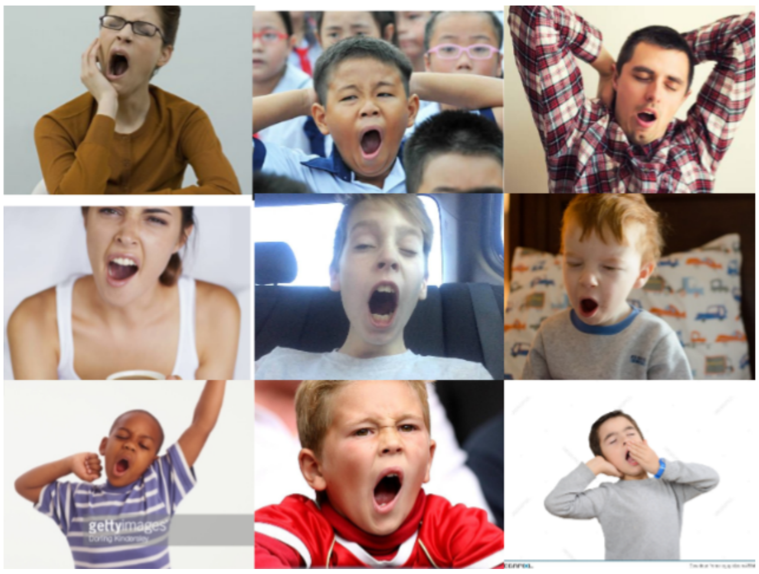


Figure 4.7: *Examples of images for testing the model with persons that are yawning.*

But in this test not everything was perfect. Both images shown in Figure 4.8 are images that give us a true negative result. The image on the left side is a person that is yawning and on the other side is an image of someone screaming.

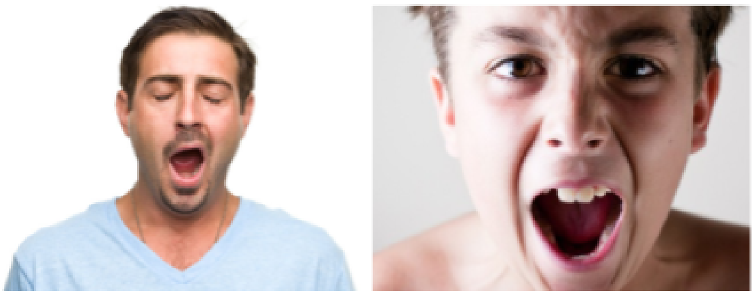


Figure 4.8: *Image for testing that give us a false-positive result*

For detecting closed eyes, we had the following results. All the images we tested gave us correct results. But if we looked for an image of a person with half-closed eyes, the system detected it as closed eyes (Figure 4.9).

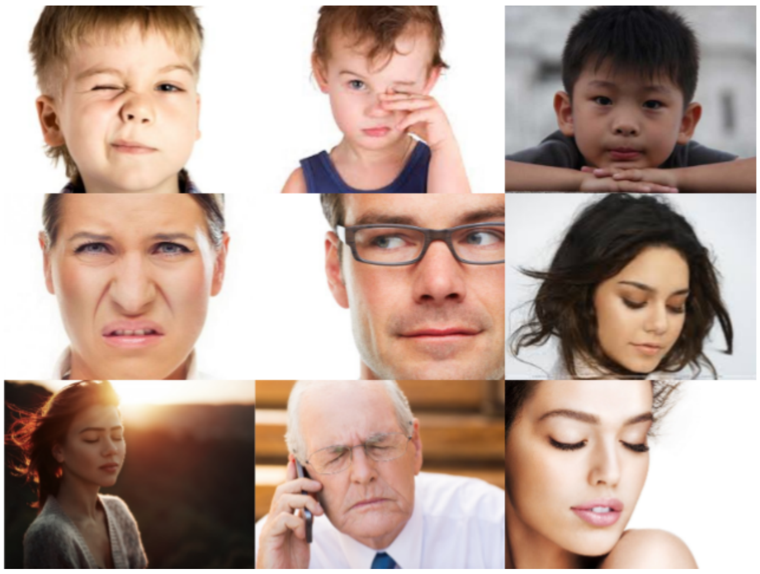


Figure 4.9: *Examples of images for testing the model with closed eyes*

For detecting an inclined head, these were the results of the testing: all the images tested gave us correct results. However, if you look for images of people looking up, it also gives a positive result (Figure 4.10).

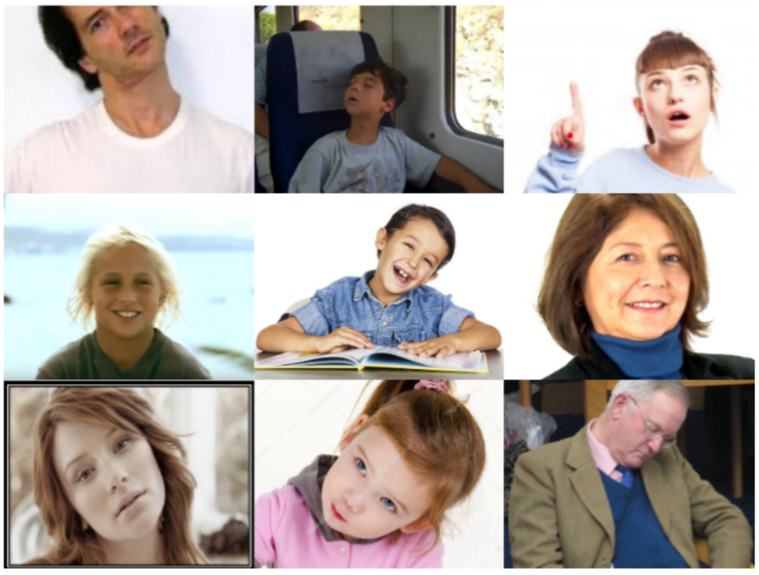


Figure 4.10: *Examples of images for testing the model with inclined head*

4.2 Evaluation of the prototype

4.2.1 Connect camera with software and Adapt vision computer algorithm

In order to build the system for fatigue detection, we first connect a camera with the prototype. Then, we use the well-known haar-like feature algorithm to face recognition. In (Figure4.11), we show an example of this task.

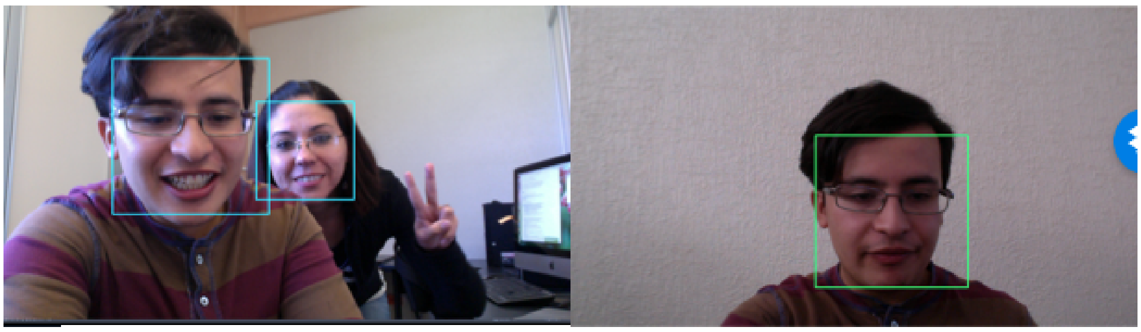
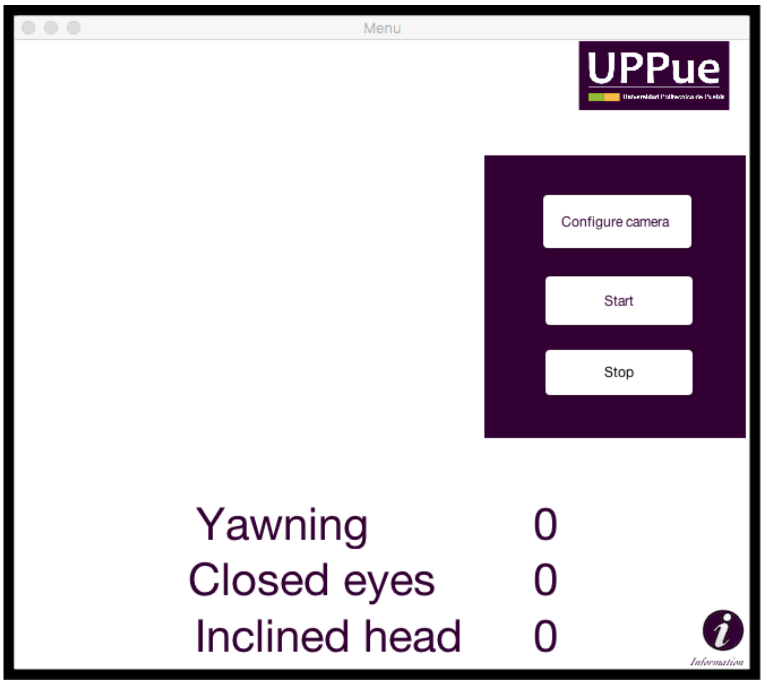


Figure 4.11: *Results for face recognition to build the prototype.*

4.2.2 Development of the prototype

For building the system, we implemented the pre-trained network models in the prototype. Thus, the input images from the camera are compared with the different models to detect any of the three fatigue symptoms: yawning, closed eyes and inclined head.

Once we verified that the system perform as we expected, we built the interface to manage some functionalities of the prototype (Figure 4.12).



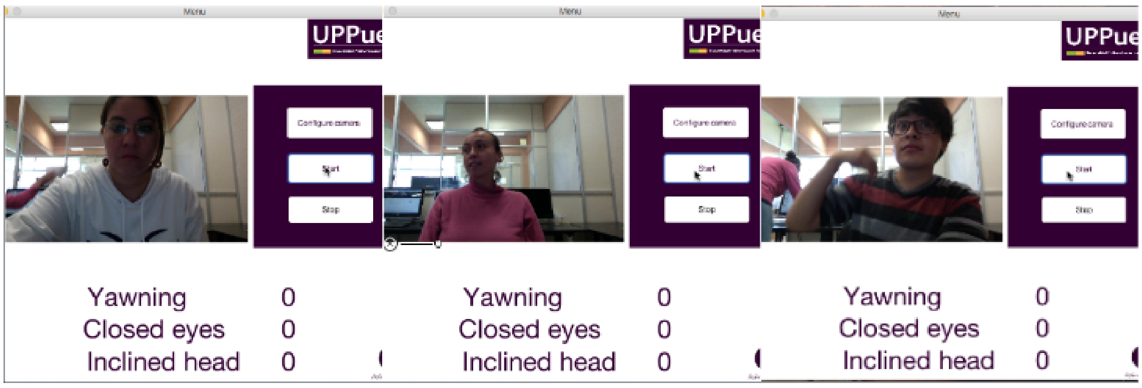
Yawning 0
Closed eyes 0
Inclined head 0

Figure 4.12: *The interface of the prototype*

The three functionalities of the system are the following:

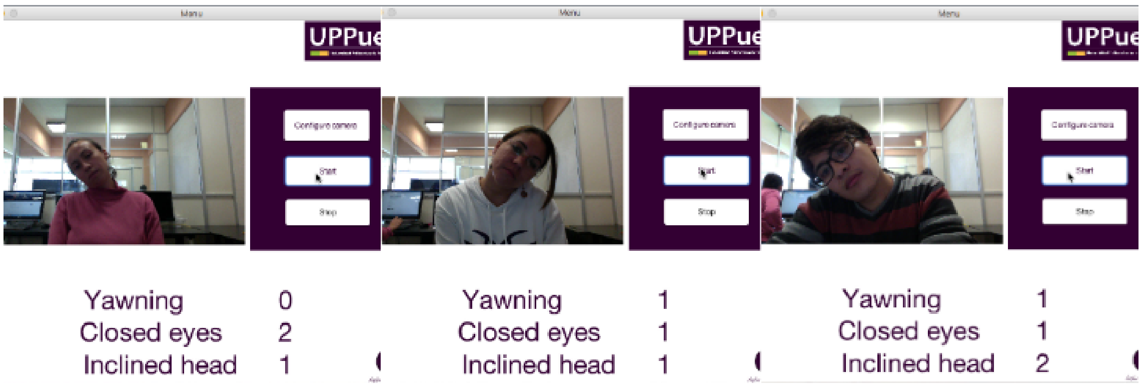
- **Configure camera** allows one to select the camera of any computer, camera driver and operating system. The system shows all the cameras that you have in your computer and format. The user just needs to select the operating system indicated.
- **Start** charges all the fatigue models and pre-trained networks for feature extraction. It starts the web cam function and starts to capture the images from the camera.
- **Stop** ends all functions and turns off the program.

When the system detects one of the fatigue factors, it counts the number of its respective label and, at the same time, shows an image to support the system's decision. Figures 4.13, 4.14, 4.15 and 4.16 show different examples of of the performance of the system for detecting the symptoms of fatigue. As we can observe, the system is able to detect yawning, inclined head and closed eyes.



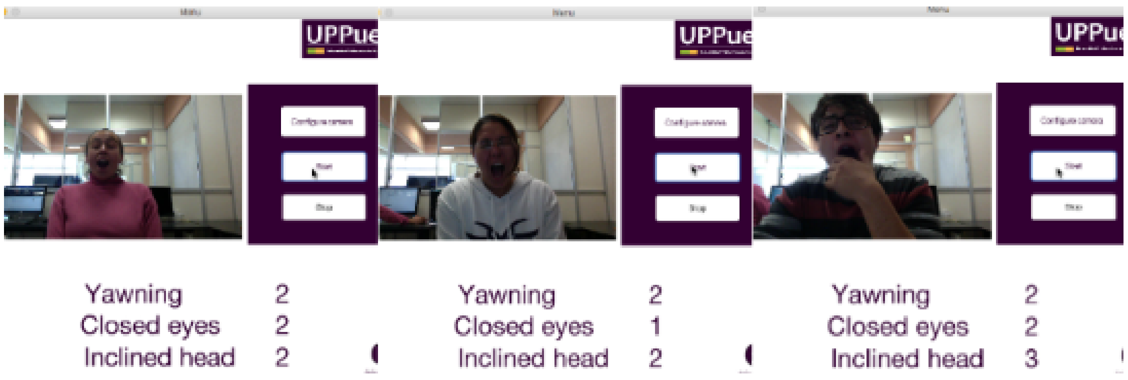
Yawning	0	Yawning	0	Yawning	0
Closed eyes	0	Closed eyes	0	Closed eyes	0
Inclined head	0	Inclined head	0	Inclined head	0

Figure 4.13: *Examples when the system not detects any symptom.*



Yawning	0	Yawning	1	Yawning	1
Closed eyes	2	Closed eyes	1	Closed eyes	1
Inclined head	1	Inclined head	1	Inclined head	2

Figure 4.14: *Examples when the system detects inclined head.*

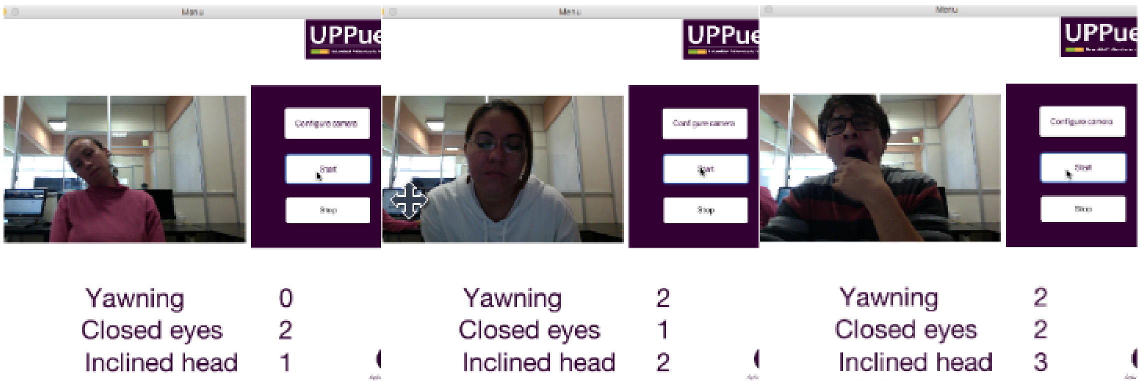


Yawning 2
Closed eyes 2
Inclined head 2

Yawning 2
Closed eyes 1
Inclined head 2

Yawning 2
Closed eyes 2
Inclined head 3

Figure 4.15: *Examples when the system detects yawning*



Yawning 0
Closed eyes 2
Inclined head 1

Yawning 2
Closed eyes 1
Inclined head 2

Yawning 2
Closed eyes 2
Inclined head 3

Figure 4.16: *Examples when the system detects closed eyes.*

V

Conclusions

Chapter 5: Conclusions

5.1 Conclusions

We have introduced a prototype for detecting fatigue using machine learning and computer vision. Particularly, we have used deep learning to extract relevant features to detect the three main factor of fatigue: yawning, closed eyes and inclined head. We used a pre-trained convolutional neural network from MatConvNet to extract the features. Experimental results indicate that support vector machines obtains the best results to detect the fatigue factors, this was implemented in the prototype. The image dataset was obtained through search engines from several web sites. Finally, the prototype was built in Matlab, programming an interface to manage three functionalities.

Also, we can mention that looking for images on the web, is a good way to create a complete database without a review from experts. In addition, we can highlight that deep learning permit to extract high level features to obtain good results, which are similar to the reported int the literature. Finally, these models could be used to prevent fatigue in many industries or in the field of software engineering.

5.2 Future work

From the results presented in this work, several topics can be researched, some could include:

- Using and end to end learning scheme.
- Training, using as negative images that ones that give us positive-negative results.
- Parallelize the feature extraction part.

Bibliography

[1] T. M. Mitchell, *Machine Learning*, 1997.

[2] A. Y.-T. Ng. (2015, Nov.) What data scientist should know about deep learning? [Online]. Available: https://www.slideshare.net/ExtractConf

[3] S. R. Gunn *et al.*, “Support vector machines for classification and regression,” *ISIS technical report*, vol. 14, pp. 85–86, 1998.

[4] S. Soo, “Object detection using haar-cascade classifier,” *Institute of Computer Science, University of Tartu*, 2014.

[5] J. Johnson and A. Karpathy. Cs231n: Convolutional neural networks for visual recognition. [Online]. Available: http://cs231n.github.io/convolutional-networks/

[6] F. CEA, *El sueño y la fatiga en la conducción: ¿Cuáles son los hábitos de los conductores españoles?*, Jul. 2015, no. 1.

[7] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” vol. 349, no. 6245, pp. 255–260, Jul. 2015.

[8] J. Brownlee. (2013, Nov.) A tour of machine learning algorithms. [Online]. Available: http://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

[9] A. V. K. Chatfield, K. Simonyan and A. Zisserman, “Return of the devil in the details: Delving deep into convolutional nets,” *CoRR*, vol. abs/1405.3531, 2014. [Online]. Available: http://arxiv.org/abs/1405.3531

[10] Y. B. Y. LeCun and G. Geoffrey, “Deep learning,” *Nature*, vol. 521, 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14539

[11] J. Brownlee. (2016, Aug.) What is deep learning? [Online]. Available: http://machinelearningmastery.com/what-is-deep-learning/

[12] Y. Liang, M. L. Reyes, and J. D. Lee, “Real-time detection of driver cognitive distraction using support vector machines,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 340–350, June 2007.

[13] X. B. Olabe, “Redes neuronales artificiales y sus aplicaciones,” *Publicaciones de la Escuela de Ingenieros*, 1998.

[14] H. White, *Artificial Neural Networks: Approximation and Learning Theory*. Cambridge, MA, USA: Blackwell Publishers, Inc., 1992.

[15] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[16] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, “A convolutional neural network for modelling sentences,” *arXiv preprint arXiv:1404.2188*, 2014.

[17] L. E. Sucar and G. Gómez, "Visión computacional," *Instituto Nacional de Astrofísica, Optica y Electrónica. Puebla, México*, 2011.

[18] R. J. Schalkoff, *Digital image processing and computer vision*. Wiley New York, 1989, vol. 286.

[19] M. L. Guevara, J. D. Echeverry, and W. A. Urueña, "Detección de rostros en imágenes digitales usando clasificadores en cascada," *Scientia et technica*, vol. 1, no. 38, 2008.

[20] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, 2010.

[21] A. Aronoff. Deep dive: Implementing computer vision with powervr. [Online]. Available: https://www.imgtec.com/blog/deep-dive-implementing-computer-vision-with-powervr/

[22] S. Singh and N. P. Papanikolopoulos, "Monitoring driver fatigue using facial analysis techniques," in *Intelligent Transportation Systems, 1999. Proceedings. 1999 IEEE/IEEJ/JSAI International Conference on*. IEEE, 1999, pp. 314–318.

[23] X. Fan, B.-C. Yin, and Y.-F. Sun, "Yawning detection for monitoring driver fatigue," in *Machine Learning and Cybernetics, 2007 International Conference on*, vol. 2. IEEE, 2007, pp. 664–668.

[24] M. Saradadevi and P. Bajaj, "Driver fatigue detection using mouth and yawning analysis," *International journal of Computer science and network security*, vol. 8, no. 6, pp. 183–188, 2008.

[25] S. Hu and G. Zheng, "Driver drowsiness detection with eyelid related parameters by support vector machine," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7651–7658, 2009.

[26] C. Zhao, C. Zheng, M. Zhao, Y. Tu, and J. Liu, "Multivariate autoregressive models and kernel learning algorithms for classifying driving mental fatigue based on electroencephalographic," *Expert Systems with Applications*, vol. 38, no. 3, pp. 1859–1865, 2011.

[27] R. C. Coetzer and G. P. Hancke, "Eye detection for a real-time vehicle driver fatigue monitoring system," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011, pp. 66–71.

[28] Y. Jiao, Y. Peng, B.-L. Lu, X. Chen, S. Chen, and C. Wang, "Recognizing slow eye movement for driver fatigue detection with machine learning approach," in *Neural Networks (IJCNN), 2014 International Joint Conference on*. IEEE, 2014, pp. 4035–4041.

[29] X. Peng, Z. Xia, L. Li, and X. Feng, "Towards facial expression recognition in the wild: A new database and deep recognition system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 93–99.

[30] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," in *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.

[31] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[32] MATLAB, *version 7.10.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc., 2010.