Machine Learning Project

«Machine Learning for Pattern Recognition»

Winter Semester 2015-16

Kalodimas Panagiotis
February 2016

Chapters

Figures

1. Introduction

In this thesis a pattern recognition algorithm was designed using machine learning methods for the usage of human face detection within into-the-wild images. The problems has to be solved are not only the part of detecting a face within the image but also to classify the detections and separate them from faulty detections due to the image noise (into-the-wild images). Any shape within the image that is similar to the human face can cause this kind of faulty detections.

One of the greatest problems in pattern recognition is the ratio between the ability of an algorithm to detect objects that exists and objects that do not actually exist. If the model used is a flexible generative model then usually the detection efficiency is high but the fake detections ratio is also high reducing its reliability. On the other hand when a strict model is used then the fake detections ratio is reduced but also the real detection ratio is reduced causing again the reduction of the algorithms' reliability. The real challenge is creating a model that has high real detection efficiency and low fake detection ratio. This would offer a algorithm with a great reliability.

2. Performance Indexes

An algorithm performance is depended by three kinds of detections

1. Real detections: A detection can be consider a real one when the algorithm returns that the object is detected in a specific place and this object does actually exists in this location within the image
2. Failed detections: A failed detection exists when there is an object within the image but the algorithms does not return that it has detected an object in the place it exists. The algorithm has actually missed the detection.
3. Fake detection: A fake detection exists when the algorithm returns that an objects exists within an image in a specific place but in reality there is no object of this king on this place. The algorithm has been actually fooled by the image noised.

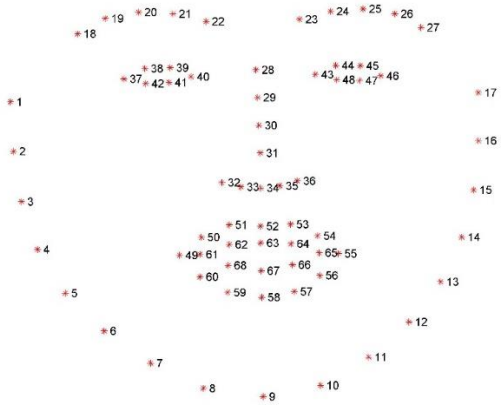
Using these kinds of detection there are some indexes that are going to be used in this thesis for watching the algorithm performance. These indexes are the Reliability index (Function (1) and the detection efficiency (Function (2). On these two functions all the detections cost is considered to be equal. Although in some cases, depending on the application using the algorithm, the cost of the different kinds of detections can be different. For example, sometimes the detection efficiency can be considered more important than the reliability that means that the real detections cost is greater than the fake ones.

$$Reliability = \frac{Real}{Real + Failed + Fake} \quad (1)$$

$$Detection\ Efficiency = \frac{Real}{Real + Failed} \quad (2)$$

3. Algorithm Abstract

The algorithm we designed calculates an images into its HOG descriptors convolves it with specific filters. The convolution process return arrays called filters responses that contains high values in the pixels cells where specific landmarks of the human face exists. Every filters is designed to produce high scores when it is convolved with the HOG descriptors of a specific human face landmark. These landmarks are shown in image 1 and 2. These 68 landmarks are globally acceptable and used in multiple algorithms.



1. Global prototype of 68 human face landmarks

2. The 68 landmarks of the human face according to the global prototype

The landmarks that the 68 filters detect belongs to parts of the human face like the eyes, the eyebrows, the nose, the mouth and the jaws. This algorithm aims to firstly detect these parts of the human face and later to detect the whole face by combining these parts positioning within the image. The algorithms uses the same strategy for combining the human face landmarks on parts detection and combining the human face parts for the whole face detection.

4. Machine Learning Applications in the Algorithm

In the design of this algorithm, machine learning applications were used in many stages.

The first stage that the algorithm is using the machine learning methods is at the classification of the filters responses results. There are a lot of shapes within an image that can possibly be similar with the landmarks that a filters is designed to detect. This would cause the filter to produce high values on the corresponding cells. Simultaneously, not all the images create the same results as the sharpness sometimes affects the filter performance. In addition the face view angle and its positioning can also affect the filter performance. On this case the machine learning classification techniques are needed in order to analyze the filters responses values and decide which of them should be considered as the landmark detection and which of them as noise

At the procedure of the landmark or the parts positioning combination in the parts or face detection process the machine learning methods are also useful. Firstly, a data analysis is needed for modeling the position relations between the landmarks and parts. Then a classifier has to be

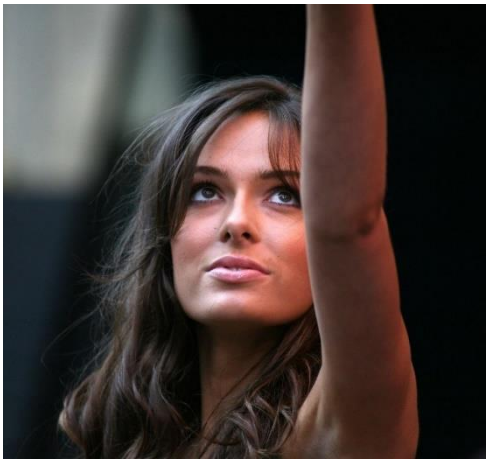
designed in order only the right parts of the image to be selected for this combination and not to be affected by neighbor noise.

The combination of the human face parts results to the face detection. This comes from the usage of a parts responses arrays that also result to the face response array. This array contains values that might be produced by the detected parts of the human face or by the image noise. On this stage it is important to analyze this results and decide which values are related to human faces and which one to noise. This is actually a pure classification.

One more stage where the machine learning methods are used is on the combination of the human face parts. At this stage what is very important is to understand which parts of the human face are more reliable and can be proved more critical in the whole face detection process. It is needed to add weights to every one of them test multiple values until a weight vector is produced that return the maximum reliability to the detection procedure. This is actually a regression problem.

5. Training

For the algorithm training procedure a set of images created especially for that purpose was used. This training set is offered with positioning information of every human face landmark in every image and this is exactly what is needed for the training procedure. This set contains 2000 images with faces within them and their positioning information. It also contains a set of 330 images also with faces within them and the same information for evaluation. For training the algorithm against the image noise, a set of 1330 images without faces within them was used.



3. Training Set Image Example

4. Noise Training Set Image Example

6. Human Face Landmark Classification

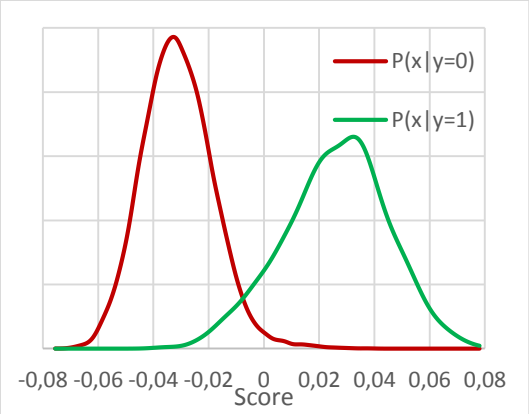
Using the 2000 face training set and the 1330 noise training set images, an analysis of the landmark filters responses was made. The convolution process of the landmark filters with the training images HOG features returned a range of values in the filters responses arrays that had

to be processed in order to separate the noise values from the landmarks ones. As is described in the following paragraphs, a function was designed for classifying these values and return the possibility density of each one of them to be a landmark detection. The reason this method was used is because on this stage it is too early for rejecting the landmarks that are not clearly separated from noise. The combination of the landmarks is one affective method of rejecting the noise. More methods are used in later stages that also clear the noise. So in this stage it is important that even values with low possibility of being a landmark to be forwarded in the next stage. This method is also making the algorithm more configurable as it gives the opportunity of selecting the certainty with which the landmarks detections will occur.

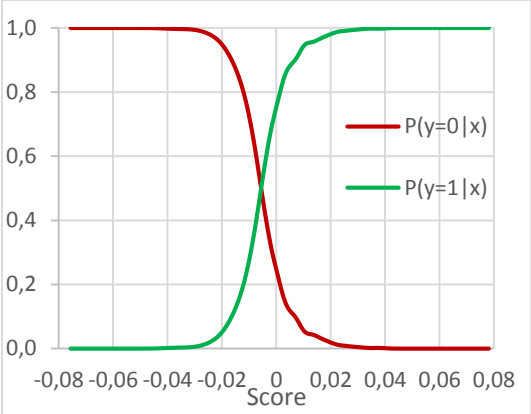
The first stage of this analysis was to measure the value range that the faces training set produced and create its distribution. The same procedure was used for the noise training set using only the values of the noisy data that produced values in the same range with the face training data. What worth to me referred is the face that only a set of 2000 values was available for every landmark detection value while more than 5 million was collected by the noise training set as every area of an image can be considered a noise creator if it is not the landmark one. Even if the possibility of a noisy area to create a filters response score in very low the fact that the largest part of an image is a noisy are gives the conclusion that the noisy inputs in our system are much larger than the real detections ones. Even the human face area within an image creates noisy inputs as every landmark only holds a small area over it and the rest parts of the human face creates noise on every landmark filter response.

The 68 landmark detection filters have maximum performance in faces at the size of 100 pixels high. This size of face corresponds to filters responses array of the size of 20 cells (after the creation of the HOG features and the convolution with the filter). Using this information and considering that a human has to hold an area of 100x100 pixels (20x20 cells in the filters responses array) in order to be detected, it is easy to understand that the ratio of landmark and noise inputs cannot be greater than 1/400. This means that for every landmark input in our system, at least 400 noise inputs have to be processed. In real world though this ratio is much lower as the greatest part of the images containing faces is the background and the faces capture only a small are of the image.

For the analysis of the noise distribution the maximum number of training samples was used. The results of one of the landmarks are shown in the graph 5. As seen in this graphs there is an important spacing between the scores of the landmark detection and the highest noise scores. By this distributions it is easy to produce the possibility density of a score to be produced by a landmark or by noise as shown in the graph 6.

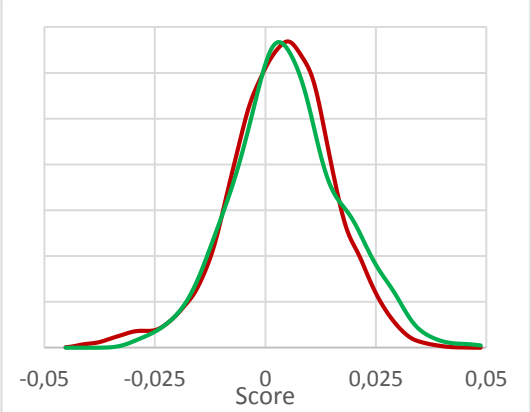


5. Normalized distribution of scores values of Noise (Red) and Landmark (Green)

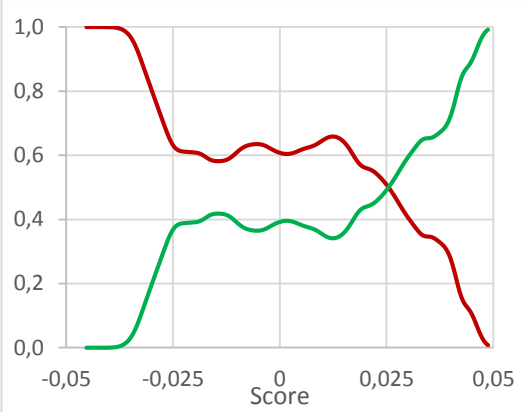


6. Possibility Density graph of Noise (Red) and Landmark (Green)

By the graphs 5 and 6 it is visible that the classification of landmark and noise scores can be an easy process. In the graphs 6, the possibility density function of the landmark distribution is much similar to a sigmoid function. Although the majority of the scores distribution of the landmark responses are similar to those in the graph 5, there is a small (less than 5) amount of landmark filters that produces scores distributions similar to the one of the graph 7. In this graphs what can be seen is that it is impossible to classify the scores produced by the landmark filter with a great certainty. As seen in the possibility density graph (Graph 8) of these distributions the certainty of the classifier does not exceed the 65% in the largest range of the distribution.

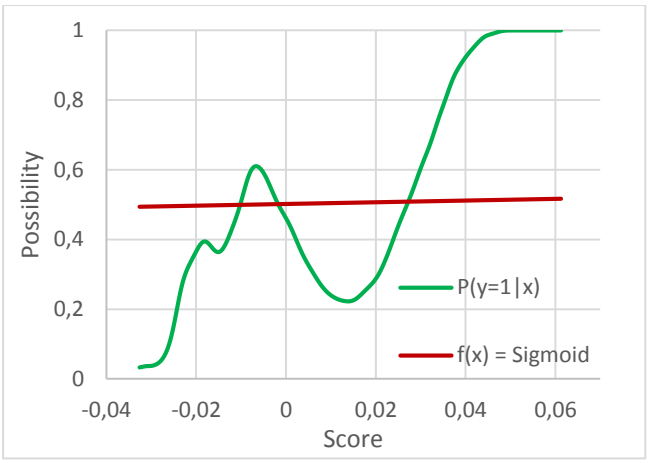


7. Special case of noise and landmark distribution with great uncertainty



8. Possibility Density of special landmark and noise distribution with great uncertainty

In the graph 8 it is visible that if the possibility density of the landmark filter is tried to be profiled using a sigmoid function it would lead to a sigmoid one with a great slope like the one shown in the graph 9. The results of using a sigmoid function like this one would produce return values greater than zero in the areas left of the score range is shown in the graph. As a result, scores much lower than the landmark range ones would produce outputs similar to the ones of the scores in the range of the landmark ones. This way our system would create more noise instead of removing it.



9. Faulty usage of sigmoid function in our system

According to the previous paragraph a system was designed that firstly classifies the scores rejecting the certain noisy ones and then using a sigmoid function to return as output a value that represents the possibility of the score to be a landmark detection. The reason that not only a classifier is used that probably would classify the scores according to its possibility (greater than 0.5) of being a landmark is because the areas around a landmark create also high score values. These would result of classifying more than one scores as the detected landmark. Then in the next stages it would be impossible to select which one of them is the right one and locate the right position of the landmark within the image. On the other hand the score that is produced when the filters is convolving exactly at the position of the landmark is always greater than the scores of the areas around it. This way the exact position of the landmark can be detected. Using a function that does not return a true or false output but a certain range values expressing the belief of having a landmark detection, this difference between the correct and the neighbor area scores is retained. So the ability of holding the information is needed for right position detection of the landmark remains. The greatest value represents the right position. In the image 10 a filter response example is presented showing how the area around a landmark reacts in the convolution process.

10. Example of the 37 landmark filter response

For avoiding any unwanted result as described above a “safety” classifier was created that makes all the input scores lower than a limit to have a zero output. The input score limit was calculated

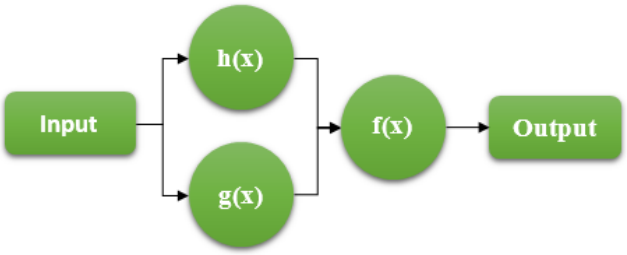
using the landmark scores possibility density and intuitively was set to the score values that have less than 0.1 possibility of being a landmark (0.9 possibility of being noise score). All this score values have system output equal to zero meaning that the system consider this score only as noise and for no case as a landmark score. The reason that the score possibility was chosen to be as low as 0.1 was because it was not considered a good tactic to reject all the low score values at such an early stage. Choosing the most certain landmark detection is later stages can be configurable is the possibility of every detection is known. This "safety" classifier acts as exactly as a perceptron would in a neural net with one input and one output.

Our system is actually a neural network with two hidden nets. The "safety" function is the one and the possibility estimator the second one. The "safety" function h(x) is described by the expression (3) shown below while the possibility estimator g(x) function is expressed by the (4) expression. The whole neural net system function is expressed by the expression (5) below and is graphically presented in the figure 11.

$$h(x) = 1 \text{ when } x \geq \text{Threshold}$$
$$h(x) = 0 \text{ when } x < \text{Threshold} \quad (3)$$

$$g(x) = sigm(x, \sigma, \mu) \quad (4)$$

$$f(x) = h(x) \times g(x) \quad (5)$$



11. Neural Net for landmark classification and possibility estimation diagram

The difficult part of this neural net was to design the possibility estimation function g(x). For doing that a sigmoid function was used as a kernel function. What is very important though is the ability of the possibility estimation function to classify correctly the input data. This means that score values that have greater possibility (greater than 0.5) of being a landmark score should always get a possibility estimation greater than 0.5. What is also important is that the g(x) function should produce the least error on estimating the possibilities of scores that are consider to be landmarks (possibility density > 0.5). For designing such a kernel function it is need to estimate the right mean and variance of the sigmoid function used as kernel.

The first part that was calculated was the mean parameter of the sigmoid kernel. That's because all the score values greater than the mean would always have a returned value greater or equal to 0.5. This is exactly the solution to the classifier was described in the previous paragraph. After calculating the mean parameter the second parameter to estimate is the variance that affect the slope of the sigmoid function. This slope would affect the accuracy of the possibility estimation and the function error compared to the real possibility density of the filter (Graph 6). As the possibility estimation of the more certain scores is more significant, the sigmoid kernel was

designed in order to produce the least estimation error for this range of score values without paying attention to the score values with low possibility of being landmarks. The estimation of the mean and the variance parameters was estimated using the linear regression method of the machine learning theory. First for the mean and then for the variance.

In the first stage of the linear regression process the mean parameter value was estimated. What was desired was to separate the score values that have possibility density greater than 0.5 with the ones that have lower. By calculating the classification error of these two classes through the whole score range as its size allows it the costless mean parameter value was estimated. The functions used by the linear regression method are the ones shown below (*(6)*, *(7)*, *(8)* and *(9)*). As seen in the function *(9)* the moving step through the score range was not stable flexible according to the error changes. This way the linear regression process converge faster to the most efficient mean value without losing its reliability.

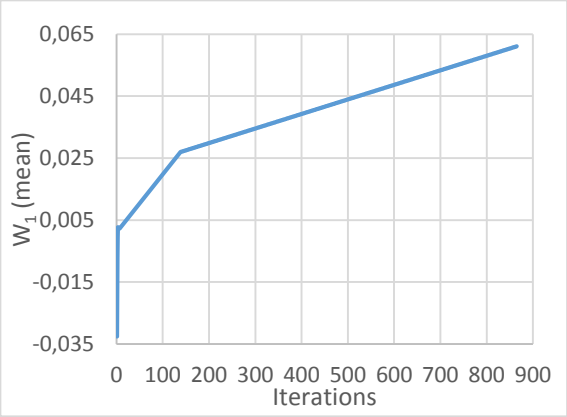
$$h(x) = [x >= 0.5 \rightarrow 1, x < 0.5 \rightarrow 0] \quad (6)$$

$$E_i = \sum (t - h(x))^2 \quad (7)$$

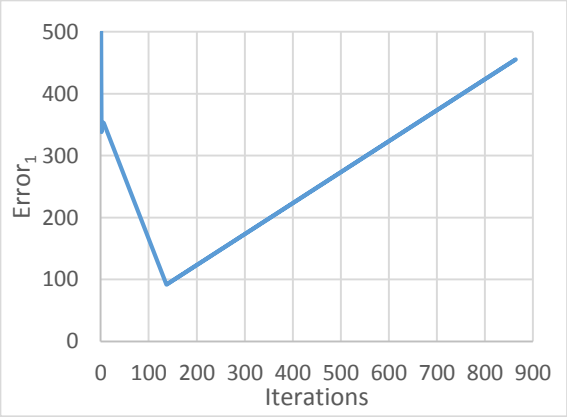
$$\partial E_i = E_i - E_{i-1} \quad (8)$$

$$w_{i+1} = w_i + \min(a \cdot \partial E_i, 1) \quad (9)$$

The scores values range is small and the system dimension just one so the linear regression procedure last time was short. As shown in the graph 12 and 13 examples of a single landmark kernel function parameters, the regression steps needed was about one thousand.



12. Mean parameter converge graph during the regression process



13. kernel function error during the regression process of the mean parameter

The second stage of the kernel function parameters estimation was the linear regression for the variance (slope of sigmoid function) parameter. For the linear regression of the stage the functions (10) to (13) were used. The start value of the variance parameter was 10000 that produce a sigmoid function of a 90 degrees slope. For the linear regression step parameter, parameter α, the value of -500 was used as after multiple testing it was proved to be the most reliable as with this value the most efficient and fast results came. As is visible in the expression (11) the output values used for the error estimation function was those that succeed larger than

0.5 outputs. That is because, as described above, it mi more important for the kernel function to be accurate in the detections that are more likely to be landmarks.

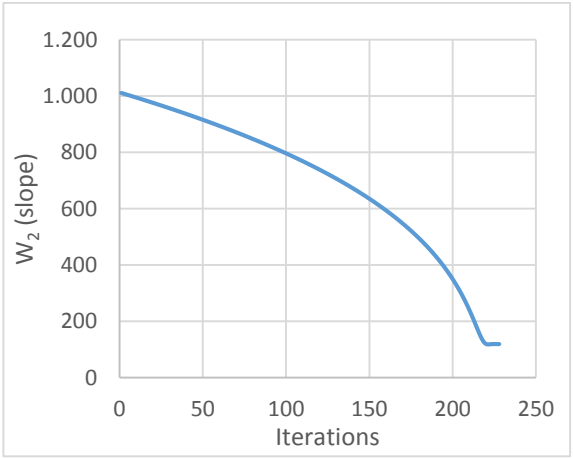
$$g(x) = sigmoid(x, \sigma, \mu) \tag{10}$$

$$E_i = \sum \left(t_{>0.5} - g(x)_{>0.5} \right)^2 \tag{11}$$

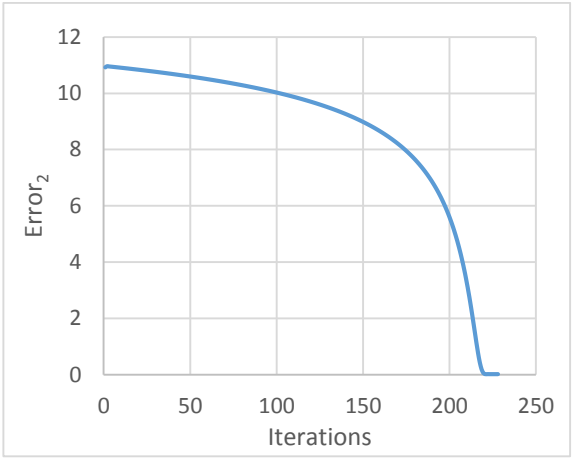
$$\partial E_i = \frac{E_i - E_{i-1}}{w_i - w_{i-1}} \tag{12}$$

$$w_{i+1} = w_i + a \cdot \partial E_i \tag{13}$$

As seen in the graphs 14 **Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.**and 15 the regression process of this stage needed less regression steps to converge. This is because the regression scale (parameter α) that was used was larger. It was also the fact that the sigmoid function slope was not large enough. A difference between the two regression proccess stages is that in the first regression proccess the whole range of possible values for the mean parameter was scanned while on the second regression process this did not occur. This happened because in the first occasion there can be more than one extrema in the error function as for example in a case like the left graph 16

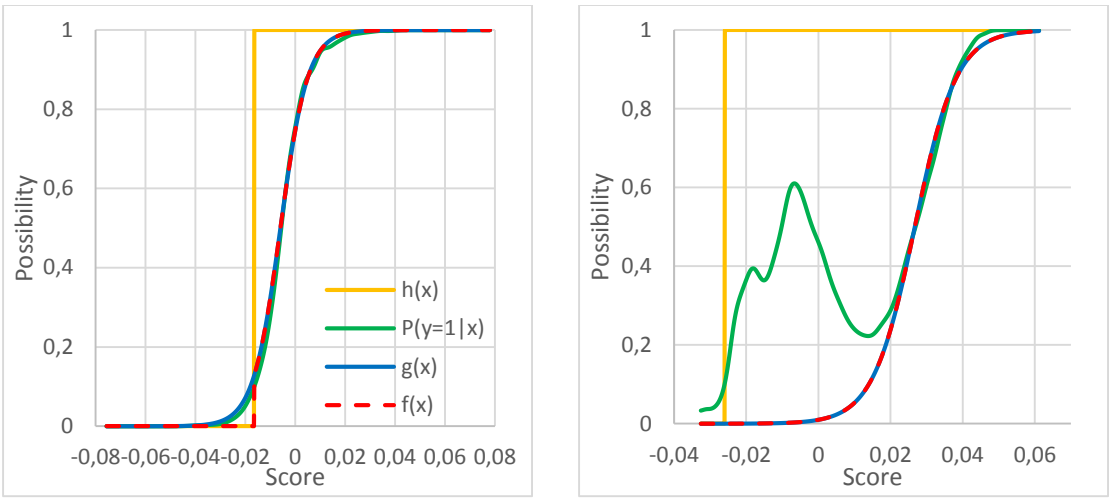


14. Variance parameter converge graph during the regression process



15. kernel function error during the regression process of the variance parameter

In the graphs 16 the results of the regression process and the final neural net output results are shown. The green line shown the actual possibility density of the filter scores as calculated by the training data. The orange line is the output results of the “safety” hidden net h(x) of our system while the blue one shown the output of the possibility estimation hidden net g(x). The whole neural net output is shown by the slashed red line. In the left graph 16 it is visible the effect of the h(x) “safety” hidden net while in the right one its does not affect the output at all.

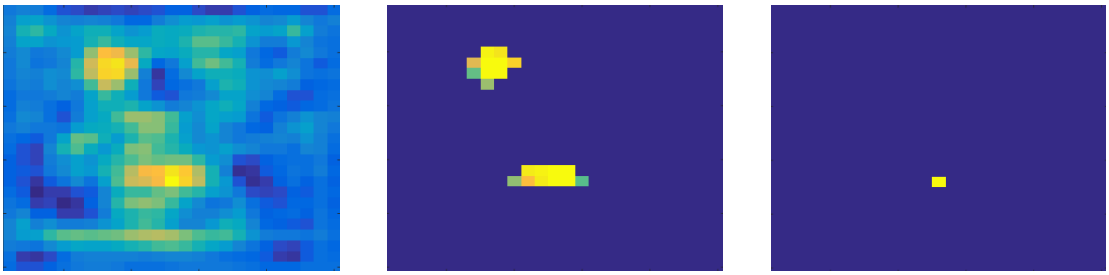


16. Linear Regression Results Example for Landmark Detection System

Using this procedure of the neural net the algorithm transform the scores arrays to possibility arrays. Instead of arrays containing unknown and different range values of multiple filters responses, now all the filters responses arrays contain scores in the range of 0 to 1 which express the possibility of a landmark detected in the corresponding area of the image. Practically the majority of the responses arrays cells are containing zero values and only in noisy and landmark areas the values of the cells create peaks. These peaks are later distinguish and only the highest value remains as a result of further processing described later in this thesis.

7. Face Parts Detection

The results of the convolution process of the landmarks detection filters and the HOG descriptors of the image are the filters responses arrays. On these arrays the algorithm applies the neural network described in the previous chapter and creates the landmarks possibility arrays. In these arrays the cells values range is between 0 and 1 expressing the possibility of a landmark detection on the corresponding area of the image. Usually these arrays are filled with zero values while only in specific cells and probably in its neighbors exist values greater than zero as the example image 17 shows.

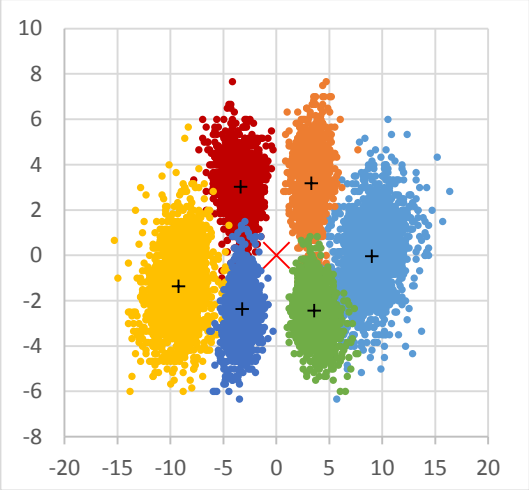


(a) *(b)* *(c)*

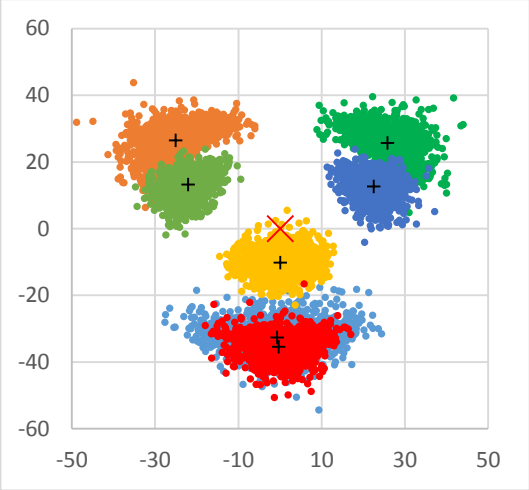
17. Example of the filter response array (a) transformed to possibility array (b) and cleared by the noise (c)

For the face parts detection the algorithm adds the landmarks possibility arrays shifting them in a way that the candidate landmarks positions would move in the center of the part. This way, in

an ideal situation, all the peaks of the possibility clouds (Image 17) should be added in the same cell of the part response array. Although in the real life the landmark positions are not always in the same position according to the part center as for example on closed or open eyes or mouth. This problem is fixed by creating a filter that adds all the cell values in a certain area expecting them to be the possibility values of the landmarks possibility arrays. In order to do that add avoid the collection of the possibility values created by noise and by the landmark neighbor cells a filter removing the noise and keeping only the top value in a specific area is used creating the results of image 17. As it is expected that every landmark appears one time at every face this filter keeps only the highest possibility value in an area covering a whole face (20x20 area).

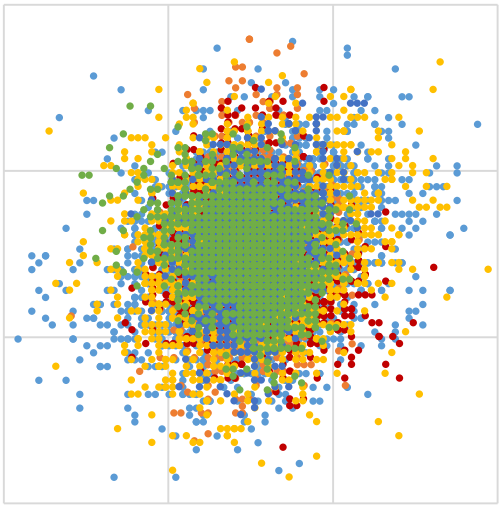


18. Left Eye Landmarks Positioning Distribution

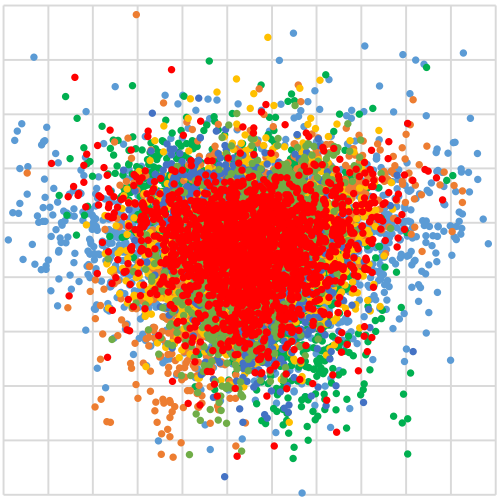


19. Face Parts Positioning Distribution

In order to know the area in which to collect landmark possibility values it is necessary to know the positioning relation between the parts and the landmarks distributions. The face parts detection using their landmarks is almost the same procedure when using the parts for detecting the face. The landmarks positioning distributions around the parts center are a useful information for creating a system that collects them and estimating the center of the part. In the image 18 the left eye landmarks positioning distribution is presented. By these distribution the mean distance of every landmark from the left eye center is calculated (cross inside the cloud). By adding all the landmarks means (crosses) in the center of the left eye part (X Symbol) a new distribution is creating showing the area that the possibility array values are expected to appear, as shown in image 20. The same thing happens when the algorithm tries to detect the whole face, as shown in the images 19 and 21.

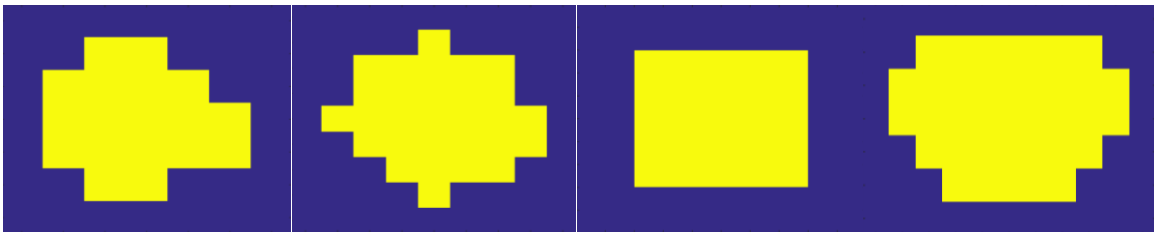


20. Left Eye Landmarks Total Distribution after Centered Addition



21. face Parts Total Distribution after Centered Addition

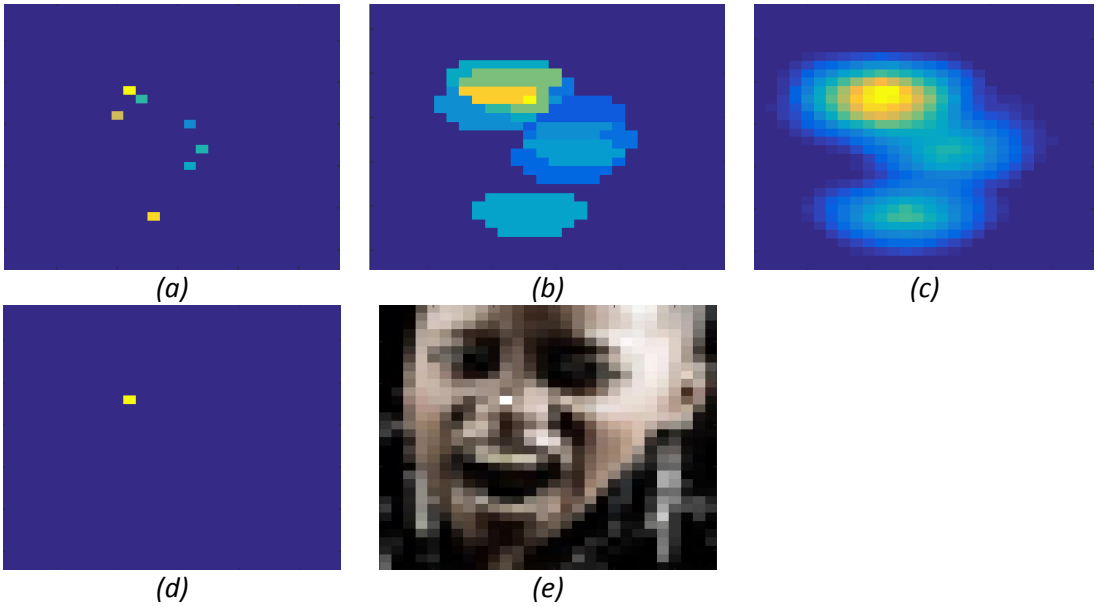
By using the data distributions of the landmarks or parts (in face case), filters can be designed that collects the possibility values come from the filters (or parts) possibility arrays and add them together creating the parts responses (of face one). In these case as the distributions are large enough to create to large filters the usage of a margin that holds the filter size until there are more less 1% of training samples was needed. The usage of a margin like that is technique used by the SVM in the machine learning theory. As our system is discrete and the filters applying the SVM system are cells in an array instead of keeping the support vectors also inner area vectors was used. For every cell that is supposed to constitute the area of candidate landmarks, one support vector was used. These filters are actually operate like an combination of SVMs and 1-Nearest Neighbor classifying the areas to the ones that the landmarks possibility values can appear after the possibility arrays addition and of those out of that range. Filters like these are the ones shown in the images 22



(a) Nose *(b) Mouth* *(c) Left Eye* *(d) Face*

22. Landmark Collection Filters for Nose (a), Mouth (b), Left Eye (c) and the whole face (d)

So, as a result of keeping the peaks of the landmarks possibility arrays and adding them centered to the part center, a part response array is created as the one shown in image 26(a). Then the landmarks collection filters select the values in the part distribution area and add them creating value hills like those in the image 23(b). Then a smoothing filter regularizes these value hills and creates a peak at its center (image 23(c)). At last the noise and the neighbor values are cleared and only the highest (peak) value remains in the response array of the part revealing the position of the detected face as shown in image 23(d) and 23(e).



23. From Parts to Face Detection Data Processing

8. Face Detection

By the time that the landmarks possibility arrays have been estimated and by their combination the face parts responses a question is asked before the algorithm uses these parts responses arrays for the final face detection. Are all the parts of the human face the critical in the detection process? Are the landmark filters reliable enough for the parts detection? Do some parts detection create more noise than detections?

Some parts of the human face have simple shapes resulting to the production of much noise during their detection procedure (ex. Eyebrows). Others may not always be visible in the images, like the eyes when sunglasses are wearied or parts of the jaws when there long hair in front of the face. Some of them may have a great variety of shaping due to the face viewing angle and the natural differences that exist in human faces. All these cases make the parts of the human face non-equally critical during the detection procedure. The conclusion is that the machine learning regression procedure is the solution for testing and understanding which of them are the more crucial and should be used better in the face detection procedure.

The detection procedure of every part of the human face produces the part response array result. The combination of these responses arrays, as exactly is done with the landmark ones, is returning some values in the face response array which are the final criteria which is used for the final decision of having or not a face detection. In this final stage of the face detection procedure is where the machine learning method of using a weight vector changing the participation cost of every part rising the cost of more efficient and reliable parts and reducing the cost of the more noisy and less useful ones. Using this vector the regression method can be applied in order to converge to the most efficient values of its columns. The face detection function that creates the face response arrays is described by the expression (14) shown below. As seen in this function the algorithm adds the parts responses arrays by multiplying them before with the corresponding

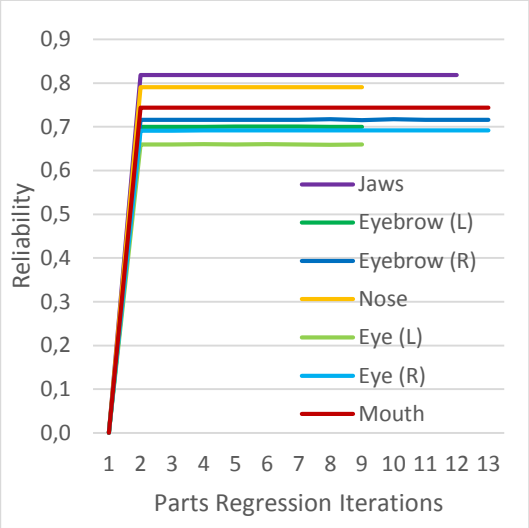
eights vector column. These way some parts have more important role in the production of the face response arrays.

$$f(face) = \sum_{i=1}^{7} w_i \cdot f(x_i) \qquad (14)$$

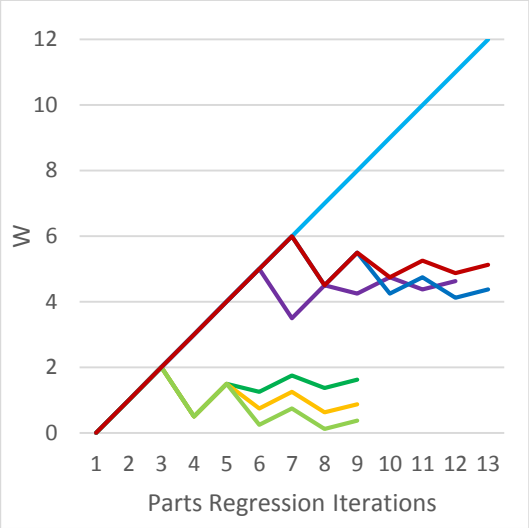
In the attempt to detect the best W vector values the regression procedure was used. As it is sensible because our system is a seven dimensions system and it uses about 4000 samples of training data, despite some techniques used for its speedup, the whole procedure was a much time consuming one. By calculating the reliability performance of the algorithm while testing different values of the W vector the regression procedure converged to a most profitable set of W values. The reliability function is not a stable and continues one as the reliability performance of the algorithm is affected not only by the true and failed detection but also by the faulty ones. So while a change of the value W_i of the vector might cause the increment of the true detections, it might also increase the number of faulty ones. The ratio between the true and the failed detection is the one that makes the reliability performance to increase or decrease. For that reason after a lot of tests the conclusion was that there is no reliable stepping function that can form the changes in the W vector values but that the values of W had to be chosen through a range of predefined values.

First of all, the values of the vector W are all set to zero. For every part of the human face the corresponding column w_i of the vector W starts rising with step size to 1 until a reliability extreme is detected. Then the step size is scaled by the factor α that in this procedure haw the value of 0.5. After changing the size of the step the next w_i values would be the ones ±α*step from the w_i value that caused the extreme. The algorithm then would check again for the reliability extreme and it will scale again the step size. Following this execution floe the algorithm converge to the global extreme of the reliability function. The idea is to scan the effect of the w_i value on the reliability function starting with a large regression step. Then following the temporary extreme in every iteration reduce the size of step and rescan a smaller area around this extreme.

The procedure of the previous paragraphs is repeated for every part of the human face. Having the rest face parts w values set to zero the training process is followed detects which of the parts is the most crucial for the face detection process. Except of the regression method applied for every part of the human face for detecting its best w_i value the training algorithms applies one more regression process perpendicular to the ones of the face parts. After the regression procedure is completed for all the columns of W (face parts), the training algorithm selects the one that produced the best reliability results and holds the w_i value that caused. Then it repeats the previous procedure initializing the W vector with zeroes, except of the w_i that caused highest reliability in the previous iterations. In every iteration one more w_i value is retained building the W vector values structure. In the graphs 24 and 25 the Reliability and the W vector columns converge is shown for the first iteration of the face detection function regression process. As shown in these graphs for every face parts another regression process is applied.



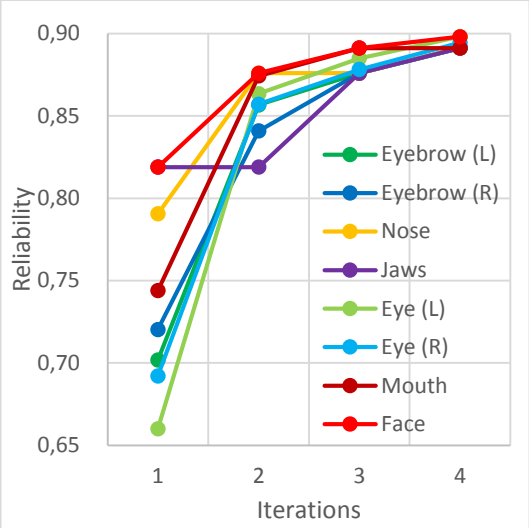
24. Reliability Converge in the Parts Regression Process



25. W_i Converge in the Parts Regression Process

At the face function regression procedure at every iteration the wi value that cause the best reliability value is saved and used as an initializing value of the W vector. This was at every iteration a new initialization value is added in the vector. The case where a wi value might be update instead of a new value to be added is not excluded.

In the training regression procedure of this algorithm the face function regression procedure needed four iteration, until it converge to a maximum reliability value. As the table 27 presents (also in the graph 28), in the first iteration the jaws parts produced the maximum reliability value of 0.819. The regression algorithm then sets the w_i of the jaws as an initial value of W vector and runs again the same process. In the second run the nose part caused the maximum incremental of the reliability to 0.876 and its w_i values was also set as an initial value. This procedure was repeated two more times resulting to the W vector of the table 31.

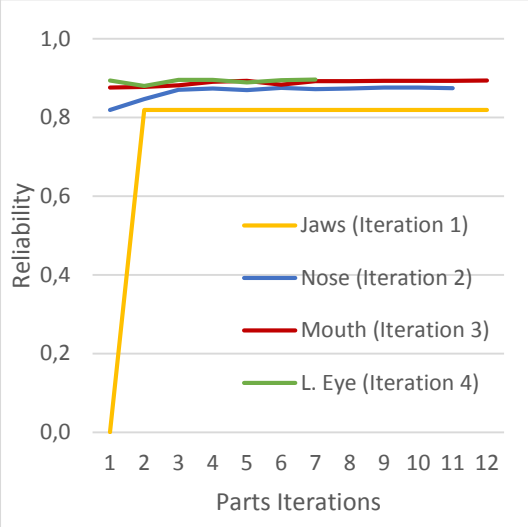


26. Reliability Converge of All Parts Regressions for every Iteration of the Face Function Regression Process

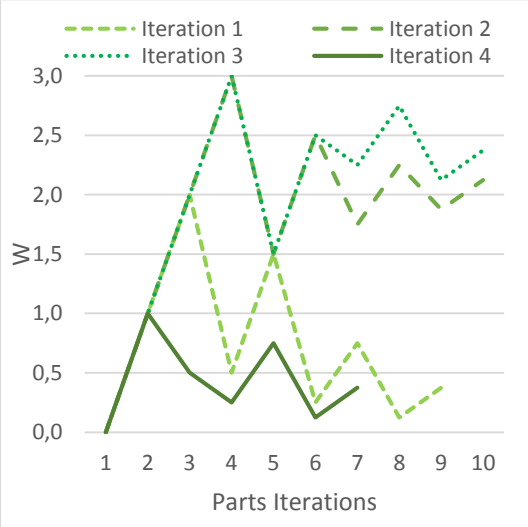
Face Function Regression Reliability Results				
Part	Iterations			
	1	2	3	4
Jaws	0,819	0,819	0,876	0,893
Eyebrow (L)	0,701	0,859	0,881	0,893
Eyebrow (R)	0,717	0,840	0,876	0,893
Nose	0,791	0,876	0,876	0,894
Eye (L)	0,660	0,863	0,885	0,896
Eye (R)	0,692	0,857	0,878	0,893
Mouth	0,744	0,875	0,893	0,893

27. Reliability Results Table for all Parts in every Iteration of the Face Function Regression Process

In the graphs 28 below, the convergence of the reliability in the parts that succeed the maximum one at their regression process is shown. As seen in this graph the reliability value at every iteration starts form the top of the previous one. This way the algorithm searches the value of the W vector that can rise more the reliability value. When none of the W vector columns can cause a significant rise then the regression process has converge enough and the procedure is completed. The minimum rise limit was set to 0.005.



28. Reliability Converge of the four Critical Face Parts in

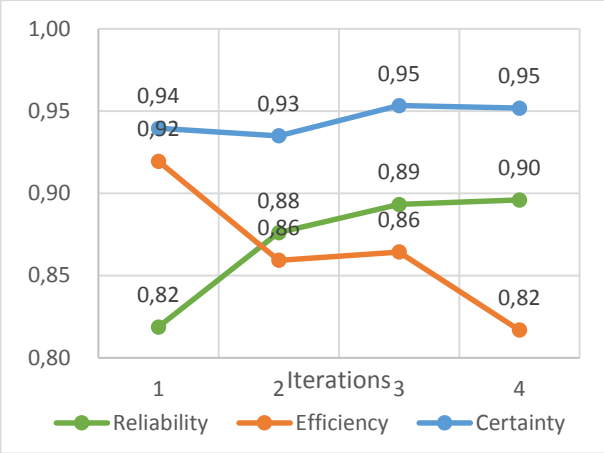


29. Διάγραμμα ιδανικών τιμών διανύσματος W κατά την διαδικασία του Regression

In the graph 29 the W_i of the left eye face part convergence in every iteration of the face detection function regression process. As is visible the initialization of the W_i values of the other three parts affects it new convergence. As the reliability value is increasing is getting more difficult to be

affected by the rest parts. As is visible in the table 27 the effect of the left eye in the whole algorithm reliability is tiny and it could be avoided.

In the next graph the performance indexes values are shown during face detection function regression process. As is visible the as reliability performance is increasing the detection efficiency is reducing. This is very sensible because as the algorithm becomes stricter in face detections it also reduces the number of fake detections that makes it more reliable. As the ratio of fake and true detection reductions number is getting closer the reliability reaches its extreme. As is also visible the value of certainty with which the algorithm decides if a detection is a true one is very high and it is not affected by the reliability and detection efficiency values. This is because the face scores values distributions are very separable as presented in graphs 32 in the next chapter.



30. Algorithm Performance Indexes Convergence during the Face Detection Function Regression

Regression Results		
Face Parts	Τιμές W	Reliability
Jaws	4,625	0,819
Eyebrow (L)	0	-
Eyebrow (R)	0	-
Nose	2,625	0,876
Eye (L)	0,375	0,896
Eye (R)	0	-
Mouth	4,125	0,893

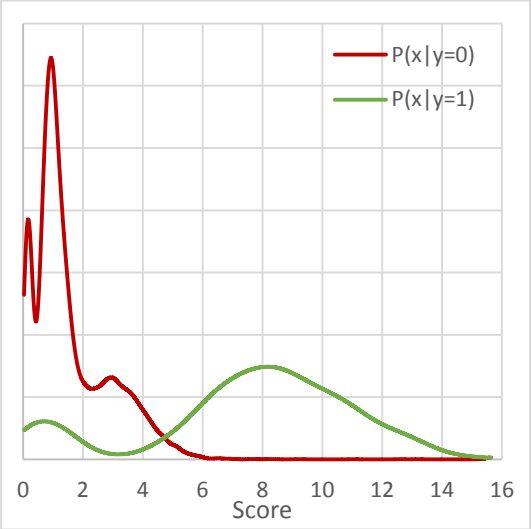
31. Face Detection Function Regression Results Table

In the table 31 the final results of the face detection function regression process are shown. What can be seen is that some parts of the human face do not offer any help at the detection process at all and can be omitted from the detection procedure. Even the left eye offering in the procedure is tiny and easily could be omitted. A conclusion can be made is that the parts of the face constitute by more landmarks offer better detection performance and if the same side eyes and the eyebrows could be joined as one part, they may increase its participation effect in the detection process. On the other hand the part of the human face that constitute by multiple landmarks consume more time in the detection process than the less landmark ones. Observing the table 27 an assumption can be made that the combination of smaller part of the face could also offer high performance to the algorithm. For sure not as high as this set of face parts but high enough to sacrifice a small amount of reliability in order to gain an execution time speedup.

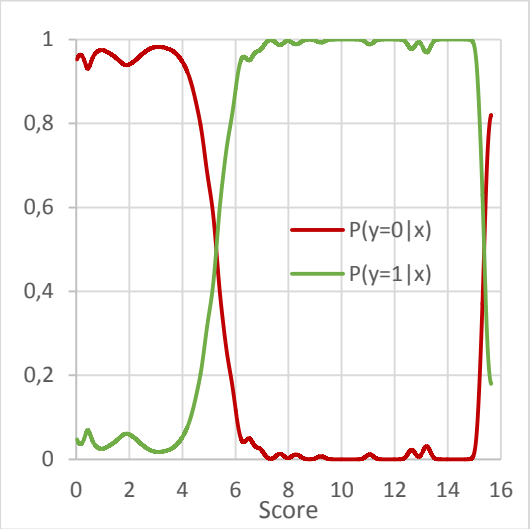
9. Final Classification

In the previous chapter the vector W of the face detection function estimation is described. In this chapter the design of a classifier that classifies the face detection function results in true and faulty detections is described. In the graph 32 the distributions of the results returned by the face detection function when a true (green line) or fake (red line) detection occur are shown. In this graphs is visible that the majority of true detections gets a separable return value from the face

detection function as shown in the possibility density graph in the graph 33. Looking these graphs makes it easy to design a linear classifier using the machine learning SVMs for example with just two support vectors.



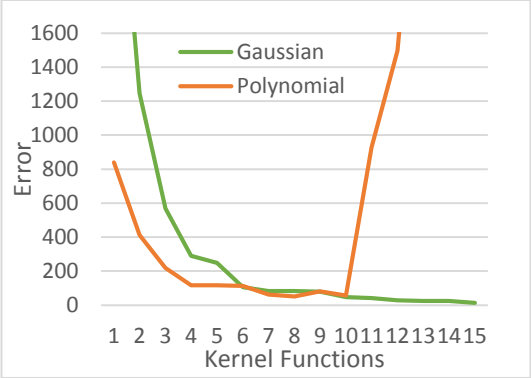
32. True (y=1) and Fake (y=0) Detection Scores Distribution



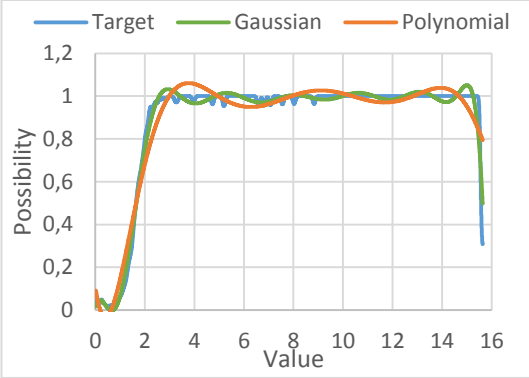
33. True (y=1) and Fake (y=0) Detections Possibility Density

Although it would be easier to create a simple linear classifier that classifies the detections using the probability density graphs (true detection the ones with probability density greater than 0.5) it was consider a more flexible tactic, a possibility estimation function to be created giving the classifier the ability to be configured. This way the algorithm user can decide the possibility threshold that a detection can be decided as a true one. The reason for doing so is that by reducing the possibility threshold for true detections, the detection efficiency is increasing and this can be a good reason for doing that. In some applications the detection efficiency can be more important than the reliability.

Using the training results a Face Detection Possibility Estimation function was designed. For this designed two kind of kernel functions were tested, polynomial and Gaussian ones. The number of the kernel functions in the FDPE function come after regression procedure. The regression procedure was also used for the variance parameter of the Gaussian functions. The maximum number of kernel function tested was 15. In the graph 34 the estimation error of the two kinds of kernel functions tested is shown as the number of kernels was increased. As it is visible the polynomial kernels converge better as the number of kernels is small but after the tenth one the estimation error is getting extremely increased. This attitude makes this set of kernel function a bit unreliable. On the other hand the Gaussian set of kernel function has a large estimation errors as the number of kernel function stays small, although as the number of kernel function is increasing, its performance is getting better and better with the estimation error to be slowly increasing.



34. Estimation Error of the FDPE function Regression Procedure



35. FDPE function results using polynomial and Gaussian kernel functions

According to the results of the regression process and by observing the graphs 34 and 35 the Gaussian FDPE kernels was preferred as they seem more reliable and efficient in the estimation process. With the FDPE function our classifier can be configured easily on how certain a detection should be to be classified as a true one. The FDPE function returns values in the range of zero to one representing, how sure is our model, for its detections to be true. The expression (15) is the one for our FDPE function while the expression (16) is the one for our classifier.

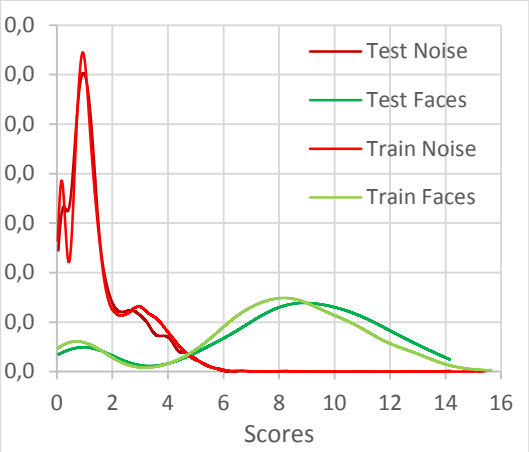
$$fd(x) = \sum_{i=1}^{i=15} W_i \cdot N(x, \mu_i, \sigma) \tag{15}$$

$$C(fd(x)) = True, \ when\ fd(x) \geq Threshold$$

$$C(fd(x)) = False, \ when\ fd(x) < Threshold, \ Threshold \in [0 \rightarrow 1] \tag{16}$$

10. Evaluation

The last stage of this project is the evaluation of the results using the testing set of the 330 images offered by the training set used just for the evaluation procedure. Running our model for this testing set the result distribution come of is the one shown in the graph 36. In this graphs the true and the fake detections scores distributions are shown for both the training and the testing ones. As it is visible the distributions are very similar. This last conclusion is evaluated by the performance indexes that are also very similar with the ones of the training procedure as shown in the table 37.



36. Evaluation Samples Detection Scores Distribution

Evaluation Results		
Indexes	Test	Training
Reliability	0,902	0,896
Detection Efficiency	0,845	0,817
Detection Certainty	0,948	0,952

37. Evaluation Results Table

11. Conclusions

In this project our face detection algorithms was trained using machine learning methods only in same critical stages of the detection process. The size of this project does not allow us to extend the training procedure to more stages than the ones referred. Despite that the face detection performance of the algorithm seems to be satisfying. This last observation makes it obvious that the performance of the algorithm can be increased more if it is trained in more stages like the part detection procedure using methods like the one used in the face detection stage. Another stage that the training can be extended is the landmark detection filters that was not designed in this project.

Another improvement could also be the extension of the training samples. In this project 2000 samples were used for modeling the human face. The viewing angles of a face combined with the differences between each human face and the large size of noise exists in into-the-wild images makes it clear that this training set is a small one. This claimed was referred also in the chapter 7 as in the filters of the image 22 there is no symmetry, as it should, and this is caused by the small number of training samples. More training samples would probably offer better performance in the algorithm.

At last, on more conclusion is that the number of landmarks constitute the human face parts affects its ability of contributing in the face detection procedure. In our model some of the human face part do not contribute at all. If instead of combining the parts, all the 68 landmark were combined for the whole human face detection this might cause a better performance to the algorithm. On the other hand as shown in the table 27 the algorithm can succeed good performance using less parts that means less landmarks. The usage of less landmarks in the detection procedure can offer the algorithm a great time speedup as the convolution procedure is the most time consuming one. As is understandable the more landmarks or parts are used the more efficient might be the algorithm but this costs detection time. The challenge is the selection of the most efficient ones succeeding the best performance to execution time ratio.