Spring 2016

Detection of Locations of Key Points on Facial Images

Manoj Gyanani
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

Part of the [Artificial Intelligence and Robotics Commons](#)

Recommended Citation

Gyanani, Manoj, "Detection of Locations of Key Points on Facial Images" (2016). *Master's Projects*. 470.
DOI: <https://doi.org/10.31979/etd.5ut9-kn86>
https://scholarworks.sjsu.edu/etd_projects/470

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

Detection of Locations of Key Points on Facial Images

A Project

Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfilment

of the Requirements for the Degree

Master of Science

by

Manoj Gyanani

May 2016

© 2016

Manoj Gyanani

ALL RIGHTS RESERVED

The Designated Project Committee Approves the Project Titled

Detection of Locations of Key Points on Facial Images

by

Manoj Gyanani

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2016

Dr. Robert Chun	Department of Computer Science
Dr. Thomas Austin	Department of Computer Science
Mr. Amit Singh Jadon	Software Engineer at Cisco Systems

ABSTRACT

Detection of Locations of Key Points on Facial Images

by Manoj Gyanani

In field of computer vision research, One of the most important branch is Face recognition. It targets at finding size and location of human face on digital image, by identifying and separating faces from the surrounding objects like building, plants etc. For the purpose of developing an advanced face recognition algorithm, Detection of facial key points is the basic and very important task, basically it is about finding out the location of specific key points on facial images. This key points can be mouths, noses, left eyes, right eyes and so on.

For implementation of solution, I have used amazon ec2 gpu instance and convolutional networks consisting of multiple levels. Outputs of multiple networks are fused at every level for accurate and robust evaluation. At the stage of initialization, high level features are extracted over the whole face region which helps in locating key points with high accuracy. Local minimum occurred by data corruption and ambiguity in difficult samples of image caused by occlusions, extreme lightings and large variations in poses can be avoided by this method. At later levels, training of networks is adopted to locally refine the initial predictions and the input supplied to them are limited to smaller regions around the predictions that are obtained in initial stage.

ACKNOWLEDGEMENTS

It would have been very difficult for me to complete my thesis auspiciously without the help and motivation from various people.

Firstly, I would like to thank my project advisor, Dr. Robert Chun, for his continuous guidance and support throughout the project. It would have never been possible without his helping hands.

I would also like to thank Dr. Thomas Austin and Mr. Amit Singh Jadon, for devoting their valuable time and being my committed members. They have helped me with their significant reviews and beneficial advices.

Additionally, I would like to express my gratitude toward my parents. They have always been with me in my good and bad times, they have always provided moral support to me. At last, I would like to thank all my friends who directly or indirectly supported me throughout the project.

Contents

1. Introduction	1
2. Background and Related Approaches	2
2.1. Background:	2
2.1.1. Detection of face and Metrics linked with human features	2
2.1.2. Inspecting expressions of face to know the emotional state of a person	3
2.1.3. Helping dysmorphology by disclosing facial structural defects	4
2.1.4. Providing aid in identifying an important person in images as well as in videos	4
2.2. Related Studies:	5
2.2.1. Method 1: Prediction using active shape model, r-part decision tree and neural network	6
2.2.1.2. Recursive partitioning decision tree approach:	10
2.2.1.3. Using Neural Network for key point detection	12
2.2.2. Method 2: Combined use of neural network and machine learning for building model	16
3. Prospective Procedure	20
3.1. Python Programming Language:	20
3.2. Lasagne for neural networks:	21
3.3. Theano for numerical computation:	22
3.4. AWS EC2 GPU Instance for faster computing:	22
4. Model Implementation	24
4.1. Description of method used	24
4.2. Description of data used	24
4.3. Flowchart explaining the procedure adopted for building model	24
4.4. Comparison of method used with other methods	26
4.5. Steps involved in the implementation of model:	27
5. Experimentation and Outcomes	31
5.1. Outcomes:	31
5.1.1. After training according to the initial model: The following results were observed	31
5.1.2. Testing the model on some sample images	32
5.1.3. After adding convolutional neural networks, following are the observations obtained:	33
5.1.4. After implementing data augmentation, following are the observations obtained	34
5.1.5. After changing momentum and learning rate over the course of time, following observations are obtained	36
For net 4:	36
5.1.6. For net5:	37

5.1.7.	After implementing dropout, Following observations are obtained	38
5.1.8.	Comparison with different methods:.....	39
5.1.9.	Runtime Complexity:.....	39
6.	Conclusion and Future Work	41
6.1.	Conclusion.....	41
6.2.	Future Work.....	41
LIST OF REFERENCES		42

LIST OF TABLES

1. Comparison of proposed approach with studies approaches.....	26
2. Training Loss, Validation Loss and ratio of Training Loss and Validation Loss for the initial model.....	31
3. Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after adding convolutional network.....	34
4. Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after implementing data augmentation.....	35
5. Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after changing momentum and learning rate for net4.....	36
6. Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after changing momentum and learning rate for net5.....	37
7. Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after adding dropout.....	38
8. Comparison of RMSE values of proposed approach with other studied approaches..	39

LIST OF FIGURES

1. Different Facial Expressions.....	3
2. Different Syndromes in births.....	4
3. Tracking Faces in Images and Videos.....	5
4. Greyscale image showing key points.....	6
5. Active shape model training from contours.....	7
6. Identification by ASM approach.....	10
7. Identification by R-part decision tree approach.....	11
8. RMSE values of key points predicted by R-part decision tree approach.....	12
9. Identification by R-part decision tree.....	13
10. Learning rate in R-part decision tree approach.....	14
11. Improved results in R-part decision tree approach.....	15
12. Training and validation curve in R-part decision tree approach.....	15
13. Sample Greyscale Image.....	16
14. Figure showing mean right eye patch and mean nose tip patch used in approach.....	17
15. Neural Network created in the approach.....	18
16. RMSE values with respect to number of hidden layers.....	18
17. Flowchart explaining the proposed approach.....	25
18. Training Loss and Validation Loss for the initial model.....	32
19. Prediction by model on some sample images.....	33
20. Training Loss and Validation Loss curve after adding convolutional network.....	34
21. Training Loss and Validation Loss curve after implementing augmentation.....	35
22. Training Loss and Validation Loss curve after changing momentum and learning rate for net 4	36
23. Training Loss and Validation Loss curve after changing momentum and learning rate for net 5.....	37
24. Training Loss and Validation Loss curve after adding dropout.....	38

CHAPTER 1

1. Introduction

As I am very much interested in Machine Learning. I would like to apply my knowledge of computers to one specific branch in computer vision research: Identifying the location of key points on facial images.

The problem that I have chosen to solve is how to efficiently and effectively detect keypoints like nose, mouth, left eye, right eye etc. in facial images. RMSE (Root Mean Square Error) value is used for measuring how effective this learning algorithm is. On hold-out test data set this RMSE provides a good measure of the generalization error of the proposed approach and my main focus is to lower this RMSE value.

I am planning to implement a machine learning algorithm which can efficiently and accurately detect facial key point locations.

CHAPTER 2

2. Background and Related Approaches

2.1. Background:

In field of computer vision research, One of the most important branch is Face recognition. It targets at finding sizes and locations of human faces on digital images, by identifying and separating faces from the surrounding objects like building, plants etc. For the purpose of developing an advanced face recognition algorithm, Detection of facial key points is the basic and very important task, basically it is about finding out the locations of specific key points on facial images. This key points can be mouths, noses, left eyes, right eyes and so on.

In this project, the main aim is to anticipate key point positions on images of faces. This designed prediction model can be used as an elementary unit in variety of applications, such as:

- Detection of face and Metrics linked with human features
- Inspecting expressions of face to know the emotional state of a person
- Helping dysmorphology by disclosing facial structural defects. Dysmorphology is the study of differences that are observed in the body structure, how this differences are originated and their classification
- Providing aid in identifying an important person in images as well as in videos

2.1.1. Detection of face and Metrics linked with human features

Metrics that are linked with human features are often know as Biometrics. For the purpose of providing access control and as a type of identification method, Verification of this metric is

used. Every individual has different metric identifier that makes them unique. A computer application that can be leveraged for verifying a person from an image is referred as face recognition system.

There are other systems also that can take care of human metrics verification like fingerprint scan. Authentication through face is more preferable, this is because in this method, a person is not required to make physical contact with the device used. He/She can get his/her identity verified from a distance without coming in contact with device. Face identification also helps in preventing crimes as images captured once can be always used to identify the person associated with the crime.

2.1.2. Inspecting expressions of face to know the emotional state of a person

An expression of face is created, when some muscles moves inside the skin of face. As per studies, this expression tells a lot about a person's emotional state. Human can endorse a facial expression by their choice or by force. Neural mechanism governs them all. Eyes play a very important role in facial expressions. A person can be judged on whether he is lying or not lying on the basis of eye blinking rate.

Fig 1: Different Facial Expressions

2.1.3. Helping dysmorphology by disclosing facial structural defects

The branch of genetics which makes researchers study and strive to explicate the defects of structures and patterns that are observed in human growth is referred to as dysmorphology. It is very important for families of children with dysmorphic symptoms to get meticulous checkup of their child. It will help in knowing more about their child's condition.

A dysmorphic child is different from a normal child in terms of his body structure. It may be congenital defect or hereditary disorder.

Fig 2: Different Syndromes in births

2.1.4. Providing aid in identifying an important person in images as well as in videos

For identifying a person in an image or video, Face recognition techniques are used. In broadcasting of video, human face processing methods have been of much interest among

researchers. The prime reason for this interest is that a human face gives ample information that can help in locating the presence of a person of attention.

Fig 3: Tracking Faces in Images and Videos

Limiting faces and finding their identities are crucial problems as face appearance gets changed with many factors like makeup, aging, different poses, different lightning situations.

Identifying facial key points is not an easy task as features of a face varies a lot from one human to another. Factors such as lightening, emotion, pose, size makes a lot of change even on the same person's face. My target is to build an efficient and accurate algorithm to detect key points that are located on facial images.

2.2. Related Studies:

Many other researchers have attempted key point recognition on facial images, the work here is unique in that it makes use of deep learning. The solution is designed using libraries of Python programming language and neural networks on Amazon EC2 GPU instance. I have tried to find an efficient and accurate solution. I have read few published papers that have helped enormously as they have analysed and examined several approaches out of which I can leverage some and some parts of an approach.

Through my research, I have placed effort to design a model which can identify key points on facial images in a meticulous and adequate way.

2.2.1. Method 1: Prediction using active shape model, r-part decision tree and neural network

In this method, for preparation of data, Jun[9] divided key point locations in to strings of image pixels and target variables. Later this strings were converted in to matrixes, which were anticipated as a 96*96 pixels sized photo.

Fig 4: Greyscale image showing keypoints

2.2.1.1. Use of Active Shape Model:

This model obtains training from contours, this contours are obtained from training images with manual drawings. Principal of Component Analysis is used to get principal variations in the data used for training. This model can identify if a contour is good fit or not on the basis of this variation in data. In order for making correction in search step in the positions, Active shape model leverages on matrices illustrating line textures that are horizontal to a point of control.

Fig 5: Active shape model training from contours

Crippling of a basic contour takes place after Active shape model is built. It happens with acquisition of finest texture match for points of control. In this constant process, movements of points of control are governed by this model identification of natural object contour from data used for training. This process occurs in two stages: Train and Search

Train:

Lets consider there is an image named i and it has m number of keypoints

$$\mathbf{X}_i = (x_1^i, y_1^i, x_2^i, y_2^i, \dots, x_m^i, y_m^i), i = 1, 2, \dots, n.$$

(x_j^i, y_j^i) serve as j^{th} keypoint of i^{th} image. Sample taken for training has size n .

a. Adjusting the data used in train process:

This step is adopted in order to make data set comparable and correlative

Reduction:

$$D = \sum |\mathbf{X}_i - \bar{\mathbf{X}}|^2$$

If we consider two shapes, Lets call them X1, X2 then they are represented in below way:

$$\mathbf{X}_1 = (x_{11}, y_{11}, x_{12}, y_{12}, \dots, x_{1m}, y_{1m})^T$$

$$\mathbf{X}_2 = (x_{21}, y_{21}, x_{22}, y_{22}, \dots, x_{2m}, y_{2m})^T$$

Trying to adjust X2 according to X1. Following conversion, whirl and orbit. To reduce

$$E = [\mathbf{X}_1 - M[s, \theta](\mathbf{X}_2) - \mathbf{t}]^T \mathbf{W} [\mathbf{X}_1 - M[s, \theta](\mathbf{X}_2) - \mathbf{t}]$$

$M[s, \theta](\mathbf{X})$: scope X with s and make it revolve with an angle Θ

This conversion can be shown by $\mathbf{t} = (t_x, t_y)$:

A diagonal matrix that has weights associated with it is represented by $\mathbf{W} = \text{diag}(w_1, \dots, w_m)$:

If a key point is balanced, It represents it has very less movement in comparison to other points

Rkli : It is a measure of distance in i^{th} shape, this distance is between k^{th} and i^{th} point

VRkl : It is a measure of Rkli variance in the dataset used for training purpose

$$w_k = \left(\sum_{i=1}^n V_{R_{kl}} \right)^{-1}$$

After this step. A constant method to adjust training data according to mean of shapes is used

b. Leveraging Principal of Component Analysis for representing variations in shapes

1. In different shapes, mean is calculated by :

$$\bar{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^n \mathbf{X}_i$$

2. In different shapes, covariance is calculated by:

$$\mathbf{S} = \frac{1}{n-1} \sum_{i=1}^n (\mathbf{X}_i - \bar{\mathbf{X}})(\mathbf{X}_i - \bar{\mathbf{X}})^T$$

3. Eigen values involved are calculated by

$$S = \frac{\sum_{i=1}^t \lambda_i}{\sum_{i=1}^m \lambda_i} > 0.95 V_T$$

c. Train Process

This alliance can be used to represent every shape used in train process:

$$\mathbf{X}_i \approx \bar{\mathbf{X}} + \mathbf{P}\mathbf{b}$$

Specifications of shape are denoted by:

$$\mathbf{P} = (p_1, \dots, p_t):$$

$\mathbf{b} = \mathbf{P}^T(\mathbf{X}_i - \bar{\mathbf{X}})$: set of t variables making a vector, that characterize criterion of the model

d. Search Process

Mean of the shapes has been used as starting nearby value by Jun[9]. This search process performs with selection of shape specifications $\mathbf{b}$ for the used model. A co-ordinate frame that is centred on an object is used to characterize the object's shape. They Jun[9] built an model's instance named $\mathbf{X}$ by defining location, direction and size in the frame of image. The constant method that they Jun[9] used to boost the aptness of $\mathbf{X}$, the instance to the image is described below:

An image's portion is scrutinized over every point to derive best possible nearby point's analogue

In order to increase the aptness of newly found points following variables are modified:

b, s, t, Θ

This process is continued until the concurrence starts happening.

Outcomes:

In the model described above, shapes are unstable when we have root mean square error values greater than 6. Even this model approximately picks the image contours. It has been observed that if there are more points the result obtained will be more accurate.

Fig 6: Identification by ASM approach

2.2.1.2. Recursive partitioning decision tree approach:

A decision tree is the basis of this approach. They Jun[9] made use of images used for training for the purpose of training and making predictions for photos that do not have any key points location. This was done according to different variables categorizing the members of the populace. They calculated eigen values of photos to make them fitted into recursive partition decision tree and later found out an analogous relationship between the feature matrix and

every photo. On the basis of eigen values, Every photo has been categorize by the decision tree. After that from every group in set used for training. They Jun [9] draw average values of locations of key points and committed this values of locations to groups in set used for testing.

Fig 7 : Identification by R-part decision tree approach

Aberration:

Firstly they anticipated left eye centre, the results were good. Later they resorted to calculation of RMSE for each key point to be detected. The observed RMSE values for x and y coordinates were 3.61751 and 3.14066. Both values were decent. After conducting tests, They Jun[9] made observation that method of categorization works well with decision tree for face detection. The below table shows RMSE values of key points anticipated by this approach

LEFT_EYE_CENTER_X	RIGHT_EYE_CENTER_X	LEFT_EYE_INNER_CORNER_X
3.617515	3.45202	4.322909
left_eye_outer_corner_x	right_eye_inner_corner_x	right_eye_outer_corner_x
5.082524	3.631528	4.063313
left_eyebrow_inner_end_x	left_eyebrow_outer_end_x	right_eyebrow_inner_end_x
5.069073	6.275277	4.47991
right_eyebrow_outer_end_x	nose_tip_x	mouth_left_corner_x
5.032551	6.650454	5.721183
mouth_right_corner_x	mouth_center_top_lip_x	mouth_center_bottom_lip_x
5.638065	5.105302	5.377807

LEFT_EYE_CENTER_Y	RIGHT_EYE_CENTER_Y	LEFT_EYE_INNER_CORNER_Y
3.140662	4.027928	3.249608
left_eye_outer_corner_y	right_eye_inner_corner_y	right_eye_outer_corner_y
3.625195	3.165341	3.776114
left_eyebrow_inner_end_y	left_eyebrow_outer_end_y	right_eyebrow_inner_end_y
3.695398	4.885833	4.166424
right_eyebrow_outer_end_y	nose_tip_y	mouth_left_corner_y
5.030088	5.641408	5.286611
mouth_right_corner_y	mouth_center_top_lip_y	mouth_center_bottom_lip_y
5.252617	7.289094	6.19728

Fig 8: RMSE values of key point predicted by R-part decision tree approach

In their Jun[9] approach, There was larger error appearing in anticipating mouth centers, whereas smaller error was observed in anticipation of eye corners and eye centers.

2.2.1.3. Using Neural Network for key point detection

They Jun[9] tried building the model using a neural net with a hidden layer

$$\begin{aligned}
 Z_m &= \sigma(\alpha_0 m + \alpha_m^T X), m = 1, 2, \dots, M \\
 T_k &= \beta_{0k} + \beta_k^T Z, k = 1, \dots, K \\
 f_k(X) &= g_k(T), k = 1, \dots, K
 \end{aligned}$$

There is use of a function for activation in this model. It is denoted by

$$\text{sigmoid}(v) = \frac{1}{1 + e^{-v}}$$

Train Process

They Jun[9] aimed at lowering the cost parameter during train process. Cost parameter was calculated using cost function and this cost function is calculated as squared errors sum.

$$R(\theta) = \sum_{k=1}^K \sum_{i=1}^N (y_{ik} - f_k(x_i))^2$$

Use of SGD

According to the suitability to their model, They Jun[9] made use of chain rule for the purpose of calculating gradient and depended on gradient to compute the minimum. SGD stands for Stochastic Gradient Descent and they Jun[9] used it for training their model.

Outcomes:

The results obtained were not very impressive. They managed to get RMSE value of 4.03645

Fig 9: Identification by R-part decision tree

After this, They started making attempts to improve the accuracy of the model. They made efforts to change learning rate

$$\beta_{t+1} = \beta_t + \Delta\beta_t$$

$$\Delta\beta_t = -\eta g_t$$

Alignment of learning rate:

LEARNING RATE	TRAIN OBJECTIVE	VALID OBJECTIVE
0.1	38245.73	186372.25
0.01	197.96	138.23
0.001	196.21	137.64
0.0001	194.87	134.20
0.00001	197.37	135.38

Fig 10: Learning rate in R-part decision tree approach

After conducting some experiments, they came to know that the neural network with one layer that they are using is not suitable for their purpose. They made efforts to resolve this issue by inserting one more layer in the middle. Later after conducting experiments, they noticed that their Jun[9] model started to learn and RMSE value of predictions became 3.93669

They opted for reduction of dimensions and used denoising encoder to boost the results.

Outcomes:

They Jun[9] managed to reduce the value of RMSE up to 3.8137

Fig 11: Improved results in R part decision tree approach

Fig 12: Training and Validation curve in R-part decision tree approach

2.2.2. Method 2: Combined use of neural network and machine learning for building model

Identification of all the important key points on a facial image based on greyscale is the chief target of this approach. For the purpose of training and testing of the model, They Akrita[11] made use of dataset consisting of seven thousand forty nine images. Test dataset they Akrita[11] selected to work on consists of one thousand seven hundred eighty three images. For implementation of the model, They Akrita[11] used R programming language.

a. Setup of the data used for implementation

The data set used by them Akrita[11] consists of images on greyscale. In greyscale images, zero denotes white and number two hundred fifty five denotes pure black color. Every facial keypoints like corners and centers of the eyes, nose, lips, eyebrows etc. are characterized on each training image with coordinates of x and y axis.

b. Characteristics of images used in model:

Below is a sample image used in this model

Fig 13 : Sample Greyscale Image

c. Assessments of anticipations made by model:

In order to test their Akrita[11] model, They Akrita[11] tried contrasting the results obtained in model and computed root mean square error value of outcomes. In their implementation, they computed the mean of locations of every key point. Later they Akrita[11] enforced this mean values as solution to the test images.

Fig 14: Figure showing mean right eye patch and mean nose tip patch used in approach

For the purpose of increasing the accuracy of model. They Akrita[11] rather than considering only mean, they Akrita[11] focused on considering an image patch nearby the key points. Following this made easy for them Akrita[11] to generalize discern outputs for images.

d. Assessment of patches computation approach

They Akrita [11] tested the model for various patch sizes in the range of ten to fifteen. Everytime outcome was different. They Akrita[11] were able to get a RMSE value of 3.7554 with patch size of 14. Later, they used classification method based on neural networks. As the dataset was huge, they Akrita[11] decided to lower the size of images and taken only half of the training set.

Fig 15: Neural Network Created in the approach

e. Assessment of adding neural networks to the model

Following graph represents changes in RMSE values with increase in number of hidden layers added to network

Fig 16: RMSE values with respect to number of hidden layers

Closure:

The trend observed in the root mean squares error values by number of layers suggests that data characteristics are depended on each another. It is observed that faces generally follow a specific pattern so this interdependency in data should be taken in to account. Identification of key points on face is the first step leading to more complex possible models like predicting meanings out of different facial expressions.

CHAPTER 3

3. Prospective Procedure

As indicated in the introduction, I am using an approach that is combination of some approaches. The baseline in my approach is use of convolutional neural networks and amazon EC2 GPU instance for the purpose of key point detection on facial images.

Kaggle website has helped me with the idea of the project and dataset that I have used for implementation. The dataset used comprises of seven thousand forty nine images on grey scale with x and y coordinates of fifteen facial keypoints. Pixel matrix of size ninety six by ninety six is used to symbolize every image. The entry in this matrix consists of a value between number zero and two hundred fifty five, featuring the intensity of every pixel. The training dataset used for the purpose of training the model is a matrix of size seven thousand forty nine by thirty one, here one image is represented by one row. The coordinates of key point locations on x and y axis of all fifteen facial keypoints are denoted by initial thirty columns. Last column in the dataset contains entry consists of list of nine thousand two hundred sixteen numbers that are symbolizing pixel matrix of all images. I have also observed one thing that in some of the entries, locations of key points are missing.

For the implementation of the model, I have used Python programming language, Theano Theano[14], Lasagne Lasagne[12] and Amazon EC2 GPU instance.

3.1. Python Programming Language:

It is one of the most generally used, powerful, transparent, progressive programming language. It allows user to write bigger functionalities code with few number of lines. It's design emphasis a lot on readability of code. It provides constructs that helps in building understandable bigger and smaller scale programs. Python favours multiple programming and provides automated management of memory. It also has a huge and complete library:

The main doctrine on which python is based according to The Zen of Python (PEP 20) is epitomize as below:

- Clean and organised things look far better than untidy things
- Clearly stated things are easy to understand than unclear things
- Easy and uncomplicated things are always preferable over convoluted things
- It is easy to work with bigger and clear structure than tricky and convoluted ones
- Clarity is very important. It makes thing easy to understand

3.2. Lasagne for neural networks:

For the purpose of building and training neural networks in Theano, Lasagne is used. It is a library that is light weight. It's characteristics are as follows:

- It encourages use of networks that follows feed-forward mechanism
- It encourages architectures consisting of multiple inputs and outputs
- It supports various escalations methods like RMSprop etc.
- In lasagne, its very easy to define cost function
- There exists clear and great support of Graphics Processing Unit and Central Processing Unit because of Theano's compiler

According to Lasagne[12], Lasagne is designed while keeping following points in mind:

- Clarity: It is easily understandable, easily expandable and encourage research
- Pellucidity: It is transparent, It supports direct processing and Theano is not hided after abstraction
- Independent components: Any component can be used separately in Lasagne
- Realistic: Lasagne makes come tasks & actions easy achievable
- Moderation: It never hinders any users with the characteristic they don't want to use
- Target oriented: It is focused. Rather than doing many things it aims at doing a single thing in a good way

3.3. Theano for numerical computation:

Theano is a python library that is available as open source. Basically it is designed for performing numerical computations. It can be used with both Graphical Processing Unit or Central Processing Unit. It can perform complex mathematical computations easily and accurately. It has following characteristics:

- It works very well with Graphical Processing Unit. It can easily perform complex calculations at very fast rate
- It performs the computation of expressions at a faster rate due to dynamic C code that gets developed
- It makes use of good and efficient testing techniques that help in detecting and analysing errors
- It is very fast and stable. It can easily compute the logarithmic function of a very small number

3.4. AWS EC2 GPU Instance for faster computing:

In the implementation of the model, I have used EC2 GPU instance which worked with NVIDIA GPUs and provided very fast processing speed and good amount of memory. Basically Graphical Processing Unit are used to boost up many complex engineering applications by making use of Open computing language or CUDA.

CHAPTER 4

4. Model Implementation

4.1. Description of method used

For implementation of solution, I have used amazon ec2 gpu instance and convolutional networks consisting of multiple levels. Outputs of multiple networks are fused at every level for accurate and robust evaluation. At the stage of initialization, high level features are extracted from the facial region which helps in locating key points with high accuracy. Local minimum occurred by data corruption and ambiguity in difficult samples of image caused by extreme lightings and large variations in poses can be avoided by this method. At later levels, training of networks is adopted to locally refine the initial predictions and the input supplied to them are limited to smaller regions around the predictions that are obtained in initial stage.

4.2. Description of data used

The data that is used for training the model comprises of seven thousand forty nine images that are in greyscale. The size of image is ninety six by ninety six. It is assumed that learning will takes place to find the correct locations of fifteen keypoints like left eye center, right eye outer corner, mouth center bottom lip etc. There is an interesting twist with data, that is we have only two thousand labels for some images, while we do have more than seven thousand labels for some images.

4.3. Flowchart explaining the procedure adopted for building model

Fig 17: Flowchart explaining the proposed approach

4.4. Comparison of method used with other methods

Proposed Approach	Active shape model Jun[9]	R part decision tree Jun[9]	Using Machine Learning and Neural Network Akrita[11]
<p>The proposed approach is making used of convolutional neural networks. The implementation makes use of Amazon EC2 GPU instance, Python, Theano, Lasagne.</p>	<p>This model obtains training from contours, this contours are obtained from training images with manual drawings. Principal of Component Analysis is used to get principal variations in the data used for training.</p>	<p>A decision tree is the basis of this approach. They Jun[9] made use of images for purpose of training and making predictions for photos that do not have any key points location. On the basis of eigen values, Every photo has been categorize by the decision tree.</p>	<p>In their approach, they computed the mean of locations of every key point. Later they Akrita[11] enforced this mean values as solution to the test images. For the purpose of increasing the accuracy of model. They Akrita[11] rather than considering only mean, they Akrita[11] focused on considering an</p>

			image patch nearby the key points.
--	--	--	---

Table 1: Comparison of proposed approach with studied approaches

4.5. Steps involved in the implementation of model:

Step 1: Environment Setup for developing model: Following are the steps involved in environment setup:-

- Created AWS EC2 GPU instance (g2.2xlarge) with Ubuntu 14.04.1 LTS. Configured the SSD storage to be 30G.
- Upgraded everything and Installed numerical libraries with setting environment variables.
- Installed all the required dependencies.
- Installed bleeding-edge version of Theano.
- Installed Latest CUDA toolkit.
- Added CUDA package, installed CUDA driver and updated the path to include nvcc and Id_library_path.
- Installed samples and tested CUDA.
- Configured the Theano to use GPU by default and tested Theano.
- Installed Lasagne.

Step 2: Written a python code that loads the data from dataset we have.

There is a function created that can load both training and test data. The main difference between two dataset is that we don't have target values ie. key point locations in the test data

Step 3: Developing an initial model that consists of a hidden layer

After loading of data, Next step is to build a neural network. For that purpose, Lasagne is used.

A neural network is created using lasagne which consists of a hidden layer.

Step 4: Testing of the created model

In this step, The network that is build in the previous step is tested against the test data

Step 5: Building another model using convolutions

In this step, LeNet5 style convolutional neural network is used. These networks are one of advancements of deep learning. There is difference between fully connected layers and convolutional layers, and this difference is that convolutional layers use some arrangements to reduce the number of variables that needs to get learning, These arrangements are as follows:

- Sharing of weight is adopted among sub parts of neurons in the layer added for implementing convolutions
- Neurons in the layer are only connected to sub group of neurons that are located in the layer came before it.
- Inputs are run through subsampling
- I have used a convolutional neural network with two layer that are fully connected and three layers used for convolutions. The number of filters used initially are thirty two, This numbers gets doubled with every convolutional layer.

Step 6: Implementing data augmentation

By adding more training data an overfitting network can be made to perform more efficiently. In such cases where the networks that are not regularized and overfitting does not occur. Data augmentation can be helpful in increasing number of training examples by introducing noise applying transformations.

Step 7: Changing momentum and learning rate over course of time

We need to change both momentum and learning rate. As initially we were away from optimal values, it was good to move fast and learn fast. But when we are near to optimal values we need to get our speed lowered and move slowly

Step 8: Implementing Dropout

It is one of the common regularization technique that is used to improve the efficiency of a neural network. Firstly, training should be done properly and overfitting should happen. Then regularization comes into the picture.

Step 9: Training Specialists for increasing efficiency

For the purpose of improving the accuracy, Rather than training a single model, Training of few specialist model is a great idea. Each of the specialist model can be used to anticipate a single key point like right eye center etc. There are total of six specialist models trained for this approach.

Step 10: Adopting supervised pre-training

For the purpose of making the process of training the specialists faster, supervise pre-training is implemented. In this method, the specialist networks are initialised with weights that are already learned from the networks, rather than initialising them with random weights.

CHAPTER 5

5. Experimentation and Outcomes

The term that are used in results are explained below:

Epoch: One full training cycle on the training data set is known as epoch.

Training Loss: Training loss is the measurement of loss (Mean Square Error) on the training data set. The lower the value. The more accurate the model.

Validation Loss: Validation loss is the measurement of loss (Mean Square Error) on the validation data set. The lower the value. The more accurate the model.

Training Loss/Validation Loss: It is the ratio of Training loss and Validation loss. It is the measure that is used to determine how accurate the model is. Lower value is better

5.1. Outcomes:

5.1.1. After training according to the initial model: The following results were observed

Epochs	Training Loss	Validation Loss	Training Loss / Validation Loss
1	0.105420	0.031090	3.390800
2	0.020356	0.019298	1.054824
3	0.016120	0.016921	0.952662
.....			
398	0.002252	0.003263	0.690303
399	0.002246	0.003259	0.689168
400	0.002245	0.003256	0.689496

Table 2: Training Loss, Validation Loss and ratio of Training Loss and Validation Loss for the initial model

Train and Valid Loss for First Model

Fig 18: Training Loss and Validation Loss Curve for the initial model

After this step, the observed score is 2.738945.

Its calculated by multiplying valid loss observed in 400th epoch by 48

$$\text{Score} = (\text{Square root}(0.003256)) * 48 = 2.738945$$

Here we are multiplying the value by 48 because we have divided the value by 48 earlier to keep it in range of (-1,1).

5.1.2. Testing the model on some sample images

Tried testing the model on sample images and following is the result obtained

Fig 19: Prediction by model on some sample images

5.1.3. After adding convolutional neural networks, following are the observations obtained:

Epochs	Training Loss	Validation Loss	Training Loss / Validation Loss
1	0.111765	0.042742	2.614875
2	0.018501	0.009415	1.965055
3	0.008597	0.007920	1.085479

.....			
998	0.001081	0.001568	0.689413
999	0.001081	0.001568	0.689413
1000	0.001080	0.001567	0.689215

Table 3: Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after adding convolutional network

Fig 20: Training Loss and Validation Loss Curve after adding convolutional network

After this step, Score (RMSE) obtained = $(\text{squareroot}(0.001567)) * 48$
 $= 1.900096$

5.1.4. After implementing data augmentation, following are the observations obtained

Epochs	Training Loss	Validation Loss	Training Loss / Validation Loss
.....			
500	0.002239	0.002305	0.971366
.....			
1000	0.001366	0.001625	0.840615
1500	0.001068	0.001459	0.732008
2000	0.000896	0.001370	0.654014
2500	0.000762	0.001321	0.576835
3000	0.000679	0.001289	0.526764

Table 4: Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after implementing data augmentation

Fig 21: Training Loss and Validation Loss Curve after implementing data augmentation

After this step, Score (RMSE) obtained = $(\text{Squareroot}(0.001289)) * 48$
 $= 1.723327$

5.1.5. After changing momentum and learning rate over the course of time, following observations are obtained

For net 4:

Epochs	Training Loss	Validation Loss	Training Loss / Validation Loss
50	0.004218	0.003998	1.055027
100	0.003534	0.003385	1.044017
250	0.001558	0.001783	0.873808
500	0.000916	0.001435	0.638327
750	0.000654	0.001357	0.481945
1000	0.000497	0.001389	0.357811

Table 5: Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after changing learning rate and momentum for net 4

Train and Valid Loss for net4

Fig 22: Training Loss and Validation Loss Curve after changing learning rate and momentum for net4

5.1.6. For net5:

Epochs	Training Loss	Validation Loss	Training Loss / Validation Loss
50	0.004319	0.004083	1.057800
100	0.003757	0.003538	1.061899
250	0.001766	0.001846	0.956663
500	0.001137	0.001439	0.790132
750	0.000879	0.001316	0.667933
1000	0.000707	0.001262	0.560221
1500	0.000494	0.001198	0.412353
2000	0.000374	0.001185	0.315611

Table 6: Training Loss, Validation Loss and ratio of Training Loss and Validation Loss after changing learning rate and momentum for net 5

Fig 23: Training Loss and Validation Loss Curve after changing learning rate and momentum for net 5

5.1.7. After implementing dropout, Following observations are obtained

Epochs	Training Loss	Validation Loss	Training Loss / Validation Loss
50	0.004757	0.007045	0.675230
100	0.004442	0.005323	0.834491
250	0.003975	0.003930	1.011450
500	0.002575	0.002349	1.096211
1000	0.001863	0.001615	1.153560
1500	0.001559	0.001373	1.135469
2000	0.001410	0.001231	1.145410
2500	0.001297	0.001147	1.130775
3000	0.001196	0.001088	1.099264

Table 7: Train Loss, Valid Loss and ratio of Train Loss and Valid Loss after adding dropout

Train and Valid Loss after adding dropout

Fig 24: Train Loss and Valid Loss Curve after adding dropout

- After implementing the training of six specialists models predicting eye center, nose tip, mouth, lips, eye inner and outer corners, eyebrow inner and outer end.

The score observed is 2.17 RMSE

- After implementing the supervised pre-training of six specialists model predicting eye center, nose tip, mouth, lips, eye inner and outer corners, eyebrow inner and outer end.
- The final score obtained in this system is 2.13 RMSE.

5.1.8. Comparison with different methods:

Method Name	Active shape model Jun[9]	R part decision tree Jun[9]	Using Machine Learning and Neural Network Akrita[11]	Approach Used for implementation
RMSE Value	>6	3.81	3.76	2.13

Table 8: Comparison of RMSE values of proposed approach with other studies approaches

5.1.9. Runtime Complexity:

- For the first model. The time taken to train the model was very less around 5 minutes.
- For convolutional neural network. The time taken was comparatively more. It took around 22 minutes for 1000 epochs. However the results were very good.
- After implementing data augmentation. The network took about an hour to train
- After adjusting learning rate and momentum, The training time got reduced to almost half of the time taken earlier
- After adding dropout, training again become slower

- For training specialist, Early stopping is adopted to reduce the time taken to train.
- Supervised pre-training of specialist model is adopted in order to make the training faster. In this process, training runs almost 4 times faster than previous step.

CHAPTER 6

6. Conclusion and Future Work

6.1. Conclusion

A facial key point detection model is implemented using machine learning in this project. The main contribution in this project lies in applying convolutional neural networks for facial key point detection and the main aim of the approach is to identify key points accurately and efficiently in an optimal amount of time. The approach makes use of Amazon EC2 GPU instance for performing complex computations fast. Python programming language is used for writing code. Lasagne and Theano (Python Libraries) are used for implementing a couple of neural architectures. Data augmentation is used to make overfitting network perform better. For the purpose of implementing regularization in order to improve neural networks, Dropout is used. In order to improve accuracy of model, instead of training a single model, some specialists model are trained with each one predicting a different set of target values. Supervised pre-training was adopted in order to make training this specialist faster. The results prove that the proposed approach performs significantly better than the other three studied approaches by providing lower RMSE value.

6.2. Future Work

In Future work, another fully connected layer or convolutional layer can be added to experiment with the existing model and its effect can be analyzed on the model performance. Experiments by changing dropout can be done and variation can be compared against the existing results. Parameters used for individual specialist networks can be optimized.

LIST OF REFERENCES

- [1] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A*, vol. 4(3) (1987)
- [2] Contrast Stretching : <http://homepages.inf.ed.ac.uk/rbf/HIPR2/stretch.htm> (16 May 2016)
- [3] Eigenface Tutorial. <http://www.pages.drexel.edu/sis26/Eigenface%20Tutorial.htm> (16 May 2016)
- [4] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning with Applications in R*. Springer (2013)
- [5] R. Herpers, M. Michaelis, K.-H. Lichtenauer, and G. Sommer - *Automatic Face and Gesture Recognition*, IEEE Computer Society (1996)
- [6] Kaggle Online Competetion: Facial Key points Detection. <https://www.kaggle.com/c/facial-key-points-detection> (16 May 2016)
- [7] Q. Wang and J. Yang. Eye Location and Eye State Detection in Facial Images with Unconstrained Background. *Journal of Information and Computing Science*, vol. 1(5) (2006)
- [8] D. Ciresan, U. Meier, and J. Schmidhuber. Multi column deep neural networks for image classification. In *Proc. CVPR* (2012)
- [9] Jun Zhao, Renjie Chen, Rongdong Wu, Xinyu Li, Liangquan Zhou, Facial key point detections Using Active Shape Model, R-part Decision Tree and Neural Network http://rstudio-pubs-static.s3.amazonaws.com/46778_79bacb2c6375410fbc0d2b73bfc0ae4d.html#1 (16 May 2016)
- [10] Eigenfaces for Dummies. <http://jmcspot.com/Eigenface/Default> (16 May 2016)
- [11] Akrita Agarwal, Srivathsava Sista Facial key point recognition <http://www.slideshare.net/AkritaAgarwal/facial-key-point-recognition> (16 May 2016)
- [12] Lasagne <https://github.com/Lasagne/Lasagne/> (16 May 2016)
- [13] Python <https://en.wikipedia.org/wiki/Python> (16 May 2016)
- [14] Theano <http://deeplearning.net/software/theano/> (16 May 2016)
- [15] Convolutional Neural Networks <http://danielnouri.org/> (16 May 2016)
- [16] Sun, Y., Wang, X., & Tang, X.. Deep convolutional network cascade for facial point detection. In: *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)* (2013)
- [17] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML Workshop on Deep Learning for Audio, Speech, and Language Processing (WDLASL 2013)*
- [18] Amazon EC2 GPU Instance <http://docs.aws.amazon.com/AWSEC2> (16 May 2016)