International Journal of Future Generation Communication and Networking
Vol. 13, No. 3, (2020), pp. 1222–1233

Facial Keypoints Detection Using CNN And Autoencoders

Ankit Sinha[1], Akashdeep Dhar[2], Mahij Momin[3], , Mohit Gurav4, Sheetal Kumar Jain[5]

Department of Computer Engineering and Technology, MIT Academy of Engineering

ABSTRACT

Facial keypoints detection is an essential task in computer vision and augmented reality domains. We put forth a novel solution for this problem which different than existing techniques. We implement a Convolutional Neural Network-based model with Autoencoders to detect facial keypoints. The deep structure of Convolutional Neural Networks (CNN) enables extraction of high-level features and gives more accuracy while locating each key point. Convolutional networks are shaped to predict all the points simultaneously. With autoencoders it is possible to boost the performance of the model. Autoencoders are data-specific and can work effectively if similar data was used while training.

Keywords:

Facial keypoints detection, computer vision, augmented reality, Convolutional Neural Networks, Autoencoders.

1. INTRODUCTION

Facial key point detection is a task of predicting keypoints on the human face. This Method is used for solving diverse facial-analysis-problems, e.g. face recognition, face morphing. Some other applications include tracking faces in images and videos, analyzing facial expression, biometrics and with the advent of Augmented Reality the superimposition of filters realistically. In recent years, significant research has been done on this problem and various model have been developed for accurately locating the facial keypoints. This problem is especially challenging when face images are taken with extreme angles, lightings, expressions and occlusions.

Among the various approaches cascaded regression methods[1], combining model-driven and data-driven approach for facial keypoints detection[2] and the use of Deep Convolutional Network Cascade (CNN) for facial keypoints[3] detection are noteworthy methods which have shown ability to efficiently and accurately localize the facial keypoints even in challenging scenarios. In cascaded regression methods at each cascading stage visual features extraction is done by the current predicted keypoints are then they are updated via regression from the extracted features; and these points are used for regression in the next stage. And the previous information is lost. The merits of the combination of data-driven and model-driven approach is that it can be used for getting a fine-tuned output as it is amalgamation of several techniques. Convolutional Neural Network cascading involves first making accurate predictions at first level, the network takes full face as input to make the best use of texture context information and more global features are extracted in successive layers. Convolutional Neural Networks can be trained so that it can predict all the keypoints simultaneously. The last few levels of CNN are used to refine the estimation of keypoints.

Convolutional Neural Networks and Deep Learning models have been successfully implemented in computer vision tasks. In this paper, we study a Deep Learning based model comprising CNN to identify the facial keypoints accurately. Apart from being less complex than existing methods it is also easy to implement and is adequate for most of the applications. We try to study different aspects of a CNN based keypoints detector and how it can be improved to give better results.

ISSN: 2233-7857 IJFGCN

1.1 Computer Vision

Computer vision is an inter-domain area that deals with a high-level understanding of digital image or video with the perspective of engineering. It helps to automate numerous tasks in different domains that are done by the human optic system i.e. Eyes. The start of computer vision date back to the 1960s where Artificial intelligence was blooming and a desire to see through a computer. Computer Vision is vaguely divided into various tasks like Acquiring data, Processing of data, Analyzing the data. The data is acquired by a sensor like a camera or a webcam which help a system to get images or video depending on the purpose. After the data is taken the features or key points are detected for the particular task and are obtained for analysis.

In modern days Computer Vision has become very popular and is been applied to various fields. But then too there is still a lot of scopes left in it to explore, Computer Vision does not deal with the image but other things related to vision, which are motion capture, edge detection and 3 dimensional reconstruction of multiple images, etc. Computer vision work on seeing thing therefore the data is a reference to image or video which are visual data. In the low-level video are nothing but a stack or array of images that are played one by one to give motion. The higher the frames rate the finer the video is and video size also increases.
e.g. A video of 60FPS and 1 Min long we be represented as 60(Images)*60(Sec) = 3600 images.

Images are sorted in multi or single dimension array depending on the color of the image and size of an image. Therefore, if we take an example of an image. It will be represented as 96*128*3:

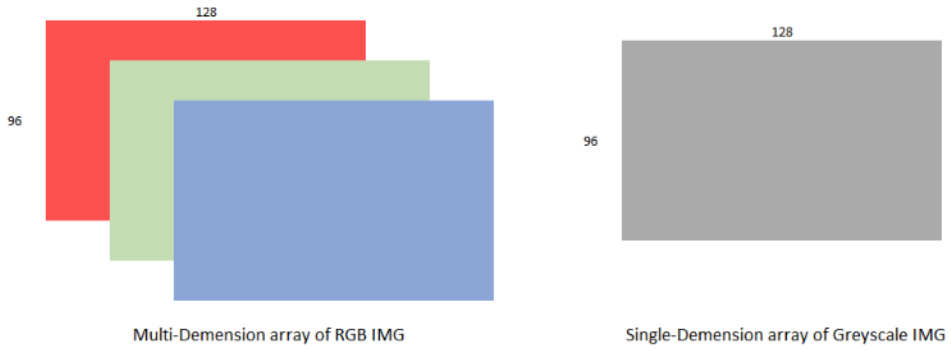

Figure 1: Representation of an image using arrays

1.2 Artificial Neural Network

Biological neurons are brain cells which take electrical signals as an input, do some processing on the signal and then send it to the next neuron.
Artificial neurons are less complicated, simplified versions of biological neurons. They try to mimic the ability of biological neurons to process information and have a similar structure. These artificial neurons when connected in different possible ways give “Artificial Neural Networks” (ANN). There can be multiple inputs for each neuron. Each input is multiplied with some weight and these products are added. The resulting value is passed through an “Activation function” and is converted into output. This output can again be given to another neuron as an input or it can be our final output depending on the architecture of the neural network. Each artificial neural network (ANN) has an input layer, some hidden layers and an output layer. Neural networks have become extremely popular because of their performance in supervised-machine-learning. One of the salient features of neural networks is that they can deal with structured as well as unstructured data.

ISSN: 2233-7857 IJFGCN

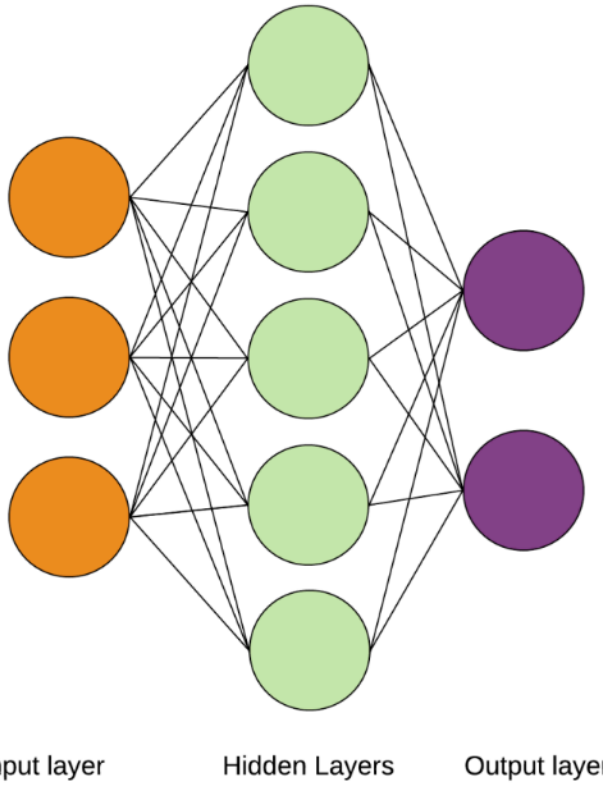

Figure 2: A visual representation of Artificial Neural Network (ANN)

Each input let's say xi is multiplied with the weight of the connecting edge wi. For all the incoming edges the sum of such products is taken and it is passed through an activation function. Hence, the output value of the neuron is also called the activation value. The weights are model parameters that are not set manually by the practitioner. It is possible to estimate the values by using the data. These values are saved as part of the trained model. Various optimization algorithms are used for estimating parameters. Some other examples of parameters include support vectors in an SVM i.e. support vector machine and coefficients of linear or logistic regression. On the contrary, a model hyperparameter is configuration that cannot be estimated by using existing data. They have to be defined by the practitioner manually or can be set using heuristics. Hyperparameters can be used to estimate model parameters.
The practitioner has to make the choice of activation function at each hidden layer and also at the final output layer. Some common activation functions are sigmoid function, tanh function, Relu function and Leaky Relu function. Relu is a de facto standard activation function used blindly by the practitioners in their models. Although other activation functions like sigmoid and tanh are also used in some applications. Tanh function performs better than sigmoid function in almost all cases except for binary classification where the expected range of output is between 0 and 1. The functions can be visualized as:

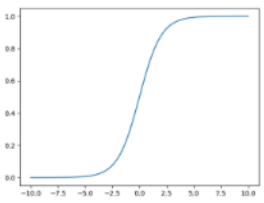

Figure 3: Sigmoid

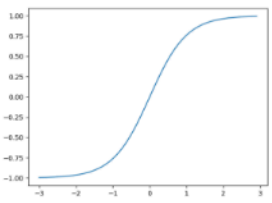

Figure 4: Tanh

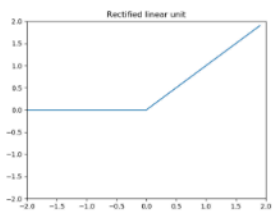

Figure 5: Relu

ISSN: 2233-7857 IJFGCN

1.3 Convolutional Neural Network

Convolutional Neural Network is a type of Artificial Neural Network which is used for extracting features and recognizing objects from an image. Convolutional Neural Networks are different than regular Neural Networks as the layers are organized in three dimensions, viz. Width, height and width. Also, the neurons in a layer are not connected to all the neurons in the next layer unlike regular Artificial Neural Networks. The final output is a vector containing probability scores. It mainly comprises two components: Hidden layers and the output layer or classification part. Hidden layers are the part where feature extraction happens with a series of convolutions and pooling layers. In the classification part, fully connected layers classify images on bases of extracted features and give final output vector containing probability values.

1.3.1 Convolutional Layer

Convolution operation involves element wise product of image matrix and convolution filter or kernel. Each element of a filter is multiplied with all the possible sections of images having the same dimensions as the filter across all the color channels. The filter is slid over the input, at every location element wise product is taken and the result is added to a feature map which is in turn an input for next convolutional layer or fully connected layer. Just like Artificial Neural Network, activation function is used to make the output non-linear. Convolutional operations allow back-propagation to learn filters to identify the features in an image instead of having them hard coded by the practitioner.
After each convolution operation on a matrix of size n*n, with a filter of size f*f we get a resultant matrix of size n-f+1 * n-f+1.

As we go on doing the convolution operation the size of the image reduces significantly. Another problem with the basic convolution operation is that the pixels in the corners are considered a very less time because they are overlapped very few times during the computation. Original input size of the image can be preserved by padding the original image. The output matrix has dimensions n+2p-f+1 * n+2p-f+1 where p is the number of pixels padded on each side of the image. "Valid convolutions" does not involve padding whereas "Same convolutions" involve padding so that output size is the same as the input size.

$$n+2p-f+1 = n$$
$$p = (f-1)/2$$

If the model is overfitting and we require to reduce layers in our convolutional neural network, stridden convolutions are used. Instead of sliding the filter by 1 step, it is slid by a specific number so we get the image in the desired form. The resultant matrix after padding and striding convolution operation has dimensions (n+2p-f/s) + 1 * (n+2p-f/s) + 1. The fraction is converted to an integer by using a floor function:

$$f(x) = [x]$$

1.3.2 Pooling Layer

The image is divided into smaller parts and the pixel with the maximum value is taken from each of the parts. The number of channels remains the same in the pooling layer. Average pooling is used very rarely in Convolutional Neural Networks. Pooling involves no parameters but has hyper-parameters like f: filter size and s: stride which can't be learned and have to be defined by the practitioner to get the image in desired size. The resultant image is of the size $(n_h-f)/s+1 * (n_w-f)/s+1 * n_c$ where n_h is the height of the image, n_w is the width of the image, f is the size of filter, s is the stride value and n_c is the number of color channels.

ISSN: 2233-7857 IJFGCN

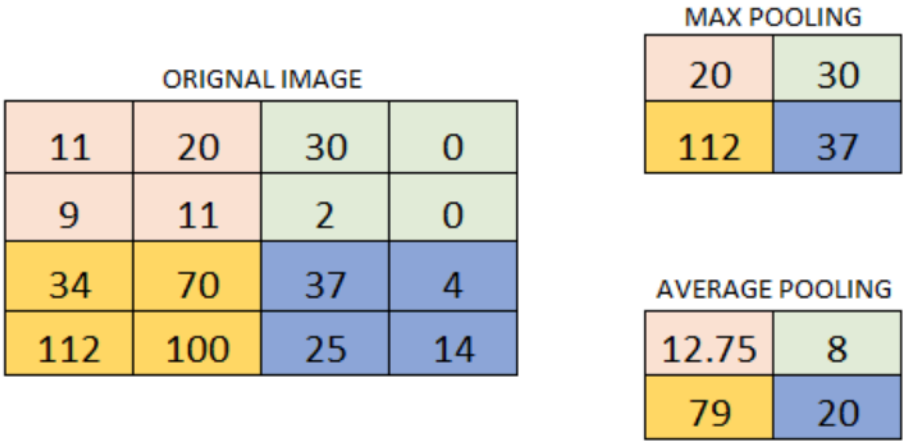

Figure 6: Max Pooling and Average Pooling

1.3.3 Advantages of convolutional layers over fully connected layers

A feature detector which is useful for a part of the image is probably useful in another part of the image. This is called parameter sharing. Convolutional neural networks with the help of such detectors reduce time and parameters required to be trained. Each layer, each output value depends only on a small number of inputs instead of the complete input. These two mechanisms allow convolutional neural networks to be trained with smaller datasets. Since it is possible to train them using small datasets, they are less prone to overfitting.

1.4 Autoencoders

Auto Encoders are used for reducing the dimensionality of data or removing data noise for high performance. Autoencoder works on both linear or nonlinear data with fast performance concerning Principal component analysis [4]. Autoencoder is a feed-forward Neural network that is trained in an unsupervised manner for efficient data coding and feature learning. Autoencoder is used in a lot of areas like in reconstruction of the face in computer graphics, removing watermarks, removing blurs from an image, and a lot more. We are using autoencoders to reduce the dimensionality of our data set. It is done by modifying the complex data dimension into a simple data dimension and with that boosting the training of our neural networks for facial keypoints detection. As we know autoencoder works on reducing the dimensionality, it also has a decoder which reconstructs the reduced data to almost original data. We have used 'binary cross-entropy' for our loss function of the auto-encoder. Binary cross-entropy is used for comparing one to one pixel of the original image and reconstructed image [5].

ISSN: 2233-7857 IJFGCN

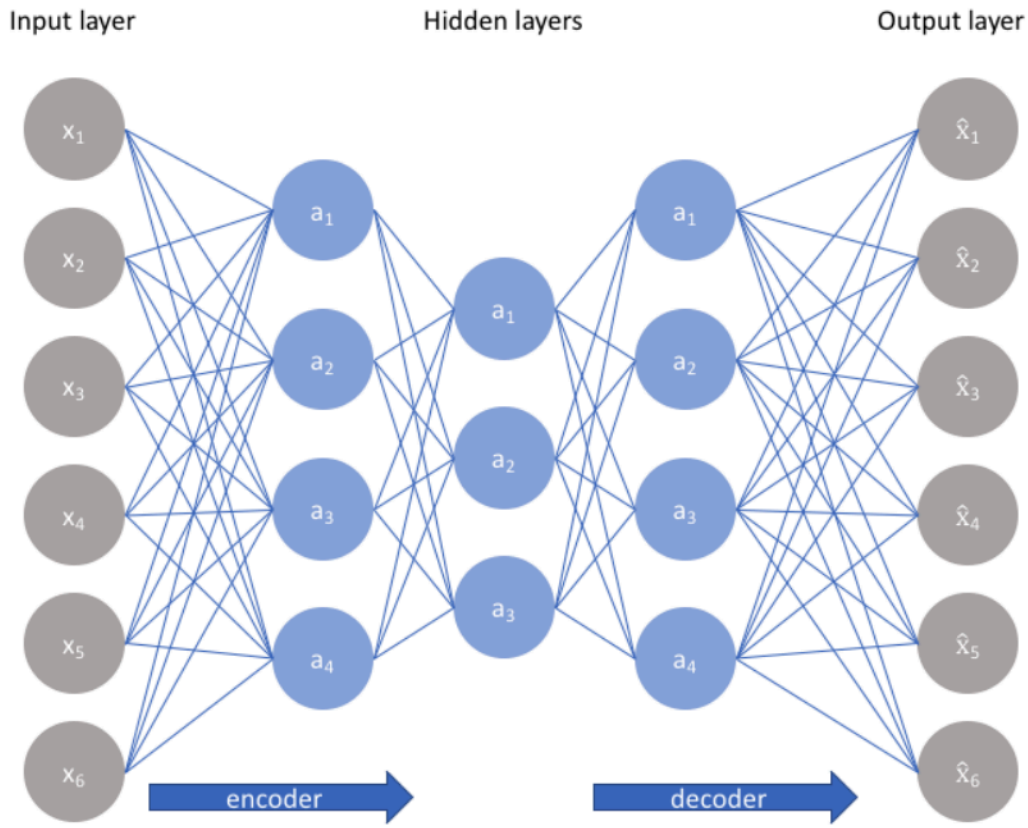

Figure 7: Autoencoders

2. LITERATURE REVIEW

Facial keypoints detection has innumerable applications in various fields. Face detection based on Cascaded Convolutional Networks[1], combining Data-driven and Model-driven methods[2] are the latest research being done in the field of facial keypoints detection along with Deep Recurrent Regression for facial landmark detection[3]. The facial keypoints detection has been studied extensively over the years[6,7,8,9,10]. Classifying search windows[6,7,8,9,10] and directly predicting key points positions[11,12,13]. For the first category, a classifier is trained for each key point and decisions are taken based on local regions. In comparison directly predicting key points positions is more efficient since it does not need scanning. Regressors are usually used as the predictor, based on limited patches resembling the facial point [13], or the whole image region[12]. Spatial constraints can also be added to regressors[12,13].

One of the recent approaches includes part-based methods[14], Part based models give refined results by assembling the outputs form part models. Some part-based methods which typically produce a local model for each facial feature to recommend the update direction and use a shape model to regulate the result globally.

Many researchers have tackled the problem as a regression problem[15]. Generally, approaches [15] includes cascaded regressors to predict the coordinates of key points directly from shape-indexed features. Recently some methods like [16] developing strategies like coarse-to-fine prediction and global-to-local regression in order to capture information at different scales. Predicting density depicting heat-maps via (Fully Convolutional Network) FCN-based networks is exploited for dual purpose of human pose estimation [17] and facial landmark detection.

[1] H. Lai et al., "Deep Recurrent Regression for Facial Landmark Detection,"

ISSN: 2233-7857 IJFGCN

The approach is used a deep convolutional and deconvolutional network followed by recurrent networks. The three parts have the following functionalities: Encoding an input face image via a deep network, estimating initial co-ordinates of the facial key point, Refining the co-ordinates of facial key points by a recurrent network. The model performed better than after combining both convolutional and deconvolutional networks.

[2] H. Zhang, Q. Li, Z. Sun and Y. Liu, "Combining Data-Driven and Model-Driven Methods for Robust Facial Landmark Detection,"

This paper proposed a new approach by combining data and model-driven methods: A fully connected convolutional network (FCN) was trained to compute response maps of all facial landmark points. The maximum points in the response maps were fitted using a pre-trained Point Distribution Model (PDM). It corrects an inaccurate location by considering prior information. The proposed Estimation-Correction-Tuning (ECT) method gives better or similar results to state-of-the-art methods.

[3] Y. Sun, X. Wang and X. Tang, "Deep Convolutional Network Cascade for Facial Point Detection,"

The methodology used in this research paper was using deep convolutional networks for accurately locating each key point. Two advantages of this approach are: First, the texture context information is used to locate each key point. The trained network predicts all the key points simultaneously. Deep convolutional networks give robust initial estimation which is further tuned by shallower convolutional networks.

[4] W. Wang, Y. Huang, Y. Wang and L. Wang, "Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction,"

This paper discusses the approach of autoencoders to reconstruct the pixel itself when decoded from the compressed data. So, the method proposed of generalized autoencoder in the paper adds to the working of a vanilla autoencoder by making each instance reconstruct other set of instances rather than itself. Also, the error calculation is changed to the relational function that is defined for the two instances that are affected during the reconstruction phase. The experiments done using the proposed architecture is done in the paper itself.

3. METHODOLOGY

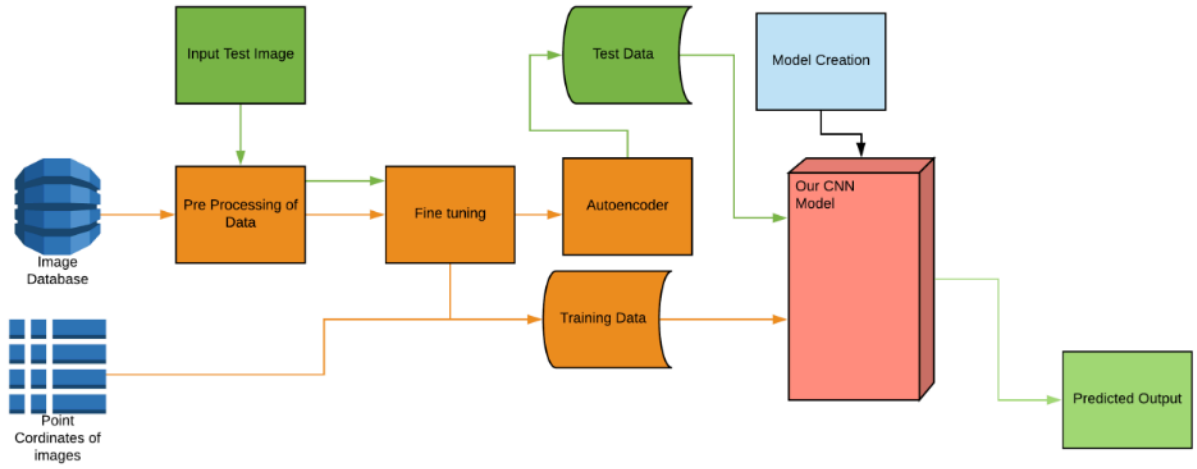

Figure 8: Block Diagram

ISSN: 2233-7857 IJFGCN

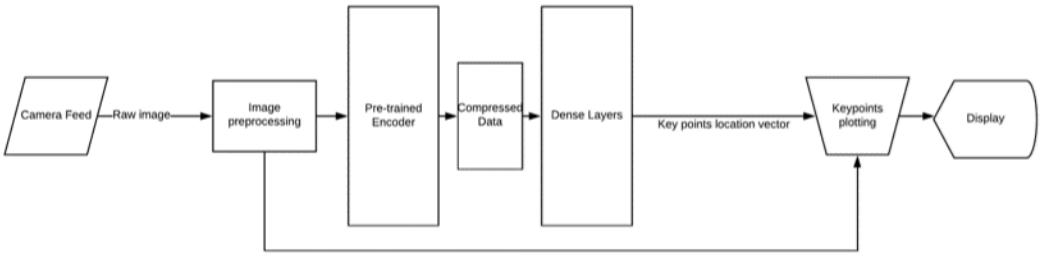

Figure 9: Flow Diagram

3.1 Datasets

The dataset is obtained from Kaggle Competition: Facial Keypoints Detection. There are 15 facial keypoints in all the images. The greyscale input images are of 96×96 pixels in size. The training dataset consists of approx. 7000 images, with 30 target values that are (x, y) coordinates for each of 15 facial keypoints. The test dataset consists of around 1700 images with no target values.

The dataset used for model training and testing purpose is Wider Facial Landmarks in-the-wild (WFLW)[18]. This dataset has 10000 faces annotated with 98 facial keypoints. The dataset has several other annotations for pose and expression estimation. These annotations can be used in some other computer vision applications. Compared with datasets like 300-W[19,20,21] and COFW[21] faces in this dataset have many variations which result in poor accuracy during testing. But performs better in the real world because of the intrinsic variance.

3.2 Preprocessing

In preprocessing we have removed all the null values then resize it to 96*96*1, As the images is converted to the greyscale for training the data, we have used 2140 images to train our model.

We have converted the preprocessed input data in a single dimension array and have provided N number of coordinates (In our demo of 15-point model) respective coordinates to images and feed it to the model for training. Using Camera, we are taking a photo/video of the face and sending it to the model for predicting the face key-points.

3.3 Preprocessing of 15 points Kaggle competition dataset

In preprocessing we have removed all the null values then resize it to 96*96*1, As the images is converted to the greyscale for training the data, we have used 2140 images to train our model.

We have converted the preprocessed input data in a single dimension array and have provided N number of coordinates (In our demo of 15-point model) respective coordinates to images and feed it to the model for training. Using Camera, we are taking a photo/video of the face and sending it to the model for predicting the face key-points.

3.3.1 Preprocessing of 98 points WFLW dataset

In preprocessing we resized images to size 100*100*1, where '*1' represents that the image is converted into the greyscale. Then converted it into tensor-data for training the model. We implemented a CNN architecture for predicting the face key points. The input data undergoes preprocessing where it is changed to meet the requirement of the model and fine-tune it to increase the accuracy of the model.

The model consists of 2 CNN Layers, 2 flattened layers, 4 Dense/Core layers.

For the real-time run, video is taking off the user and converted to greyscale video and send to the model for prediction, after that User can see his face on screen with an overlayer of key-points

ISSN: 2233-7857 IJFGCN

3.4 Training of Model

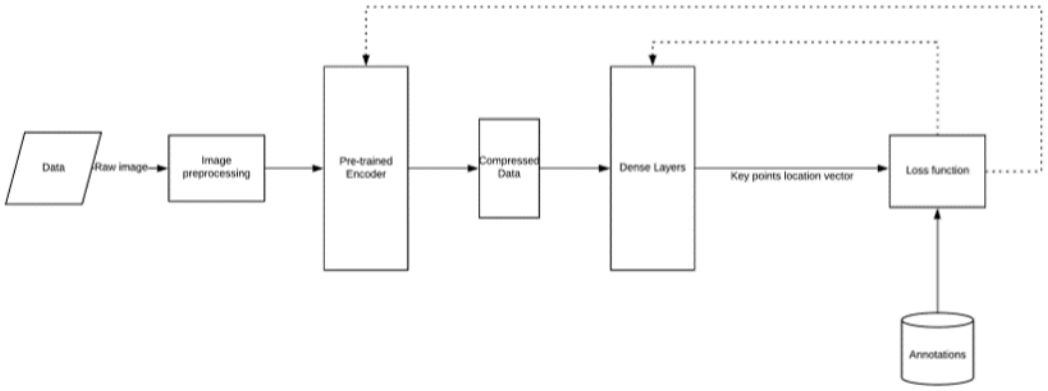

Figure 10: Training Process Flow Diagram

While training the model we are taking the processed data and sending it to autoencoder for learning the encoding of face image and decoding it. And find a best compression ratio for boosting the training process. The trained encoder's output is sent to Dense Layer/ANN for prediction of keypoints, the loss function is sent back to model for improving the accuracy of a model.

3.5 Validation/Testing

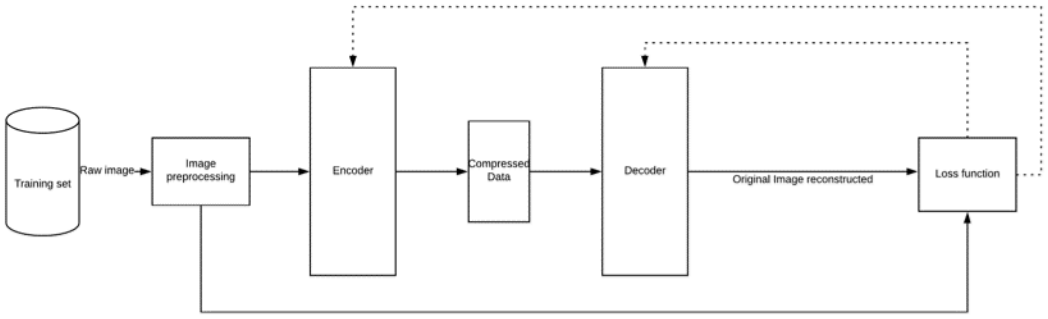

Figure 11 Validation / Testing of Model

The validation is done on a part of dataset unknown to model for evaluation of model. By validating we got to know that our model was a good fit. It didn't overfit of underfit the data. This resulted in better accuracy in test dataset. The Bias vs. Variance tradeoff was managed well.

4. RESULT

While training the 15 points model, we got and approx. accuracy of 90% (~ 0.9047) which can be seen in the Figure12, On the other hand, after training 98 points model with autoencoder we got an approx. accuracy of 81% (~ 0.8183) as shown in the Figure 13.

ISSN: 2233-7857 IJFGCN

```
Epoch 995/1000
235/235 [==============================] - 1s 6ms/step - loss: 0.0663 - accuracy: 0.8183
Epoch 996/1000
235/235 [==============================] - 1s 6ms/step - loss: 0.0681 - accuracy: 0.8223
Epoch 997/1000
235/235 [==============================] - 1s 6ms/step - loss: 0.0737 - accuracy: 0.8152
Epoch 998/1000
235/235 [==============================] - 1s 6ms/step - loss: 0.0794 - accuracy: 0.8181
Epoch 999/1000
235/235 [==============================] - 1s 6ms/step - loss: 0.0999 - accuracy: 0.8147
```

Figure 12: 15 Keypoint Model

```
Epoch 196/200
2140/2140 [==============================] - 0s 174us/sample - loss: 2.5781e-04 - accuracy: 0.9005
Epoch 197/200
2140/2140 [==============================] - 0s 172us/sample - loss: 2.8947e-04 - accuracy: 0.8874
Epoch 198/200
2140/2140 [==============================] - 0s 175us/sample - loss: 2.7502e-04 - accuracy: 0.8879
Epoch 199/200
2140/2140 [==============================] - 0s 174us/sample - loss: 2.6546e-04 - accuracy: 0.9037
Epoch 200/200
2140/2140 [==============================] - 0s 167us/sample - loss: 2.4192e-04 - accuracy: 0.9047
```

Figure 13: 98 Keypoints Model

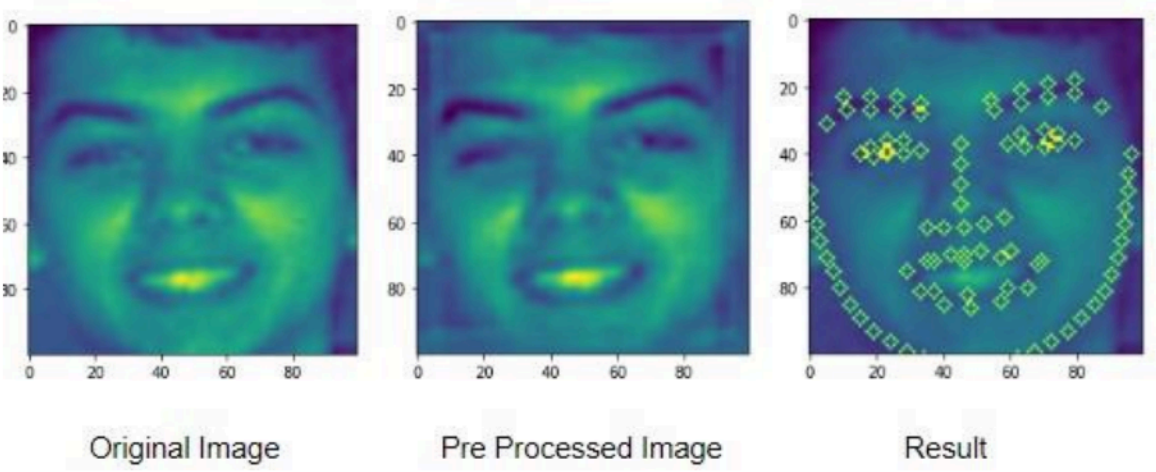

Figure 14: Final Result

To check the performance of the model in real we have plotted a ROC (receiver operating characteristic) and also found the AUC (area under the curve). This AUC is found from the ROC, ROC is formulated from the Confusion Matrix, in which We check the predicted value vs true values. For our model, we got an AUC of 0.8822 which is 88.22% and ROC

ISSN: 2233-7857 IJFGCN

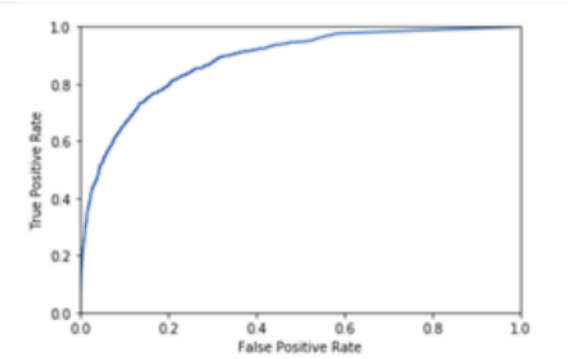

Figure 15: ROC & AUC Graph

5. CONCLUSIONS AND FUTURE SCOPE

5.1 Conclusions

After training the model on WFLW dataset with different architectures and for 1000 epochs we got an accuracy of ~81 %. We can increase the dataset size for having a more generic and accurate model. We trained autoencoders with training data and then added these pretrained layers to our CNN to achieve boosted performance. This method takes a novel approach to solve this problem and gives good results.

5.2 Future Scope

This can be used in many domains and can help us in various day to day jobs like: Face recognition, Face identification for password and security, depending on the feature we can make various applications based on it, filter or an artificial video of a real person simulated over a computer. The face filter, similarly we can extract the body features which are bigger in the area but less packed with features to identify. These can also help to simulate an artificial face/body reconstruction for a various N number of job or make our own virtual character.

6. ACKNOWLEDGEMENTS

We want to express our gratitude to MIT Academy of Engineering for providing various guidelines in writing this paper. We also thank Dr. Shitalkumar A Jain (MITAOE) for helping us sort out difficulties and constant guidance.

References

[1] H. Lai et al., "Deep Recurrent Regression for Facial Landmark Detection," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 5, pp. 1144-1157, May 2018, doi: 10.1109/TCSVT.2016.2645723.

[2] H. Zhang, Q. Li, Z. Sun and Y. Liu, "Combining Data-Driven and Model-Driven Methods for Robust Facial Landmark Detection," in IEEE Transactions on Information Forensics and Security, vol. 13, no. 10, pp. 2409-2422, Oct. 2018, doi: 10.1109/TIFS.2018.2800901.

[3] Y. Sun, X. Wang and X. Tang, "Deep Convolutional Network Cascade for Facial Point Detection," 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, 2013, pp. 3476-3483, doi: 10.1109/CVPR.2013.446.

[4] Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I. and Frey, B., 2015. Adversarial autoencoders. arXiv preprint arXiv:1511.05644.

[5] Liu, J., 2004. Global optimization techniques using cross-entropy and evolution algorithms. Master's Thesis, Department of Mathematics, University of Queensland.

[6] B. Amberg and T. Vetter. Optimal landmark detection using shape models and branch and bound. In Proc. ICCV, 2011.

ISSN: 2233-7857 IJFGCN

[7] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In Proc. CVPR, 2011.
[8] L. Gu and T. Kanade. A generative shape regularization model for robust face alignment. In Proc. ECCV, 2008.
[9] L. Liang, R. Xiao, F. Wen, and J. Sun. Face alignment via component-based discriminative search. In Proc. ECCV, 2008.
[10] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. InProc.CVPR,2012.
[11] X. Liu. Generic face alignment using boosted appearance model. In Proc. CVPR, 2007.
[12] P. Sauer, T. Cootes, and C. Taylor. Accurate regression procedures for active appearance models. InProc.BMVC,2011.
[13] M. Valstar, B. Martinez, X. Binefa, and M. Pantic. Facial point detection using boosted regression and graph models. In Proc. CVPR, 2010.
[14] D. Cristinacce and T. F. Cootes, “Feature detection and tracking with constrained local models.” in British Machine Vision Conference, vol. 1, no. 2, 2006, p. 3.
[15] X. Cao, Y. Wei, F. Wen, and J. Sun, “Face alignment by explicit shape regression,” International Journal of Computer Vision, vol. 107, no. 2, pp. 177–190, 2014.
[16] X. Xu and I. A. Kakadiaris, “Joint head pose estimation and face alignment framework using global and local cnn features,” in 12th IEEE International Conferenceon Automatic Face Gesture Recognition, vol.2, 2017,
[17] T. Pfister, J. Charles and A. Zisserman, "Flowing ConvNets for Human Pose Estimation in Videos," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1913-1921, doi: 10.1109/ICCV.2015.222.
[18] Wu, Wayne and Qian, Chen and Yang, Shuo and Wang, Quan and Cai, Yici and Zhou, Qiang. Look at Boundary: A Boundary-Aware Face Alignment Algorithm. CVPR. June 2013.
[19] C. Sagonas, E. Antonakos, G, Tzimiropoulos, S. Zafeiriou, M. Pantic. 300 faces In-the-wild challenge: Database and results. Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation "In-The-Wild". 2016.
[20] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, M. Pantic. 300 Faces in-the-Wild Challenge: The first facial landmark localization Challenge. Proceedings of IEEE Int’l Conf. on Computer Vision (ICCV-W), 300 Faces in-the-Wild Challenge (300-W). Sydney, Australia, December 2013.
[21] X. P. Burgos-Artizzu, P. Perona and P. Dollár. Robust face landmark estimation under occlusion. ICCV 2013, Sydney, Australia, December 2013.

ISSN: 2233-7857 IJFGCN