# Face Detection with 3D Models and CNN's

**Harish Pullagurla**
Department of ECE
North Carolina State University
hpullag@ncsu.edu

## Abstract

This work, presents a method for face detection in the wild, which integrates a ConvNet and a 3D mean face model in an end-to-end multi-task discriminative learning framework. 3D model, here a face, is fixed, provided by datasets as a mean avarage face (eg from AFLW dataset (2)). In this work the proposed convolutions network consists of two components: (1) The face proposal component that provided face bounding box proposals by estimating the facial key-points and 3D transformation (rotation and translation) parameters for each predicted key-point with the 3D mean face model as a reference. (ii) The face verification component computes detection results by pruning and refining proposals based on facial key-points based configuration pooling.

The proposed method addresses two issues in adapting faster R-CNN (3) for face detection: (i) One is to eliminate the heuristic design of predefined anchor boxes in the region proposals network (RPN) using a 3D mean face model. (ii) Replace the generic RoI (Region-of-Interest) pooling layer with a configuration pooling layer to respect underlying object structures. The multi-task loss consists of three terms: the classification Softmax loss and the location smooth l1 -losses (4) of both the facial key-points and the face bounding boxes. The work used images in AFLW for training and experimentation purpose

## 1 Introduction

### 1.1 Objective

This project is an attempt to implement the paper "Face Detection with End-to-End Integration of a ConvNet and a 3D Model " Yunzhun Li et.al (1). Implementing the network was done, to give an understanding about different building blocks in training a real world deep learning model.

### 1.2 Motivation

Face detection has been used as a core module in a wide spectrum of applications such as surveillance, mobile communication and human-computer interaction. It is one of the most successful applications of computer vision. Face detection in the wild continues to play an important role in the era of visual big data (e.g., images and videos on the web and in social media). However, it remains a challenging problem in computer vision due to the large appearance variations caused by nuisance variability including viewpoints, occlusion, facial expression, resolution, illumination and cosmetics, etc.

Learning how to better represent unconstrained faces has been a important area of interest among computer vision researchers. With availability of large scale annotated datasets such as ImageNet (5), significant progress has been made in the fields of generic object detection (3),(4), and similarly with face specific detection techniques (6). Combination of selective search and Region based CNN's pretrained on Imagenet is an interesting combination in this direction. Region proposal networks

Submitted as a Final Project Report for ECE763 Computer Vision.

hold the key for the success of these kinds of networks. Work in the faster rcnn integrates these two pipelines into a single network decreasing the detection time. In R-CNNs, RoI (Region-of-Interest) pooling layer (4), divides a valid RoI (e.g., an object bounding box proposal) evenly into a grid with a fixed spatial extent (e.g., 7  7) and then uses max-pooling to convert the features inside the RoI into a small feature map.

This work adapts the existing faster R-CNN architecture for generic object detection for face detection by modifying the following two limitations :

1. RPNs need to predefine a number of anchor boxes (with different aspect ratios and sizes), which requires tedious parameter tuning in training and is sensitive to the (unknown) distribution of the aspect ratios and sizes of the object instances in a random testing image.
2. The RoI pooling layer in R-CNNs is predefined and generic to all object categories without exploiting the underlying object structural configurations. This information could be obtained either from annotations on the training dataset like AFLW for face or learnt during training using deform-able part-based models.

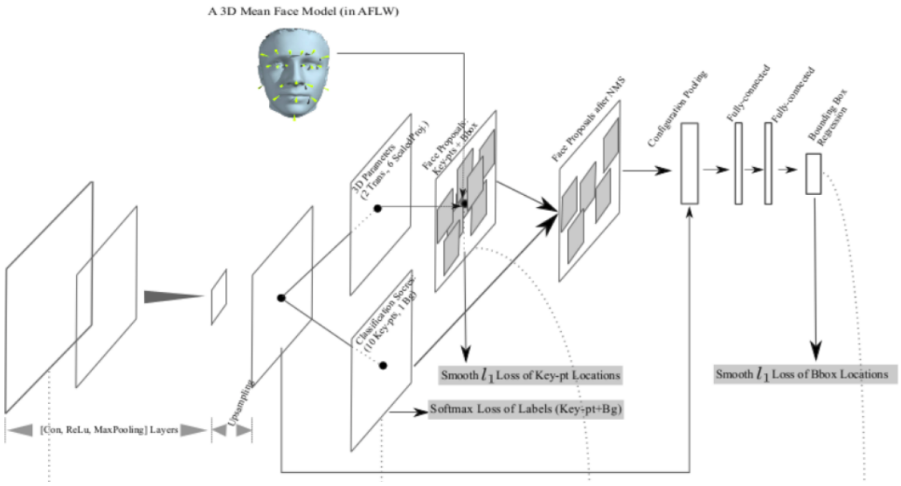


Figure 1: Network Architecture integrating 3D mean face model with ConvNets for end to end learning tasks

The network proposed in this primary reference paper (1) tries to address the two above mentioned limitations of the generic pipeline. Figure 1 shows the proposed network architecture from the paper which is being experimented in this work.

*Paper Organization :* The remainder of this paper is organized as follows. Section 2 talks about the some of the most closely related works in face detection. Section 3 presents the method of face detection using a 3D model and details of ConvNet including its architecture and training procedure from the primary reference. Section 4 presents details of experimental settings and describes the experimental results for this model. Section 5 concludes the observations and future extensions possible for this experimented work.

## 2 Literature Survey

There are a tremendous amount of existing works on face detection or generic object detection. (7) provides a thorough survey on face detection techniques. Some of the most relevant ones for this work are discussed.

Studies using fMRI experiments in the macaque reveal that faces are represented by a system of six discrete, strongly interconnected regions which illustrates hierarchical information processing in the brain (8).These findings provide some biologically-plausible evidences for supporting the usage of deep learning based approaches in face detection and analysis.

The seminal work of Viola and Jones (9) made face detection by a computer vision system feasible in real world applications, which trained a cascade of Ada Boost classifiers using Haar wavelet features.

Many works followed this direction with different extensions proposed in four aspects:

1. Appearance features (beside Haar) including Histogram of Oriented Gradients (HOG), Aggregate Channel Features (ACF), Local Binary Pattern (LBP) features and SURF, etc.
2. Detector structures (beside cascade) including the the scalar tree and the width-first-search tree , etc.;
3. Strong classifier learning (beside AdaBoost) including RealBoost and GentleBoost, etc ;
4. Weak classifier learning (beside stump function) including the histogram method and the joint binarizations of Haar-like feature etc

Most of the recent face detectors are based on the deformable part-based model (DPM) with HOG features used, where a face is represented by a collection of parts defined based on either facial landmarks or heuristic pursuit as done in the original DPM. Deep learning further boosted the face detection performance by learning more discriminative features from large-scale raw data, going beyond those handcrafted ones. In the FDDB benchmark, most of the face detectors with top performance are based on ConvNets, combining with cascade (10) and more explicit structure (11)

3D information has been exploited in learning object models in different ways. Some works used a mixture of 3D view based templates by dividing the view sphere into a number of sectors. Fe others utilized 3D models in extracting features and inferring the object pose hypothesis based on EM or DP. (12) used a 3D face model for aligning faces in learning ConvNets for face recognition. (13) is the other closest relevant work, which computes meaningful 3D pose candidates by image-based regression from detected face key-points with traditional handcrafted features, and verifies the 3D pose candidates by a parameter sensitive classifier based on difference features relative to the 3D pose. The present implementation (1) extends the same idea, integrating CNN's and 3D models in an end-to-end multi-task discriminative learning fashion.

## 3 Method

This section aims to describe in detail the network architecture being used and all necessary backgrounds on the topics explained. The key idea is to learn a ConvNet to

1. Estimate the 3D transformation parameters (rotation and translation) w.r.t. the 3D mean face model for each detected facial key-point so that we can generate face bounding box proposals and
2. Predict facial key-points for each face instance more accurately.

Use of 3D models helps us eliminate the manual heuristics of designing the anchor boxes using Region Proposal Networks. Also instead of generic RoI pooling , configuration pooling layer is used, as to respect the object structural configurations in a meaningful, considering one sample from each sub-regions. This network proposes to compute the proposals in a straight-forward top-down manner, instead of to design the bottom-up heuristic and then learn related regression parameters based on it.

### 3.1 Input Data

The networks is designed with a certain number of assumptions. It requires 3D mean face model and facial key-point annotations in the training data set. For this experiment AFLW data set is used . 10 Facial key points have been used for this purpose, which include "LeftEyeLeftCorner", "RightEyeRightCorner", "LeftEar", "NoseLeft", "NoseRight", "RightEar", "MouthLeftCorner", "MouthRightCorner", "ChinCenter", "CenterBetweenEyes". This data set also provides a facial bounding box and a facial ellipse for each annotated face. Figure 2 shows a sample annotated image from the data set.

**Input Data** Denote by C = { 0,1,$\cdots$, 10} as the key-point labels where $l = 0$ represents the background class. The image-centric sampling trick is used in this experiment. Considering a training image

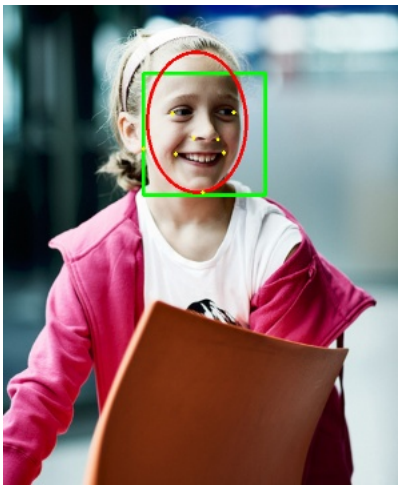

Figure 2: Example of Input Annotated Image in AFLW dataset

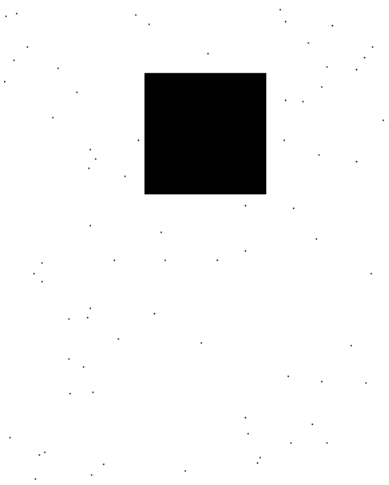

Figure 3: Background flagmap for a given image

with only one face appeared, its bounding box are, B = $(x, y, w, h)$ and $m$ 2D key-points $(m \leq 10)$, $(x_i, l_i)_{i=1}^{m}$ where $x_i = (x_i, y_i)$ is the 2D position of the ith key-point and $l_i \geq 1 \in C$. m locations outside the face bounding box B are randomly sampled as the background class, $(x_i, l_i)_{i=m+1}^{2m}$ (where $l_i = 0, \forall, i > m$). half size along both axes of the original input. All the key-points and bounding boxes are defined as being half size for this experimentation purpose, accordingly based on ground-truth annotation.

### 3.2 Network Architecture

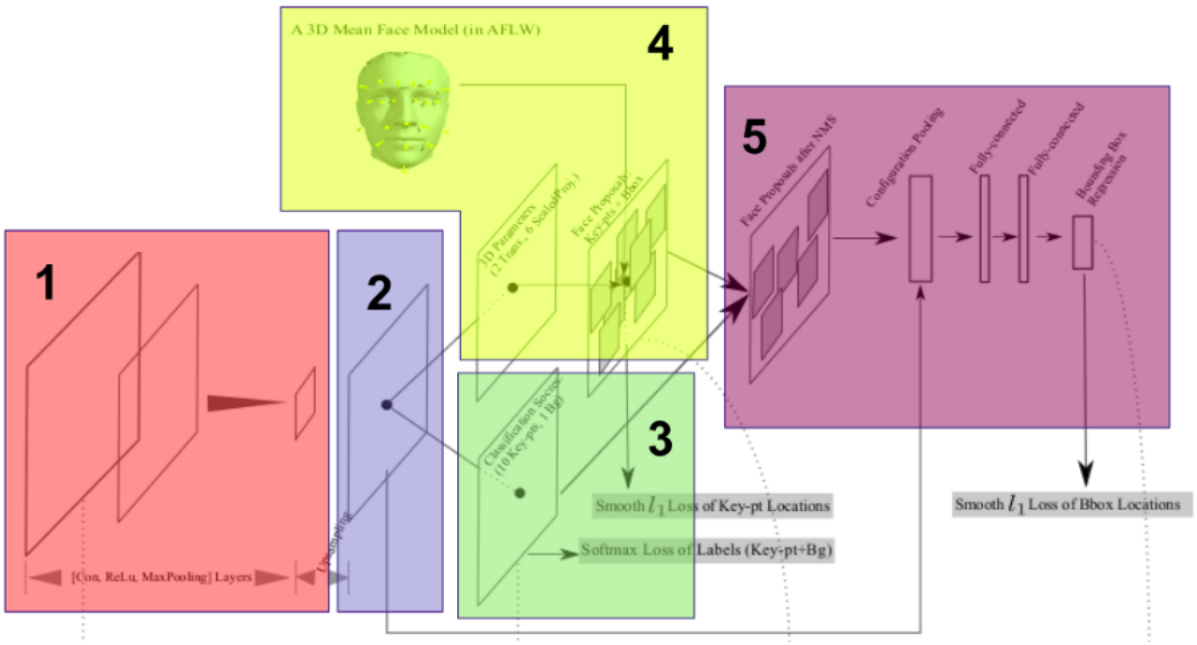


Figure 4: Network Architecture divided into sections

Figure 4, shows the network divided into sections. Each of the network sections is explained in detail in the subsection below.

### 3.2.1 Feature Backbone and Deconvolution

Section 1 in the network takes in as input the images and outputs a feature map for each image. The task of this submodule it to obtain a feature map which could be used for further processing. For this task networks like resnet or VGG (14), trained on image-net data-set can be used. In this experiment VGG 16 has been used as the feature backbone. If input to the network is of dimensions [w,h.3], output after 5 sets of conv - pool - relu would be of dimensions [w/16, h/16, 512]. This part of the network, trained on the ImageNet data set is not updated in this experiment because of the limited GPU compute capabilities.

This network down-samples the inputs by 16 times, which causes the points that are close to each other to get mapped together. Hence, for better spacial separation the feature maps obtained are up-sampled using deconvoluion operators as shown in Section 2 of the network architecture by 8 times. After this operation, the dimensions are set to one half of the input dimensions with channel depth of 64. This feature map is used for all further operations done in this experiment. The Sections 1  2 can be replaced by using a networks that designed to do the upsampling work more correctly, like unet (15).

### 3.2.2 Key-point Classification Head

Section 3 in the network architecture describes a facial key-point label prediction layer that is added. At each pixel location, there are 11 possible labels (10 facial key-points and 1 background class). It is used to compute the classification Softmax loss based on the input in training.

The **Classification Softmax Loss of Key-point Labels**, att each position $x_i$ , the ConvNet outputs a discrete probability distribution, $p^{x_i} = (p_0^{x_i}, p_1^{x_i}, \cdots, p_{10}^{x_i})$, over the 11 classes, which is computed by the Softmax over the 11 scores (16). Hence the keypoint classification loss is,

$$L_{cls}(\omega) = -\frac{1}{2m}\sum_{i=1}^{2m} log(p_{l_i}^{x_i}) \tag{1}$$

### 3.2.3 Region Proposal Sub Network

Section 4 in the network architecture acts as a 3D Transformation Parameter Estimation Layer. Originally, there are 12 parameters in total consisting of 2D translation, scaling and 3  3 rotation matrix. Since the only focus is the 2D projected key-points, only 8 parameters are needed for computation.

A face, denoted by f , is presented by its 3D transformation parameters, $\Theta$, for rotation and translation, and a collection of 2D key-points, $F^{(2)}$ , in the form of $(x, y)$ . Hence, $f = (\Theta, F^{(2)})$. The 3D transformation parameters $\Theta$ are defined by

$$\Theta = (\mu, s, A^{(3)}) \tag{2}$$

where $\mu$ represents a 2D translation $(dx, dy)$, s a scaling factor, and $A^{(3)}$ 3x3 rotation matrix. We can compute the predicted 2D key-points by,

$$F^{(2)} = \mu + s\pi(A^{(3)}\Delta F^{(3)}), \tag{3}$$

where $\pi$ projects a 3D key-point to a 2D one. Due to the projection only 8 of the original 12 parameters are required.

A Face Proposal Layer is the second sub module in Section 4. At each position, based on the 3D mean face model and the estimated 3D transformation parameters, 10 facial key-points are predicted corresponding to a face proposal and the corresponding face bounding box too.

The score for each non face position corresponding to the face proposals, is the sum of log probabilities of the 10 predicted facial key-points is calculated. At each location, the probability from classification task corresponding to that label is taken for summation.

$$FacenessScore(\hat{x}_i, \hat{l}_i) = \sum_{i=1}^{10} log(p_{l_i}^{x_i}) \tag{4}$$

The predicated key-points are used to compute the smooth l1 loss (4) w.r.t. the ground-truth key-points.

The **Smooth l1 Loss of Key-point Locations** is given by

$$L_{loc}^{pt}(\omega) = \frac{1}{m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} \sum_{t \in (x,y)} Smooth_{l1}(t_i - \hat{t}_{i,j}) \tag{5}$$

At each non background key-point location this factor is computed. Non-maximum suppression (NMS) is applied to the face proposals giving the requirements.

#### 3.2.4 Fine Tuning of Proposals

Fine Tuning of region proposals is done in Section 5 of the network. This network gets as inputs the region proposals which, crops them and stacks them together with each bounding box being of fixed dimension. This is done using roi pooling layer in faster rcnn. In this work a new experiment of using configuration pooling is done. This operator take in max from each of the keypoint regions instead of grid wise sampling the given input.
This layer is followed by two fully connected layers, which finally generate 5 bounding box coefficients. The obtained regression coefficients are used for computing the smooth l1 loss coefficients.

For each face bounding box proposal B (after NMS),the ConvNet computes its **bounding box regression** offsets, $t = (t_x, t_y, t_w, t_h)$, where t specifies a scale-invariant translation and log-space height/width shift relative to a proposal. For the ground-truth bounding box B, the same operation is done and hence $v = (v_x, v_y, v_w, v_h)$. Assuming that there are K bounding box proposals, the loss function equation gives,

$$L_{loc}^{box}(\omega) = \frac{1}{K} \sum_{k=1}^{K} \sum_{i=x,y,w,h} Smooth_{l1}(t_i - v_i) \tag{6}$$

The final over all loss would be summation of the 3 loss terms.

$$L(\omega) = L_{loc}^{box}(\omega) + L_{loc}^{pt}(\omega) + L_{cls}(\omega) \tag{7}$$

Hence the networks aim is the minimize the complete loss.

## 4 Implementation Details and Results

This section focuses on the implementation details of the above network and the results corresponding to it. This work discusses the outputs, from the Section 3 (classification head) and Section 4 (regression head) of the network architecture. Discussion of final outputs from fine tuning in Section 5 of network architecture has been left over as future work because of unexpected implementation error in one of the custom operators.

### 4.1 Implementation Details

#### 4.1.1 Training Data-set

Data from AFLW (2) has been used for experimentation. This data set has 25,993 face annotation in around 21k images. 15k images from it have been used for training and the remaining have been split for validation and testing purpose. The facial key-points are annotated upon visibility w.r.t. a 3D mean face model with 21 landmarks. For this experimentation 10 primary key points of the 21 annotated types have been used.

#### 4.1.2 Training Process

All the code, right from reading the database for annotations, training and final visualizations have been written in python 2.7. In all the images, the shorter egde is resized to 600 pixels while preserving

the aspect ratio. In doing so the model gets to learn how to handle under different scale. Also an upper bound limit of 1000 pixels was kept, preserving the aspect ratio . This step was taken to counter highly odd shaped images, which were resulting in huge memory allocations in the deconvolution layers. In addition, as a pre processing step faces were randomly blurred using a Gaussian filer with a probability of 0.25 having a kernel size in the range of (5,20), randomly chosen. Apart from the rescaling and blurring, no other preprocessing mechanisms (e.g., random crop or left-right flipping) are used.

An image centric sampling method which uses one image per batch during the training time has been adopted. It is assumed that a 3x3 grid neighbourhood around the labeled annotated position share the same label. For this reason, every 3x3 labeled key point position has been given a positive label . Figure 3 shows the background points being randomly chosen outside the bounding box, shown in black mask. The convolution filters are initialized by the VGG-16 (14) pretrained on the ImageNet. The layers taken from vgg are frozen because of limited gpu compute available for training.

For training, cloud compute resources (ec2 instances) from Amazon Web Services (17) were used. For this experiment, spot instances of P2 Xlarge type were rented. Each instance of this type comes with 1 Nvidia K80 GPU, 4 vCPU's and 61 GB RAM. Deep Learning v8 on a Ubuntu machine was choose which comes pre installed with required CUDA kernels. A s3 volume to transfer and store all data was used. I would like to thank AWS for providing with 100 USD free student credits to help this experimentation happen. Custom operators for smooth l1 loss computation , face projection computation, ellipse and bounding box computation was added separately. Mxnet was rebuilt from these sources with these custom operators. The network is trained for 5 epoch, and during the process, the learning rate is modified from 0.01 to 0.0001.

### 4.2 Evaluation of Intermediate Results

Table 1: Intermediate Keypoint Classification Accuraacy

| Category | Accuracy | Category | Accuracy |
|---|---|---|---|
| Background | 98.2 | AverageDetectionRate | 84.3 |
| RightEyeRightCorner | 97.8 | LeftEyeLeftCorner | 77.5 |
| NoseLeft | 94.7 | NoseRight | 82.6 |
| RightEar | 81.4 | LeftEar | 90.1 |
| MouthRightCorner | 87.1 | MouthLeftCorner | 93.2 |
| CenterBetweenEyes | 88.1 | Chin Center | 99.4 |

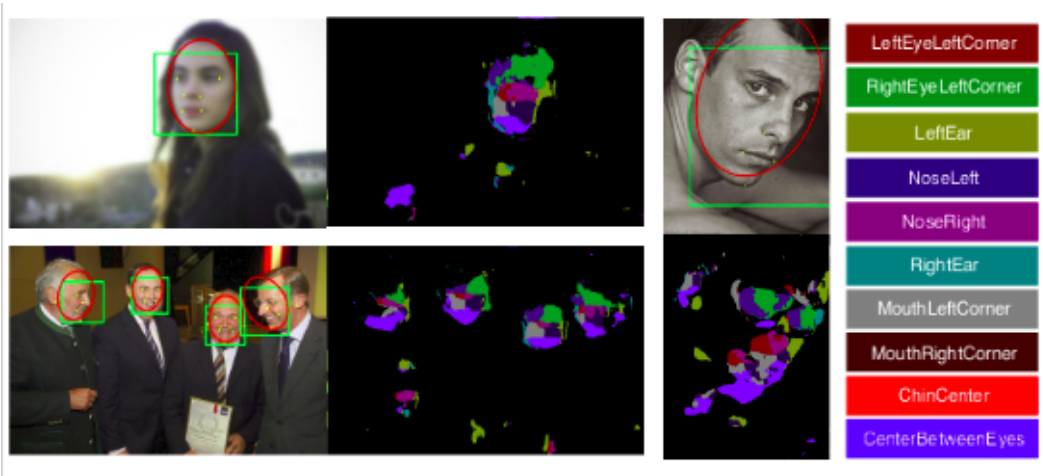


Figure 5: Qualitative heat-map output in ALFW Dataset

**Key-points classification** in the validation dataset is shown in table 1. It is seen from the detection results in the table that, this model was able to achieve results similar to the one given by the reference

paper. Training it for a few more epochs might help in achieving the required accuracy.
Figure 5 shows the heat maps of classification results. These heat maps were plotted by taking the maximum of prediction values at each pixel position, which gives the label maps. The model is capable of detecting facial key-points with rough face configurations preserved, which shows the effectiveness of exploiting the 3D mean face model.

Figure 6: Region proposal bounding boxes from different learning instances starting from the left

**Face proposals.** To evaluate the quality of face proposals, some qualitative results are shown in Figure (**?** ). These ellipses are directly calculated from the predicted 3D transformation parameters, forming several clusters around face instances. Non-maximum suppression of I0U 0.7 was applied to the generated proposals before giving the outputs to the next subsection of fine tuning proposed bounding boxes.

## 5 Conclusion

Through this process, a new experimental network architecture proposed in (1) has been reprogrammed to reproduce the results. This network proposes a method to do end-to-end integration of a ConvNet and a 3D model for face detection in the wild. The model provides a clean and straightforward solution when taking into account a 3D model in face detection. It also addresses two issues in generic object detection ConvNets: eliminating heuristic design of anchor boxes by leveraging a 3D model, and overcoming generic and predefined RoI pooling by configuration pooling which exploits underlying object configurations.

This experiment tests the classification head and the face proposal network on the complete architecture. The face key-point classification results could be reproduced to a great extent. Also the face proposals were visually plotted and evaluated. Ti is conclude that they provide reasonable good face proposals. These proposals are to be fine tuned to get the final output through bounding box regression coefficients.

## 6 Project Learning's and Future Work

Working on the project acted as a great starting point in learning about **NOT** off the shelve, convolutions neural network. It acted as a good hands on experience in knowing about individual components involvement in getting the final output and multitask learning framework. Good debugging skills and importance of use of visualizations to check the correctness of outputs was acquired.

Primary work to be completed is the inclusion of the Section 5 i.e fine tuning of the proposal bounding boxes to get the completeness of the proposed architecture. The initial structure of the network, can be replaced with other components such as resent or unet for better performance. The work can ve further extended towards other models like cars and also to include non rigid objects into this pipeline.

## 7 GitHub Repository

The code for the project was developed using python 2.7, on top of Mxnet 1.0 as a base deep learning framework. This code has several custom operators included, hence it is compulsory to rebuild the Mxnet framework after adding the operator from scratch. Mxnet 1.0 used CUDA 8.0 version, which is to be checked while building the framework. Several other open source libraries like opencv were used in helper functions.

The code could be accessed at https://github.com/pharish93/FaceDetection

## References


[1] Li Y., Sun B., Wu T., Wang Y. (2016) Face Detection with End-to-End Integration of a ConvNet and a 3D Model. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9907. Springer, Cham

[2] Koestinger, M., Wohlhart, P., Roth, P.M., Bischof, H.: Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In: First IEEE International Workshop on Benchmarking Facial Image Analysis Technologies (2011)

[3] Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS (2015)

[4] Girshick, R.: Fast R-CNN. In: ICCV (2015)

[5] Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li, F.: Imagenet: A large-scale hierarchical image database. In: CVPR. pp. 248–255 (2009)

[6] Ranjan, R., Patel, V.M., Chellappa, R.: A deep pyramid deformable part model for face detection. In: IEEE 7th International Conference on Biometrics Theory, Applications and Systems (2015)

[7] Zafeiriou, S., Zhang, C., Zhang, Z.: A survey on face detection in the wild. Comput. Vis. Image Underst. 138(C), 1–24 (Sep 2015)

[8] Freiwald, W.A., Tsao, D.Y.: Functional compartmentalization and viewpoint gen- eralization within the macaque face-processing system. Science 330(6005), 845–851 (2010)

[9] Viola, P.A., Jones, M.J.: Robust real-time face detection. International Journal of Computer Vision 57(2), 137–154 (2004)

[10] Li, H., Lin, Z., Shen, X., Brandt, J., Hua, G.: A convolutional neural network cascade for face detection. In: CVPR (2015)

[11] Yang, S., Luo, P., Loy, C.C., Tang, X.: From facial parts responses to face detection: A deep learning approach. In: ICCV (2015)

[12] Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human- level performance in face verification. In: CVPR (2014)

[13] Barbu, A., Gramajo, G.: Face detection using a 3d model on face keypoints. CoRR abs/1404.3596 (2014)

[14] Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. In: British Machine Vision Conference (2014)

[15] Olaf Ronneberger, Philipp Fischer, Thomas Brox. : U-Net: Convolutional Networks for Biomedical Image Segmentation arXiv:1505.04597

[16] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep con- volutional neural networks. In: NIPS (2012)

[17] https://aws.amazon.com/ - Amazon Web Services, Cloud computing platform - Last Accessed 05 May 2018