Head Pose Detection

Bharath Bandaru [A20493607]
Illinois Institute of Technology
bbandaru@hawk.iit.edu

Himamshu Lakkaraju [A20491089]
Illinois Institute of Technology
hlakkaraju@hawk.iit.edu

Abstract

Head Pose detection and face detection are active research topics in machine learning and computer vision. The models that try to solve these usually require large labeled datasets and require deep nets which are computationally expensive and complicated to design. As computation power increased dramatically in the last few years, especially with GPUs which provide excellent parallelization of computations, it has become relatively easy to design neural networks that can perform these operations. We find many areas that are using face detection and head pose detection especially with the emergence of service robots and surveillance cameras, these problems have received much attention in recent years. In this project, we implement a neural network for head pose estimation. It can be used in autonomous vehicles to detect driver attention, industry equipment safety or detect areas of attention of people in a group setting.

1 Introduction

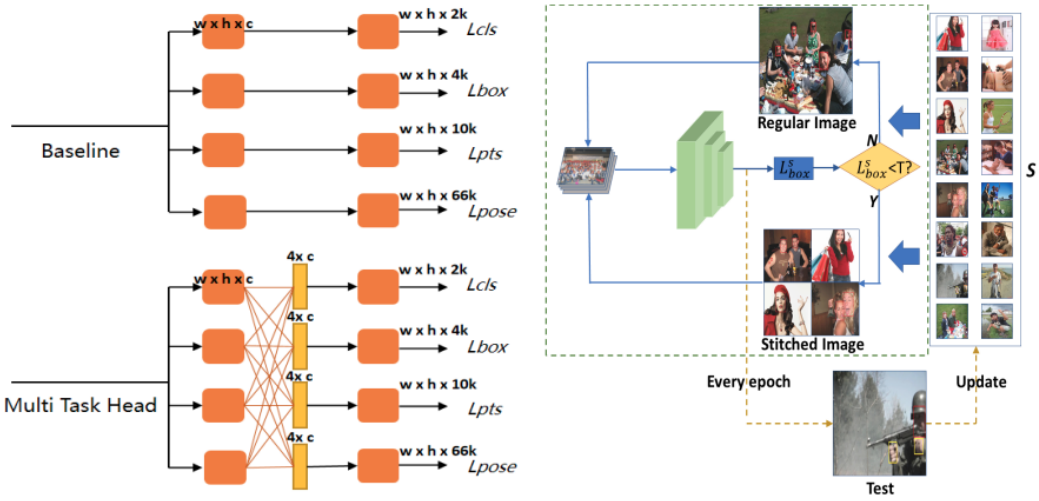
Many of the governments, industries and institutions today deploy some form of face detection/ face recognition/ face pose detection either for safety or to improve the work efficiency or detect threats or find areas of attention in a location etc. These face detection algorithms are typically trained using Convolutional Neural Networks. The networks are created in such a way that the key landmarks (eyes,nose, ends of mouth) and pose of a person are extracted with as few layers as possible. This is usually achieved by using image pyramids which can scale the input images (usually using a 3*3, 5*5 and 7*7 filters) and helps in identifying these landmarks more efficiently. Identifying the head pose from the images is a challenging task as the training data available is scarce and not consistent with the face size in the image or the angles or the face lighting conditions etc. Even after training the images with a wide range of training data, the pose estimation of the face is still a challenging task, especially if the face is at an extreme angle (some landmarks might not be available or be very close to each other for example looking sideways the landmarks for eyes, mouth are almost next to each other if not overlapped). Most of the available models at the moment either perform face and landmark detection well or perform head pose detection without using the face boundaries or landmarks. Models like MOS which perform both pose detection and face detection require large amounts of training data and require a high number of epochs during the training (MOS model is trained using 240 epochs on the entire widerface dataset). While the accuracy of these models is very high, retraining these models for a custom use case might not be a simple endeavor.

2 Related Work

Face, pose recognition is a hot topic in machine learning and computer vision and is a very active research area. Initially attempts have been made to solve this problem by using just the facial landmarks, Perspective-n-point (pnp) etc. The newer methods heavily use CNN, these include MOS[1], MTCNN[2], Hopenet[3], FSAnet[4].

2.1 MOS Architecture [1]

Methods other than MOS are good at either face detection, landmark estimation or are good at pose estimation. There aren't many methods that combine face detection and pose estimation. MOS implementation does both face detection and pose detection by using a multi task loss with cross connection. This results in the MOS being able to perform the face detection, landmark detection and pose detection all at once. The MOS head pose detection in the reference paper uses an online sampling feedback and is built on the backbone of ShuffleNet and ResNET152. These backbone models are pre trained on the Imagenet dataset. The dataset used for training is a modified version of the WIDER Face dataset with added pose labels in the target data.



(a) The different connection method of head (b) The pipeline of online feedback sampling modules

2.2 HopeNet Architecture [3]

HopeNet is another CNN model that uses ResNET to identify faces and uses the yaw,pitch,roll to compute the losses and estimates the pose of the face in the image. This model performs only Face pose detection.

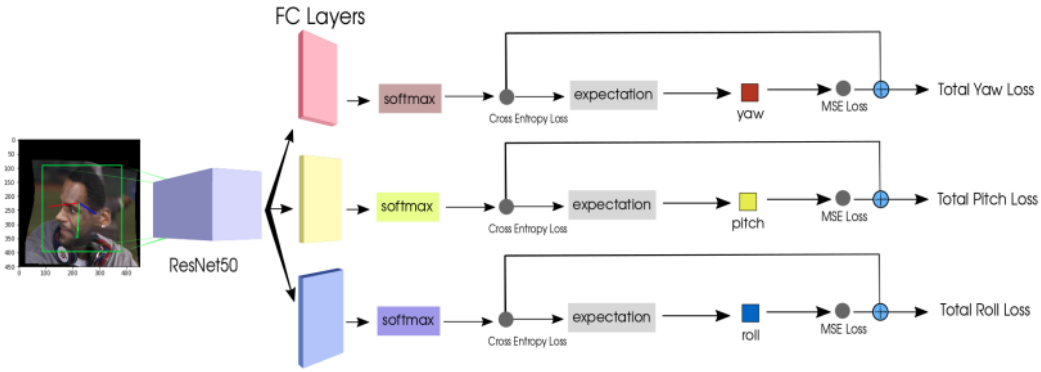


Figure 2. ResNet50 architecture with combined Mean Squared Error and Cross Entropy Losses.

3 Proposed Solution

We plan to train a simple Convolutional Neural Network for pose detection (yaw,pitch,roll) and then use the existing MTCNN model for identifying the landmarks and bounding boxes of the faces and join these two together to give the final output.

The data used to train the network for pose detection can be found at [Drive link].

This model is trained by using 3 loss functions for yaw,pitch and roll. The loss is a linear combination of binned pose classification and the angle regression component losses. The losses are factored using [1,0.3,0.3] for yaw,pitch,roll. This is done to give more significance to the yaw loss values as we have large variance for this in the training data.

4 Implementation

The dataset 300W_LP is downloaded from the drive link. A folder is chosen for training (As using all the data takes too long and might require more memory).

The cropped images which contain the face in the images along with the pose labels for that face in the image and the binned values for the yaw, pitch, roll values are sent to the model for training.

The simple neural network architecture is as below.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 240, 240, 3)]	0	[]
conv2d (Conv2D)	(None, 60, 60, 64)	23296	['input_1[0][0]']
max_pooling2d (MaxPooling2D)	(None, 29, 29, 64)	0	['conv2d[0][0]']
conv2d_1 (Conv2D)	(None, 29, 29, 192)	307392	['max_pooling2d[0][0]']
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 192)	0	['conv2d_1[0][0]']
batch_normalization (BatchNormalization)	(None, 14, 14, 192)	768	['max_pooling2d_1[0][0]']
conv2d_2 (Conv2D)	(None, 14, 14, 384)	663936	['batch_normalization[0][0]']
conv2d_3 (Conv2D)	(None, 14, 14, 256)	884992	['conv2d_2[0][0]']
batch_normalization_1 (BatchNormalization)	(None, 14, 14, 256)	1024	['conv2d_3[0][0]']
conv2d_4 (Conv2D)	(None, 14, 14, 256)	590080	['batch_normalization_1[0][0]']
conv2d_5 (Conv2D)	(None, 14, 14, 512)	1180160	['conv2d_4[0][0]']
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 512)	0	['conv2d_5[0][0]']
batch_normalization_2 (BatchNormalization)	(None, 6, 6, 512)	2048	['max_pooling2d_2[0][0]']
flatten (Flatten)	(None, 18432)	0	['batch_normalization_2[0][0]']
dropout (Dropout)	(None, 18432)	0	['flatten[0][0]']
dense (Dense)	(None, 4096)	75501568	['dropout[0][0]']
yaw (Dense)	(None, 66)	270402	['dense[0][0]']
pitch (Dense)	(None, 66)	270402	['dense[0][0]']
roll (Dense)	(None, 66)	270402	['dense[0][0]']

Total params: 79,966,470
Trainable params: 79,964,550
Non-trainable params: 1,920

This is a multi loss model which has 3 loss functions one for each of the yaw, pitch and roll. Each of these losses have a cross entropy loss (classification loss) that computes loss between the predicted bin value and the actual bin value and the other loss function is an MSE loss (regression) to calculate the loss between the predicted value and the actual value for the yaw, pitch, roll. The loss functions are as below.

$$L = H(y, \hat{y}) + \alpha.MSE(y, \hat{y})$$

For getting the final output, the input image is sent to the MTCNN (external module) and the boundaries of the face in the image are obtained. This is used to crop the face in the image and pass it as an input to our model which gives the yaw, pitch and roll of the face in the image. These are combined to obtain the output image which has the face boundaries, landmarks and the pose angles of the faces in the input image.

5 Results

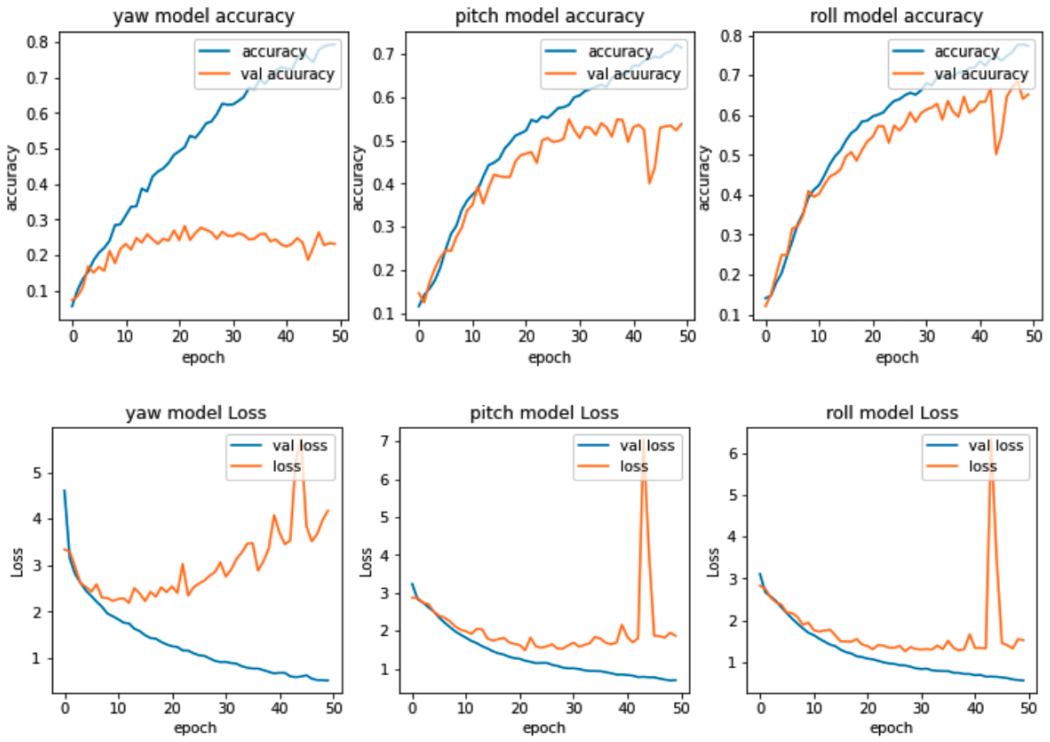
We trained 3 different models. The differences between the 1st and 2nd model is in the loss function. While the first model we created used a CrossEntropyLoss that calculated the difference between the bins in the image (works like a classification task), the 2nd model used a linear combination of CrossEntropyLoss for the bins of yaw,pitch,roll and MSE loss(regression) for yaw, pitch, roll values. The 3rd model is similar to the implementation of hopenet which uses ResNet in the model. This is done to compare against our model for the same training data and same parameters and epochs.

The pose estimation from our 1st model with CrossEntropyLoss performs poorly, especially yaw angle, which is not very accurate. This might be because of the limited amount of data used for training (lack of variation in the images etc), width of the bins used (Step of 3 is used for binning) , the number of epochs used for training or the shallow nature of the network itself.

The pose estimation of our 2nd model performs significantly better than the 1st model.

This model uses CrossEntropyLoss of the bins along with MSE of the actual yaw,pitch,roll values. The overall train and validation loss is lower compared to the first model. The MAE is less than the first model. The 3rd model that used the ResNet block and trained on the same train dataset as the other 2 models, with same train parameters and number of epochs didn't offer much improvement in losses or the MAE on the test data.

5.1 Model 1 (using only CrossEntropyLoss for the binned values of yaw, pitch, roll):



MAE on test data:

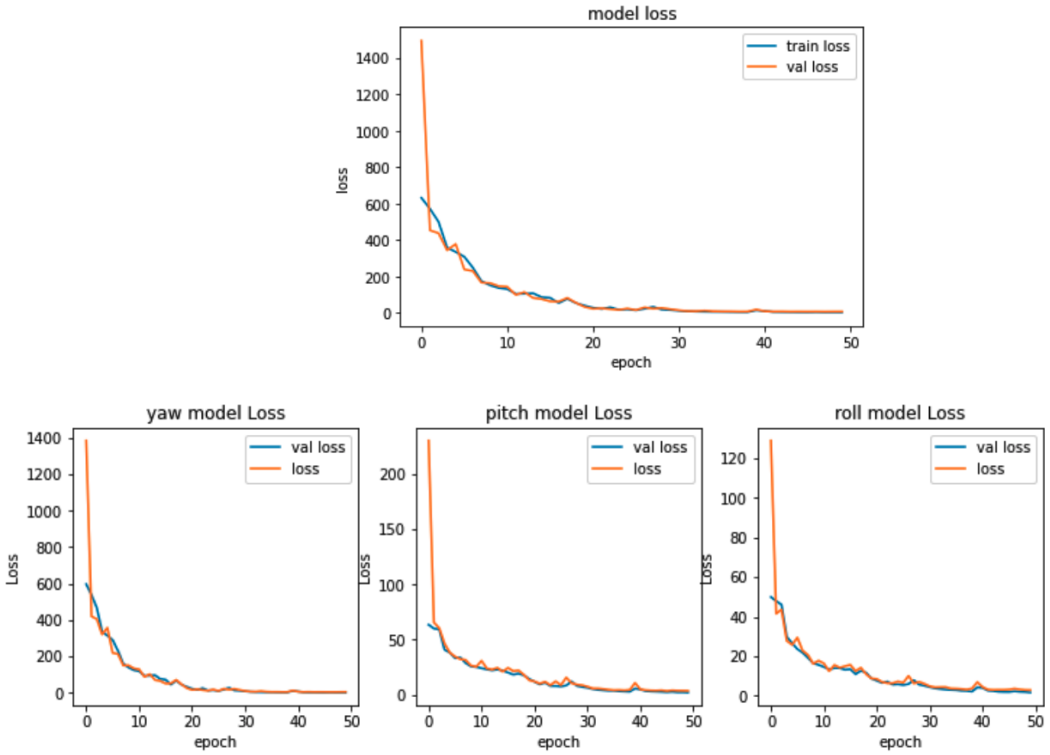
Yaw : 31.11739392743724

Pitch : 30.057766290352276

Roll : 32.32045502103953

MAE combined = 31.16520507960968

5.2 Model 2 (using linear combination of CrossEntropyLoss and MSE of yaw,pitch,roll)



MAE on test data:

Yaw : 49.08532128214418

Pitch : 10.826697878821816

Roll : 8.032791612598823

MAE combined = 22.648270257854943

For a complex problem like pose detection, the model needs to train for more epochs on a wider dataset to generalize well. In our case we were restricted due to the limited resources. Using CUDA significantly reduces the training time of the model. While using cuda we might have to use lower amounts of training data due to limited availability of the memory but using the CPU takes almost twice as long for each epoch thereby doubling the amount of time required for training.

The hyper parameter tuning part of the network included change in the batch size of the training set, change in learning rate, change in the alpha value that affects the overall combined CrossEntropyLoss and MSE loss.

6 Conclusion and Future work:

We'd like to combine WIDERFACE and the 300W_LP datasets and try training these models using the combined dataset for more epochs and see if training the model for more epochs would increase the accuracy of the pose detection and result in decrease in the MAE on the test dataset. We'd like to

create a CNN that can perform multiple scale analysis and compute face boundaries, landmarks and the pose of the face using a combined multi task loss like implemented in the MOS neural network.

7 Code Repository

`https://github.com/bharath-bandaru/head-pose-detection`

8 Acknowledgement

This project is done as part of the course project for CS-584-01: Machine Learning - Spring 2022.

9 Contributions

Task	Member
Data collection	Bharath
Data processing	Himamshu
Project Planning	Bharath, Himamshu
Model Training	Bharath, Himamshu
Testing	Bharath, Himamshu
Evaluation	Himamshu
Documentation/Presentation	Bharath,Himamshu
Report	Bharath

10 Libraries and tools used

- PyTorch
- Pandas
- Numpy
- OpenCV
- Pillow
- Scikit-learn
- MTCNN
- FaceNet
- SciPy
- VS Code

11 References:

[1] Yepeng Liu, Zaiwang Gu, Shenghua Gao, Dong Wang, Yusheng Zeng, Jun Cheng. MOS: A Low Latency and Lightweight Framework for Face Detection, Landmark Localization, and Head Pose Estimation. arXiv:2110.10953v3

[2] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao. Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks. arXiv:1604.02878

[3] Nataniel Ruiz, Eunji Chong, James M. Rehg. Fine-Grained Head Pose Estimation Without Keypoints. arXiv:1710.00925

[4] Tsun-Yi Yang, Yi-Ting Chen, Yen-Yu Lin, Yung-Yu Chuang. FSA-Net: Learning Fine-Grained Structure Aggregation for Head Pose Estimation From a Single Image. CVF Open access paper