Facial Recognition Using Convolutional Neural Networks

Project Guide:R.Shriram

Team: Vigneshwaran J - 130071601095
Ramakrishnan Ramanarayanan - 130071603007

Abstract

Facial Recognition can be used to authenticate a person passively and can be also used for mass identification. This quality is extremely useful in Home Automation systems where members of a household can be authenticated in a seamless manner which produces a good balance between security and ease of use.

With the advancement of Convolutional Neural Network(CNN), it is possible to train it to identify and classify faces much more accurately since it identifies the crucial features in the convolution layer and thus can identify a familiar face under various lighting conditions.

Modules Completed

Model Creation.

Training the Model.

Visualizing Results.

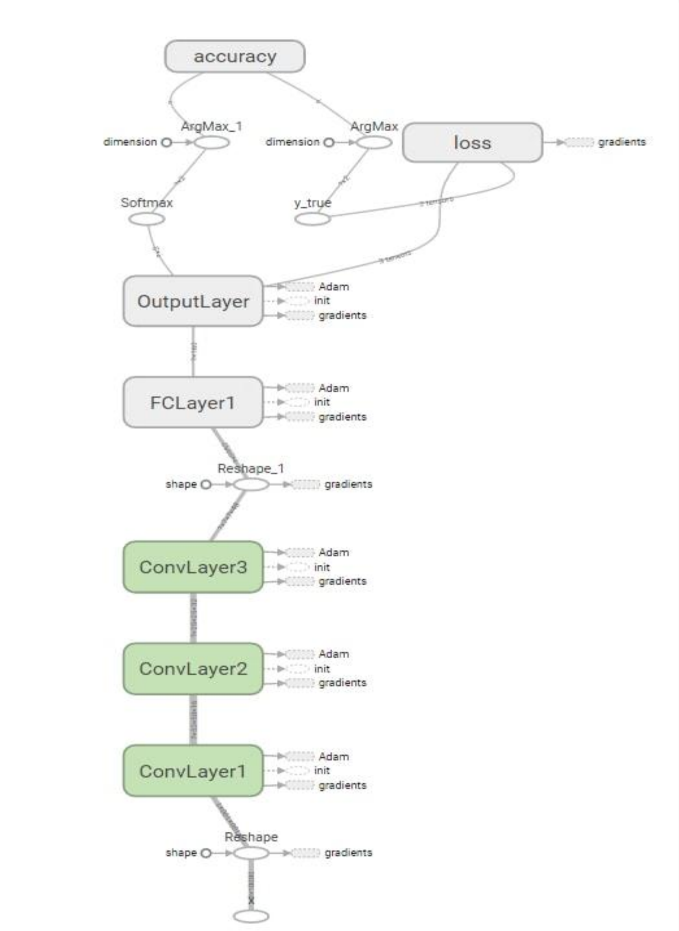
Hardware Requirement

- For Training
  - Intel Processor i5/i7/Xeon.
  - Minimum Nvidia Geforce 980M, Recommended 4 Nvidia 980M.
  - Minimum 16GB RAM, Recommended 32GB.
  - Minimum 80GB HDD, Recommended 128GB SSD.
- For Deployment 10,000 Request per Second.
  - Intel Processor Xeon.
  - Nvidia Tesla P100 8X.
  - 64GB RAM.
  - 256GB SSD.

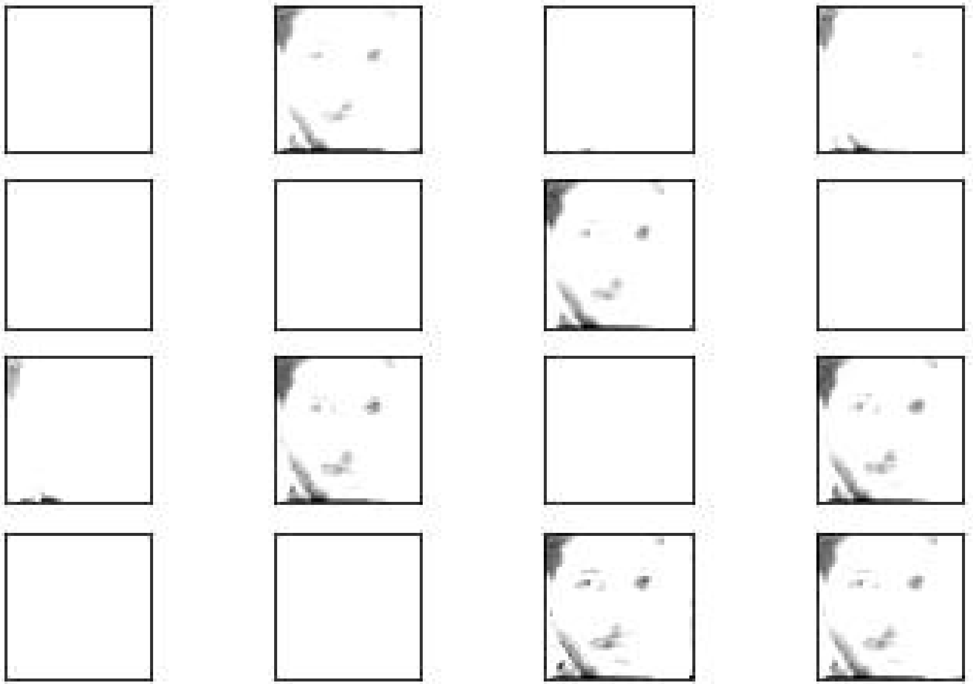
Software Requirement

- Nvidia CUDA Driver.
- Nvidia CUDNN.
- TensorFlow-GPU.
- TensorBoard.
- Python (3.5).
- NumPy.
- OpenCV.
- MatplotLib
- Jupyter-Notebook (development).

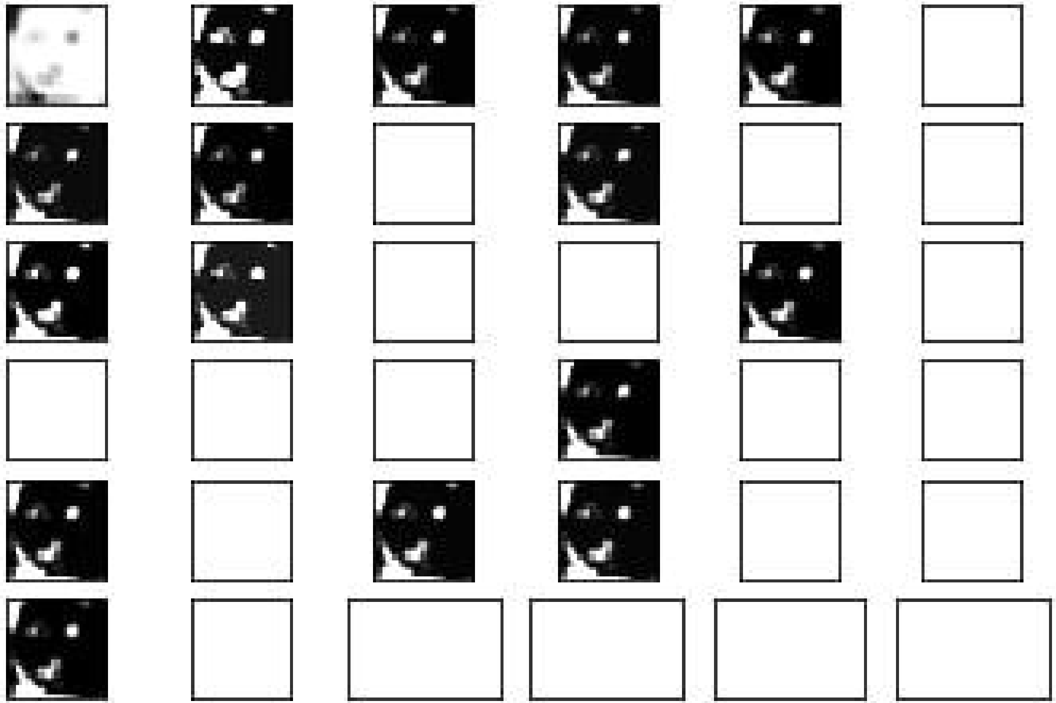
Flow Chart


accuracy
ArgMax_1
dimension
ArgMax
dimension
loss
gradients
Softmax
y_true
OutputLayer
Adam
init
gradients
FCLayer1
Adam
init
gradients
Reshape_1
shape
gradients
ConvLayer3
Adam
init
gradients
ConvLayer2
Adam
init
gradients
ConvLayer1
Adam
init
gradients
Reshape
shape
gradients

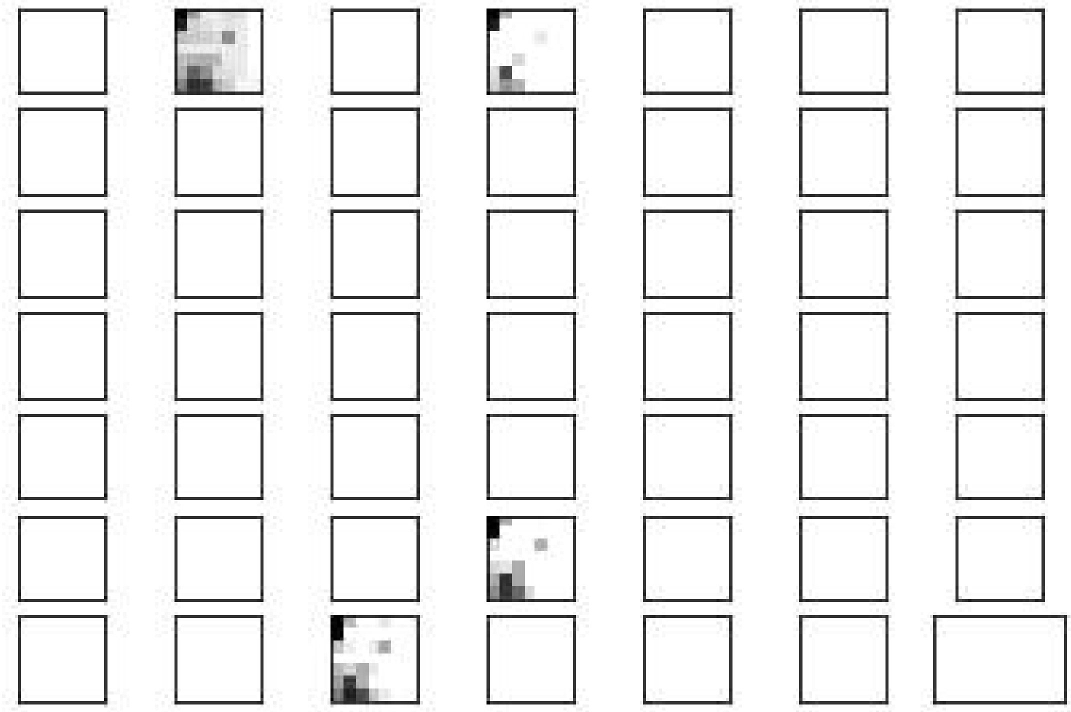
Screenshots-Convolution Layer 1



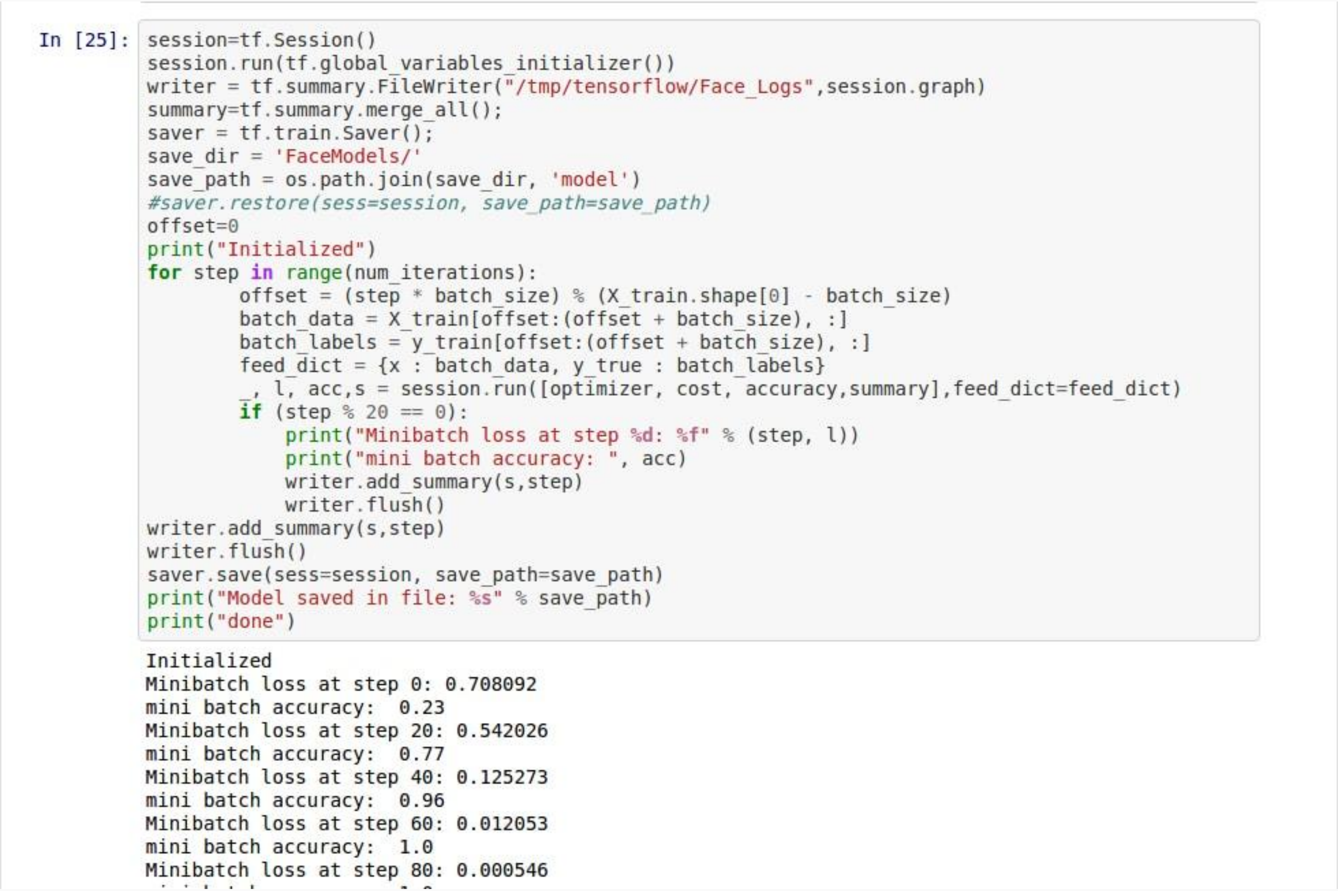
Convolutional Layer 2



Convolutional Layer 3



Code Segment- Full Code Link(https://github.com/raoger/faceProject)



```
In [25]: session=tf.Session()
session.run(tf.global_variables_initializer())
writer = tf.summary.FileWriter("/tmp/tensorflow/Face_Logs",session.graph)
summary=tf.summary.merge_all();
saver = tf.train.Saver();
save_dir = 'FaceModels/'
save_path = os.path.join(save_dir, 'model')
#saver.restore(sess=session, save_path=save_path)
offset=0
print("Initialized")
for step in range(num_iterations):
        offset = (step * batch_size) % (X_train.shape[0] - batch_size)
        batch_data = X_train[offset:(offset + batch_size), :]
        batch_labels = y_train[offset:(offset + batch_size), :]
        feed_dict = {x : batch_data, y_true : batch_labels}
        _, l, acc,s = session.run([optimizer, cost, accuracy,summary],feed_dict=feed_dict)
        if (step % 20 == 0):
            print("Minibatch loss at step %d: %f" % (step, l))
            print("mini batch accuracy: ", acc)
            writer.add_summary(s,step)
            writer.flush()
writer.add_summary(s,step)
writer.flush()
saver.save(sess=session, save_path=save_path)
print("Model saved in file: %s" % save_path)
print("done")
```

```
Initialized
Minibatch loss at step 0: 0.708092
mini batch accuracy:  0.23
Minibatch loss at step 20: 0.542026
mini batch accuracy:  0.77
Minibatch loss at step 40: 0.125273
mini batch accuracy:  0.96
Minibatch loss at step 60: 0.012053
mini batch accuracy:  1.0
Minibatch loss at step 80: 0.000546
```

Modules to be Completed

- Capture Image
- Prepare Dataset
- Wrapper

References

[1] G. Hu, Y. Yang, D. Yi, J. Kittler, W. Christmas, S. Z. Li, and T. Hospedales, “When face recognition meets with deep learning: An evaluation of convolutional neural networks for face recognition,” in 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), Dec 2015, pp. 384–392.

[2] Y. Sun, X. Wang, and X. Tang, “Hybrid deep learning for face verification,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 10, pp. 1997–2009, Oct 2016.