Deep Learning for Facial Recognition

Technical Report · May 2018

1 author:

Hrishikesh Kulkarni

Unconstrained Facial Recognition using Supervised Deep Learning on Video

Hrishikesh Kulkarni
G.Raymond Chang School of Continuing Education
Ryerson University
hkulkarni@ryerson.ca

Dr.Ghassem Tofighi
Instructor and Researcher
Data Science Lab
Ryerson University
gtofighi@ryerson.ca

Abstract :- Face recognition is the task of identifying an individual from an image of their face and a database of know faces. Despite being a relatively easy task for most humans, “unconstrained” face recognition by machines, specifically in settings such as malls, casinos and transport terminals, remains an open and active area of research. It has multiple use cases in surveillance, access control and even finding missing persons in a crowd.

However, in recent years, a large number of photos and videos have been crawled by search engines, and uploaded to social networks, which include a variety of unconstrained material, such as objects, faces and scenes. This large volume of data and the increase in computational resources have enabled the use of more powerful statistical models for general challenge of object classification in images and videos.

This research project evaluates the use of big data based machine learning approaches such as deep convolutional neural networks for the problem of unconstrained facial recognition in video data. It attempts to replicate and if feasible better performance of state of art leading commercial systems trained on large proprietary datasets, using public datasets and open source frameworks from research universities.

It is assumed that the reader has a fair understanding of neural networks and convolutional neural networks from a both theoretical and practical standpoint.

I. INTRODUCTION

This project attempts to reproduce the performance of state of art proprietary face recognition systems within video systems by using and tuning open source frameworks. For this purpose we will be utilizing today's state of art open datasets for face recognition within video. Primarily we use Youtube Faces DB Dataset . This consists of 3425 videos of 1595 subjects. The videos are broken down to frames and the face recognition is done on the frames (after doing an initial face alignment, as explained latter). The system is trained with Casia WebFaces which is a public dataset consisting of 10,575 subjects and 494,414 images

2. RELATED WORK

While most of the related work reviewed is provided within the references section, the key work referenced for this project was the Facenet system based on Inception Network Architecture from Google[5,6] for facial recognition and the Resent Architecture from Microsoft Research.[8]. Also key architecture referenced and used in this research is the combination of inception and Resnet architectures as described in [10], as it is shown to provide dramatic improvement in performance. The rest of the referenced work describes the research breakthroughs, such that led to widespread use of convolutional neural network architectures(LeNet[2],AlexNet[3], VGG Net[4], GoogleNet[5])as the state of art in technological approaches for the general challenge of object and image recognition.

For data pre-processing, primarily for face detection and alignment , to ensure pose invariant face recognition, the work referenced is Multi-Task CNN[7].

While not directly referenced, the core research papers that influenced the choice of the loss function for face recognition by the Facenet team , as well as the mathematical basis for use of deeper networks, have been mentioned.[11,12].

The two references for CASIA WebFaces and YouTube Faces datasets[13,14] describe how their respective datasets were created. The CASIA Webface team also gives a benchmark on Youtube Faces DB, which this research will try to match or better. The CASIA team achieves a best performance of 90.60 %.

Finally, the last two references [15,16] point to open source implementations of the OpenFace face recognition system from Carnegie Mellon University

3. DATA PREPROCESSING, NETWORK ARCHITECTURE AND TRAINING PARAMETERS

It was critical that both training and test sets were pre-processed to extract the face, using the same approach.

The video were preprocessed for to face detection and alignment using open source implementation of the multi-task CNN algorithm [7]. This approach is known to be invariant to poses, illuminations and occlusions and gives better results than the standard dlib library used for this purpose. The below figure showing the three stage multi-task CNN process has been extracted from the associated reference paper[7].

The open source implementation was already pre-trained and hence there was no need to do any training before using it on both the training and test sets. The image dimensions after extraction were 160x160.

).

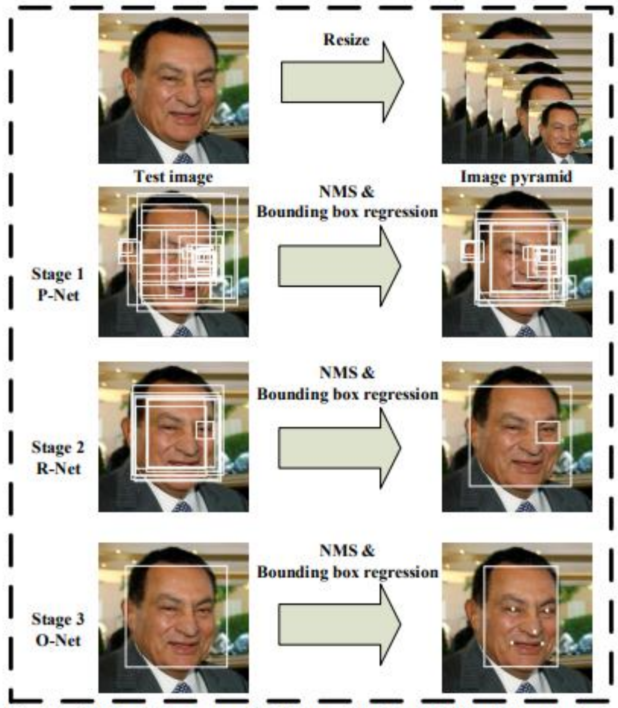


Fig. 1. Pipeline of our cascaded framework that includes three-stage multi-task deep convolutional networks. Firstly, candidate windows are produced through a fast Proposal Network (P-Net). After that, we refine these candidates in the next stage through a Refinement Network (R-Net). In the third stage, The Output Network (O-Net) produces final bounding box and facial landmarks position.

The network architectures chosen were variants of Google Inception and Microsoft Research Resnet Architecture.
Two types of architecture are explored. One is small Inception Network and the other is Inception Resnet Network (v1 and v2).

They are explained below:-

3.1 Small Inception Network (Google Inception NN4)

One of the networks used was NN4 as described in the Google Facenet paper[6]. The above figure depicting the NN4 layers has been extracted from this paper.
This was based on the Openface[16] implementation of NN4 , which does not include the layers 4c and 4d.

type	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj (p)	params	FLOPS
conv1 (7×7×3,2)	112×112×64	1							9K	119M
max pool + norm	56×56×64	0						m 3×3, 2		
inception (2)	56×56×192	2		64	192				115K	360M
norm + max pool	28×28×192	0						m 3×3, 2		
inception (3a)	28×28×256	2	64	96	128	16	32	m, 32p	164K	128M
inception (3b)	28×28×320	2	64	96	128	32	64	L_2, 64p	228K	179M
inception (3c)	14×14×640	2	0	128	256,2	32	64,2	m 3×3,2	398K	108M
inception (4a)	14×14×640	2	256	96	192	32	64	L_2, 128p	545K	107M
inception (4b)	14×14×640	2	224	112	224	32	64	L_2, 128p	595K	117M
inception (4c)	14×14×640	2	192	128	256	32	64	L_2, 128p	654K	128M
inception (4d)	14×14×640	2	160	144	288	32	64	L_2, 128p	722K	142M
inception (4e)	7×7×1024	2	0	160	256,2	64	128,2	m 3×3,2	717K	56M
inception (5a)	7×7×1024	2	384	192	384	48	128	L_2, 128p	1.6M	78M
inception (5b)	7×7×1024	2	384	192	384	48	128	m, 128p	1.6M	78M
avg pool	1×1×1024	0								
fully conn	1×1×128	1							131K	0.1M
L2 normalization	1×1×128	0								
total									7.5M	1.6B

The model produced a 128 byte embedding of each person. The value of the various parameters, specially the triplet loss margin(0.2),number of epochs(1000), number of batches per epoch(250), number of images per person(20), number of persons per batch(15), batch size(800), image size(96x96) were chosen as per default value within the OpenFace implementation. This implementation, which was primarily for measuring accuracy over public image dataset LFW(Labelled Faced in the Wild), was repurposed in this project for training and testing over Youtube Faces DB.

Gradient descent with momentum is used to train the network. Further, triplet loss, as described in Google Facenet paper [6] is used to select the right triplet of positive and negative images related to an identity for training. Triplet loss is described more in Section 4.

Additionally batch normalizations are used during training.

3.2 Inception-Resnet Architecture.

The other network used was a deeper network, namely the Inception-Restnet network v1 and v2. This combines ideas from both Google Inception network and Microsoft Resnet network.

The architectures are given below:-

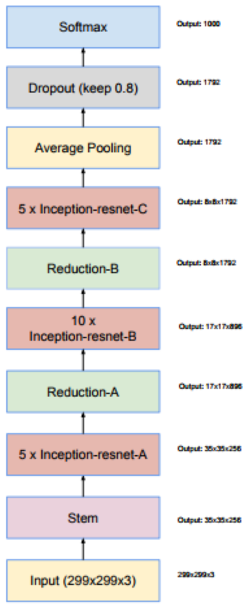
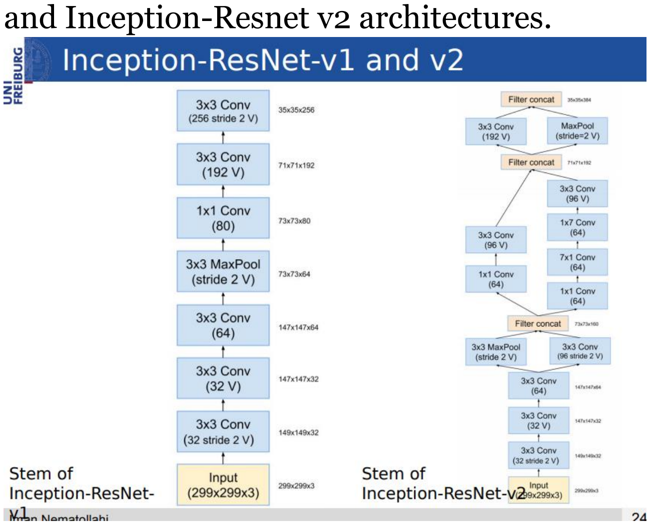
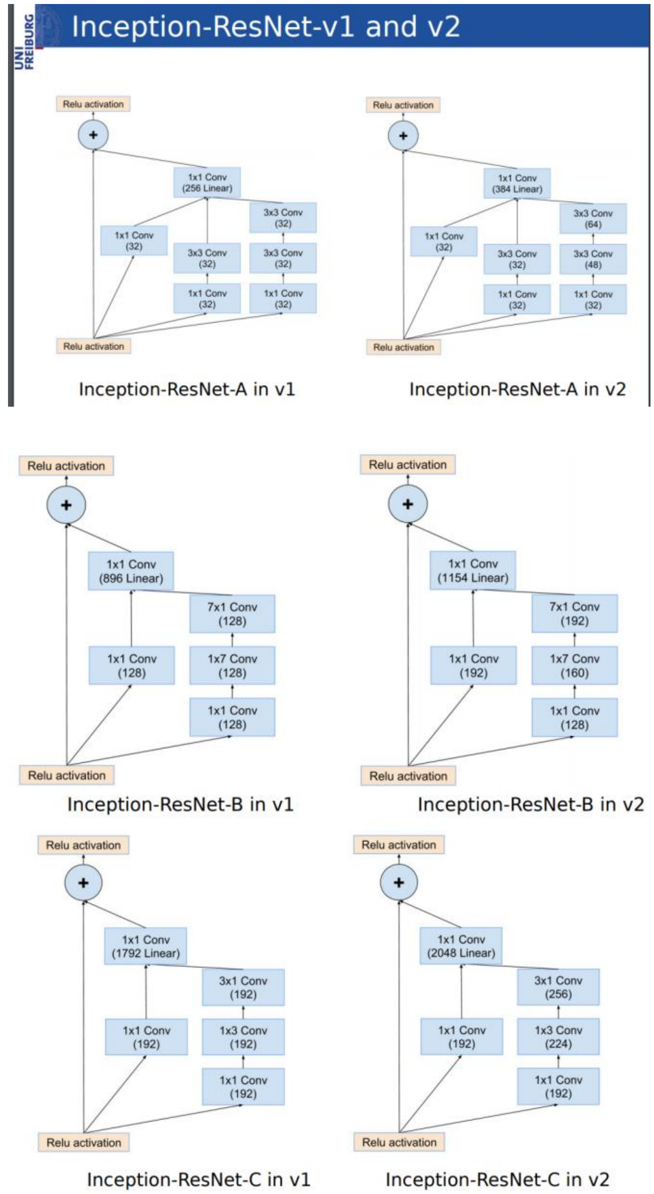
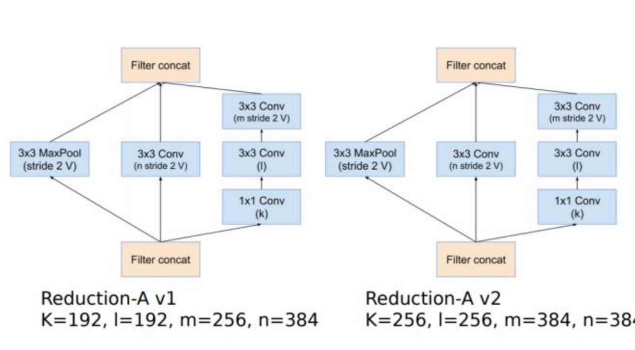


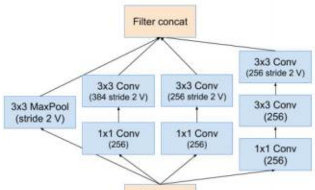
Figure 15. Schema for Inception-ResNet-v1 and Inception-ResNet-v2 networks. This schema applies to both networks but the underlying components differ. Inception-ResNet-v1 uses the blocks as described in Figures 14, 10, 7, 11, 12 and 13. Inception-ResNet-v2 uses the blocks as described in Figures 3, 16, 7,17, 18 and 19. The output sizes in the diagram refer to the activation vector tensor shapes of Inception-ResNet-v1.

The above figure is extracted from the reference work[10] which further details the individual blocks within Inception-Resent v1 and Inception-Resnet v2 architectures.

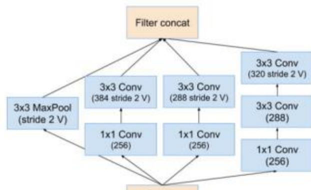








Reduction-B v1



Reduction-B v2

Below are the default values of parameters and hyperparameters during training.

Maximum Number of epochs: 100
Number of Images in a batch: - 90
Number of persons per batch: 45
Number of images per person: 40
Number of batches per epoch: 1000
Triplet margin :- 0.2
Learning Algorithm:- RMSPROP
Learning Rate:0.01
Learning Rate decay factor: 1.0
Number of epochs between learning rate decay: 100
parameter weight decay: 1e-4
Final size of image embedding:- 128 bytes

4. Training

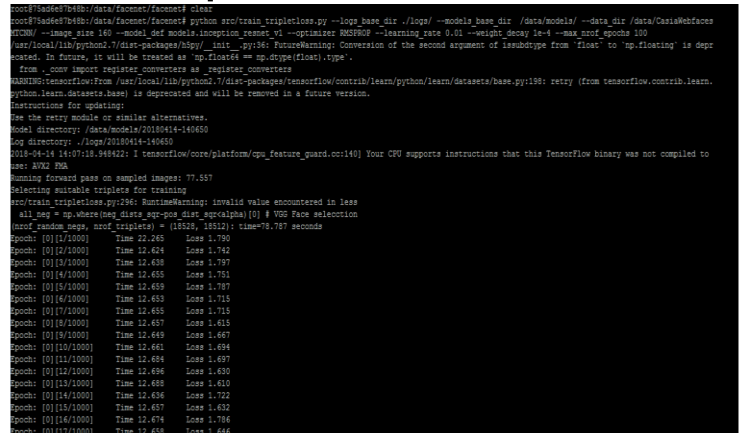
For the small inception network (NN4), pre trained models from Openface implementation were used and there was no additional training performed.

During training both Inception-Resent v1 and Inception-Resnet v2 networks were used. It was found that Inception-Resnet v1 gives less training loss from the start.

The choice of learning rate and gradient descent algorithm did not seem to impact the training. Specially there was no different between the choice of ADAM and RMSPROP during learning.

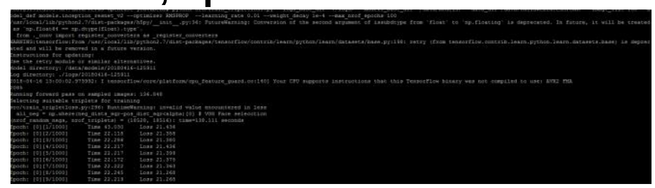
Below images show the loss , based on choice of different training parameters and hyperparameters.:-

i.)Inception-resnet v1, learning rate:- 0.01, optimizer RMSPROP



```
root@75ad6e87b48b:/data/facenet/facenet# clear
root@75ad6e87b48b:/data/facenet/facenet# python src/train_tripletloss.py --logs_base_dir ./logs/ --models_base_dir  /data/models/ --data_dir /data/CasiaWebfaces
MTCNN/ --image_size 160 --model_def models.inception_resnet_v1 --optimizer RMSPROP --learning_rate 0.01 --weight_decay 1e-4 --max_nrof_epochs 100
/usr/local/lib/python2.7/dist-packages/h5py/__init__.py:36: FutureWarning: Conversion of the second argument of issubdtype from `float` to `np.floating` is depr
ecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
WARNING:tensorflow:From /usr/local/lib/python2.7/dist-packages/tensorflow/contrib/learn/python/learn/datasets/base.py:198: retry (from tensorflow.contrib.learn.
python.learn.datasets.base) is deprecated and will be removed in a future version.
Instructions for updating:
Use the retry module or similar alternatives.
Model directory: /data/models/20180414-140650
Log directory: ./logs/20180414-140650
2018-04-14 14:07:18.948422: I tensorflow/core/platform/cpu_feature_guard.cc:140] Your CPU supports instructions that this TensorFlow binary was not compiled to
use: AVX2 FMA
Running forward pass on sampled images: 77.557
Selecting suitable triplets for training
src/train_tripletloss.py:296: RuntimeWarning: invalid value encountered in less
  all_neg = np.where(neg_dists_sqr-pos_dist_sqr<alpha)[0] # VGG Face selecction
(nrof_random_negs, nrof_triplets) = (18528, 18512): time=78.787 seconds
Epoch: [0][1/1000]      Time 22.265     Loss 1.790
Epoch: [0][2/1000]      Time 12.624     Loss 1.742
Epoch: [0][3/1000]      Time 12.638     Loss 1.797
Epoch: [0][4/1000]      Time 12.655     Loss 1.751
Epoch: [0][5/1000]      Time 12.659     Loss 1.787
Epoch: [0][6/1000]      Time 12.653     Loss 1.715
Epoch: [0][7/1000]      Time 12.655     Loss 1.715
Epoch: [0][8/1000]      Time 12.657     Loss 1.615
Epoch: [0][9/1000]      Time 12.649     Loss 1.667
Epoch: [0][10/1000]     Time 12.661     Loss 1.694
Epoch: [0][11/1000]     Time 12.684     Loss 1.697
Epoch: [0][12/1000]     Time 12.696     Loss 1.630
Epoch: [0][13/1000]     Time 12.688     Loss 1.610
Epoch: [0][14/1000]     Time 12.636     Loss 1.722
Epoch: [0][15/1000]     Time 12.657     Loss 1.632
Epoch: [0][16/1000]     Time 12.674     Loss 1.786
```

ii.)Inception-resnet V2, learning rate : 0.01, optimizer RMSPROP



iii.)Inception-resnet V2, learning rate:- 0.01, optimizer ADAM

The training was accordingly continued on better performing Inception-Resnet v1 network , for two epochs on Amazon cloud VM instance that utilized NVIDIA Tesla v100 GPU.

4.1 Triplet Loss

The 128 byte embedding is used to calculate triplet loss . Description of triplet extracted from Facenet paper [6] is provided below.

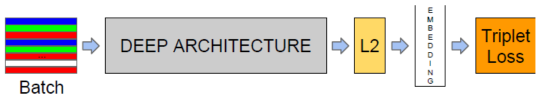


Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

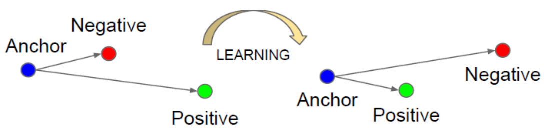


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

The embedding is represented by $f(x) \in \mathbb{R}^d$. It embeds an image x into a d-dimensional Euclidean space. Additionally, we constrain this embedding to live on the d-dimensional hypersphere, *i.e.* $\|f(x)\|_2 = 1$. This loss is

Here we want to ensure that an image x_i^a (*anchor*) of a specific person is closer to all other images x_i^p (*positive*) of the same person than it is to any image x_i^n (*negative*) of any other person. This is visualized in Figure 3.

Thus we want,

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2 , \tag{1}$$

$$\forall \left(f(x_i^a), f(x_i^p), f(x_i^n)\right) \in \mathcal{T} . \tag{2}$$

where α is a margin that is enforced between positive and negative pairs. $\mathcal{T}$ is the set of all possible triplets in the training set and has cardinality N.

The loss that is being minimized is then $L =$

$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ .$$

5. Testing

For testing of pretrained model of small NN4 network, a small DevTest subset(10%) of YoutubeDB test set was used first.

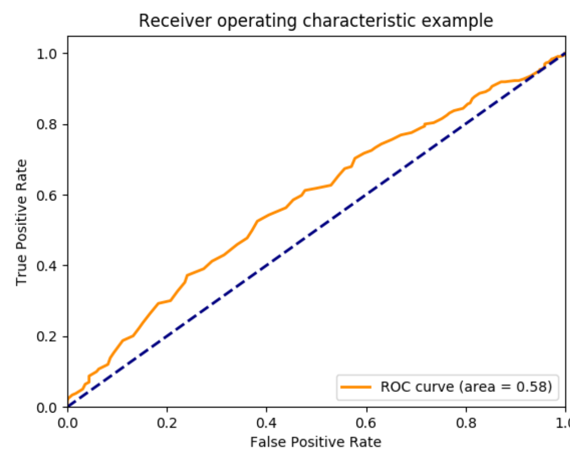
For testing of models, trained on Inception-Resnet v1 network, the test set of Youtube Faces DB was divided into a smaller Test-Dev Set and a much larger Test set. Additionally while testing in the larger Test set cross validation was used.

6. Results

The Accuracy and ROC (receiver operator characteristics) of the tests are shown below

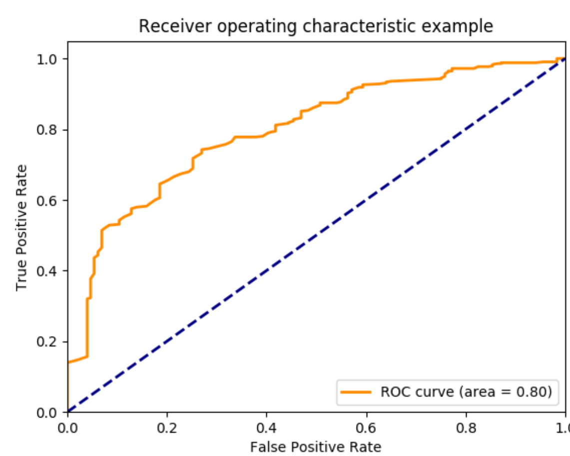
6.1 Result on DevTest set using small NN4 network

Accuracy obtained was 54.3 +- 6.2 %

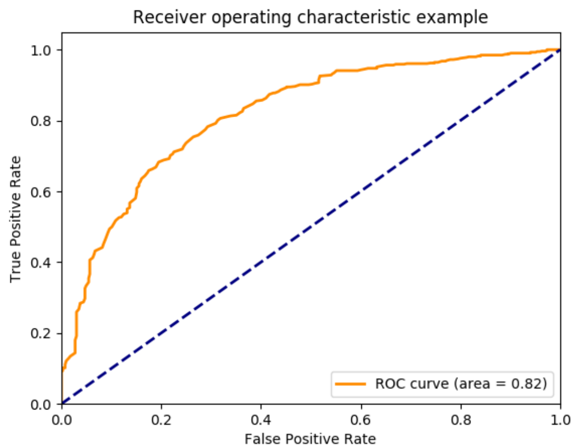


6.2 Result on DevTest set using trained Inception-ResNet v1 network

Accuracy obtained was 84.3 % +- 6.8%



6.3 Result on Youtube DB Test sert using Inception-Resnet v1 network Accuracy obtained was 73 % +- 4.7%



It is clear form above ROC curves, that the smaller NN network performs much more poorly compared with the deeper Inception-Resnet networks.
Furthermore for the Inception-Resnet networks the accuracy rate is high with just two epochs of training.

7. Conclusion

The research and tests done so far clearly indicate that deeper Inception-Resnet networks perform better than smaller networks. Further work on training network for more epochs and tuning parameters is needed to determine the optimal settings to obtain best results.

Acknowledgment

We would like to thank the Data Science department at Ryerson University for the guidance and suggestions provided during this shorth term research project, which was done as part of a Capstone project for Big Data and Predictive Analytics certification course. We also acknowledge the contributions made by research as listed in references and on which our own research is built. Lastly, we would also like to thank Professor Andrew Ng and his team from Coursera who have put together one of the best courses on deep learning that starts from the basic mathematical foundations and builds upto the latest research in this field.

References

[1] **F. Rosenblatt (1958) :** The Perceptron: A Probabilistic Model For Information Storage and Organization in the Brain. https://pdfs.semanticscholar.org/865f/b2cfe6fdb7af2c663ef346ea05889f237108.pdf

[2] **Yann Le Cun, Yoshua Bengio, Patrick Haffner:** Gradient based learning applied to document recognition(1998) http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf

[3] **AlexKrizhevsky,Ilya Sutskever,Geoffrey E. Hinton:** Imagenet Classification with deep convolutional neural networks https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[4] **Christian Szeged, Wei Liu,Yangqing Jia:** Very Deep Convolutional Neural Networks for Large Scale Image Recognition (VGGNet) https://arxiv.org/pdf/1409.4842.pdf

[5] **Christian Szegedy, Wei Liu, Yangqing Jia:** Going Deeper with Convolutions (Inception Network-GoogLenet)\ https://arxiv.org/pdf/1409.4842.pdf

[6] **Florian Schroff, Dmitry Kalenichenko , James Philbin Google Inc.:** Facenet Unified Embedding for Recognition and Clustering.pdf

https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Schroff_FaceNet_A_Unified_2015_CVPR_paper.pdf

[7] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li,:
Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks
https://kpzhang93.github.io/MTCNN_face_detection_alignment/paper/spl.pdf

[8] Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun Microsoft Research:
Deep Residual Learning for Image Recognition
https://arxiv.org/pdf/1512.03385.pdf

[9] Professor Andrew NG et al:
https://www.coursera.org/specializations/deep-learning

[10] Christian Szegedy Google Inc, Sergey Ioffe,Vincent Vanhoucke:
Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning
https://arxiv.org/pdf/1602.07261.pdf
https://pdfs.semanticscholar.org/73ac/009051bba99eaea799172b28d69168b6aa02.pdf

[11] Kilian Q. Weinberger, Lawrence K. Saul:
Distance Metric Learning for Large Margin Nearest Neighbour Classification
http://jmlr.csail.mit.edu/papers/volume10/weinberger09a/weinberger09a.pdf

[12] Sanjeev Arora, Aditya Bhaskara, Rong Ge, Tengyu Ma:
Provable bounds for learning deep representations
https://arxiv.org/pdf/1310.6343.pdf

[13] Dong Yi, Zhen Lei, Shengcai Liao and Stan Z. Li:
Learning Face representation from scratch
https://arxiv.org/pdf/1411.7923.pdf

[14] Lior Wolf1 Tal Hassner2 Itay Maoz1:
Faced recognition in unconstrained video with matched background similarity
http://www.cs.tau.ac.il/~wolf/ytfaces/WolfHassnerMaoz_CVPR11.pdf

[15] David SandBerg:
https://github.com/davidsandberg/facenet
Openface implementation using Python and Tensorflow and using Inception-Resnet v1 and v2 network architectures

[16] Brandon Amos, Bartosz Ludwiczuk,† Mahadev Satyanarayanan:
Openface: A general-purpose face recognition library with mobile applications
http://reports-archive.adm.cs.cmu.edu/anon/2016/CMU-CS-16-118.pdf

[17] Victor Sy Wang:
Openface Implementation using Python,Openkeras and tensorflow and using small NN4 Inception network
: https://github.com/iwantooxxoox/Keras-OpenFace.