News Image Classification

CS 640 Project

Paritosh Shirodkar (U98006575)

Suryateja Gudiguntla (U50264201)

Problem Identification

In this project, we were given a set of news images related to different kinds of events. The images are labeled with whether something occurred in the image, such as violence, fire, and a protest. An image can have several labels if a scene contains both a protest and violence, and it is possible an image has no label if it is not related to any events of interest to be labeled.

We developed a visual model which can recognize protesters, describe their activities by visual attributes and estimate the level of perceived violence in an image.

There are approximately 40,000 images in the dataset and 32,000 for model training and 8,000 for model testing.

In the annotation, the categories are:

protest:	1 if the image depicts a protest
sign:	1 if the image has signs
photo:	1 if the image has one/more photo(s)
fire:	1 if the image depicts a fire
police:	1 if the image indicates the presence of police
children:	1 if the image indicates the presence of children
group_20:	1 if the image depicts the number of people in it is more than 20
group_100:	1 if the image depicts the number of people in it is more than 100
flag:	1 if the image indicates the presence of flags
night:	1 if the image depicts night time
shouting:	1 if the image indicates the presence of shouting by people
violence:	The value is between 0 and 1 if the image indicates violence. The higher the number more the violence

Background Investigation

1. Baseline Results for Violence Detection in Still Images

This paper establishes a new database containing 500 violence images and 1500 non-violence images. It uses the Bag-of-Words (BoW) model to discriminate violence images and non-violence images.

This approach can only identify if an image has violence in it or not, and does not help us with identifying the other features of the image.

2. Image Classification using Deep Neural Network

This example shows how to classify images into 10 classes, by training the classes using a Convolutional Neural Network on CIFAR-10 dataset. It uses 3 convolutional layers and maintained a gradient between them.

However, this approach would only classify one image as one class, and cannot provide the 10+ classes for each of our images. But this approach helped us to decide to use CNNs as the direction for the task.

3. Protest Activity Detection and Perceived Violence Estimation from Social Media Images

This example uses resnet50 model to recognize all the required classes from each image based on the visual attributes. It also estimates a level of perceived violence based on these visual attributes.

We decide to use this as our baseline.

Baseline Reproduction

The baseline used: [Protest detection violence estimation]

Performed the following to reproduce the baseline:

- Debugged the errors in the existing code.
- Updated the code to make it compatible with current versions of the dependencies.
- Loaded the dataset in a structure suitable for the project.

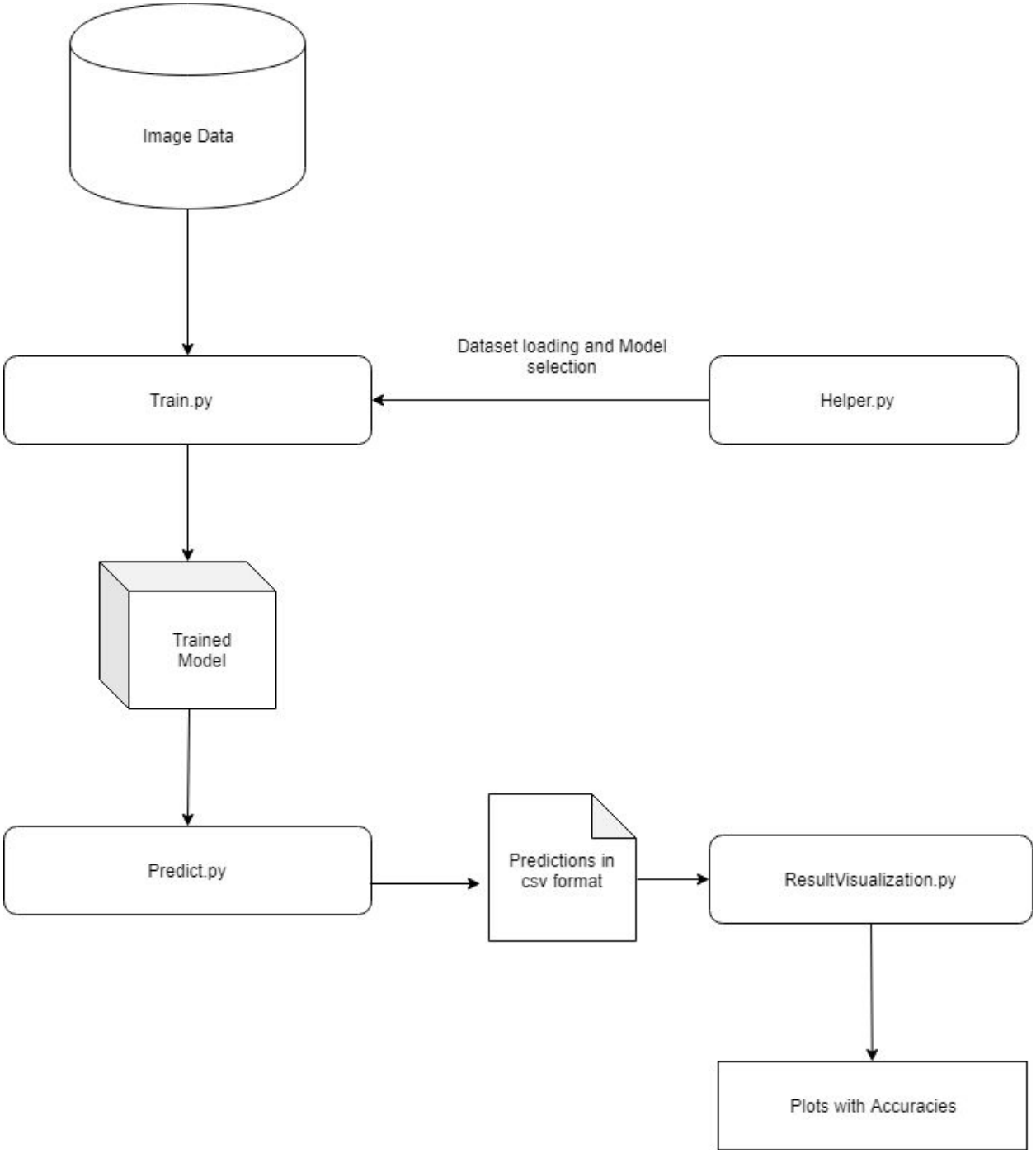
Reason for this selection:

CNN: Convolutional Neural Networks act as the perfect basis for image classification with our requirements as in CNNs, instead of feeding the entire image as an array of numbers, the image is broken up into a number of tiles, the machine then tries to predict what each tile is. Finally, the computer tries to predict what's in the picture based on the prediction of all the tiles. This allows the computer to parallelize the operations and detect the object regardless of where it is located in the image.

Resnet: In general, in a deep convolutional neural network, several layers are stacked and are trained to the task at hand. The network learns several low/mid/high level features at the end of its layers. In residual learning, instead of trying to learn some features, we try to learn some residual. Residual can be simply understood as subtraction of feature learned from the input of that layer. ResNet does this using the shortcut connections (directly connecting the input of the nth layer to some (n+x)th layer.

Resnet50: The baseline we selected makes use of Resnet50. This is a 50 layer residual network. We use TorchVision's implementation of Resnet50.

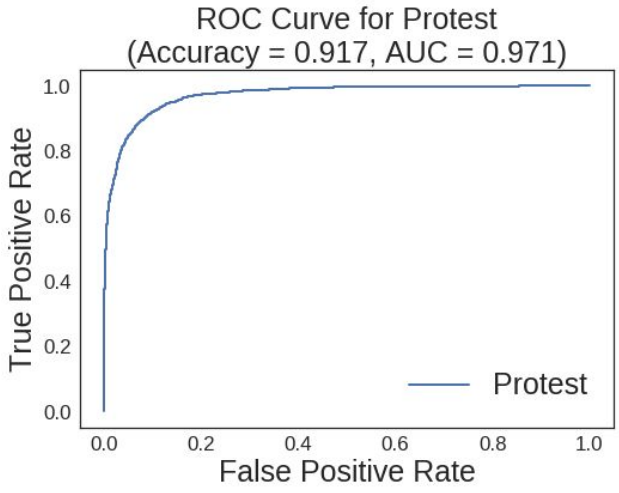
Workflow of the baseline:

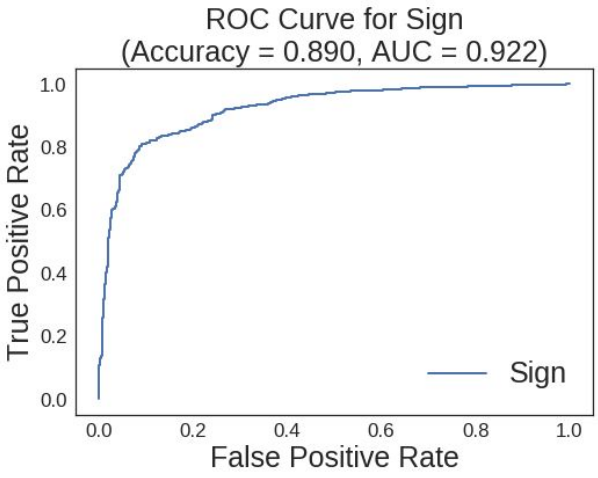


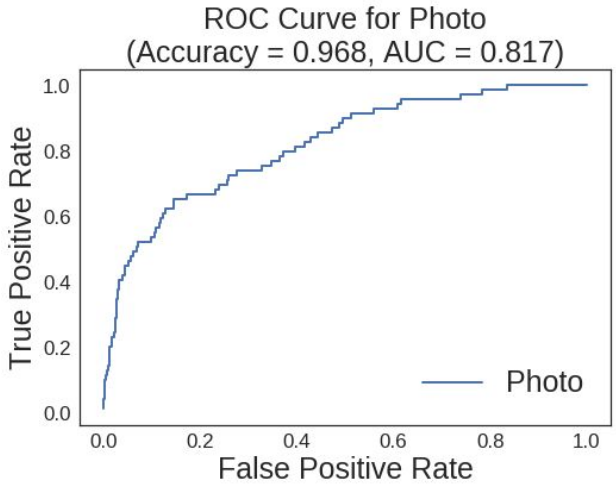
Baseline Code explanation:

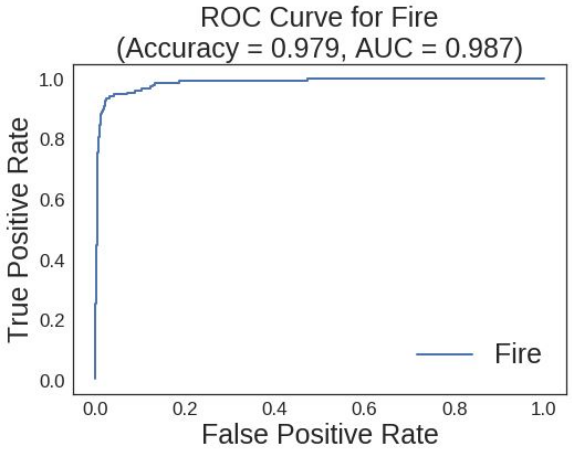
- Images are loaded using ProtestDataset() function defined in Helper.py
- These images are to be formatted to be adapted to the Resnet50 model using modifiedresnet50() in Helper.py. This involves using Lighting() and transformed into tensors to input to the model.
- We added a final layer using Sigmoid function to map it to the final 12 categories.
- We transfer the weights to our images by using transfer learning, using the pre-trained model of resnet50 trained on ImageNet.
- Validation is performed on every trained model and results are generated using CalculateLoss() function.
- This outputs a trained model in the form of a .pth.tar file.
- This is fed as input to Pred.py which generates a .csv file storing predictions for the test data.
- To visualize these results, Result Visualization.py is used to generate roc curves for all the classes and also a correlation plot between features and violence.

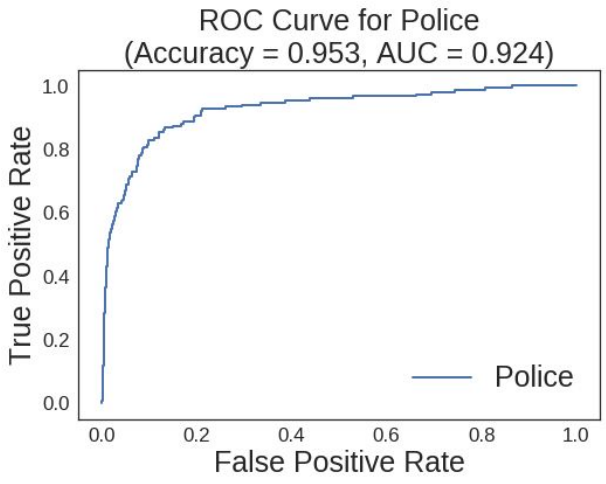
Baseline results

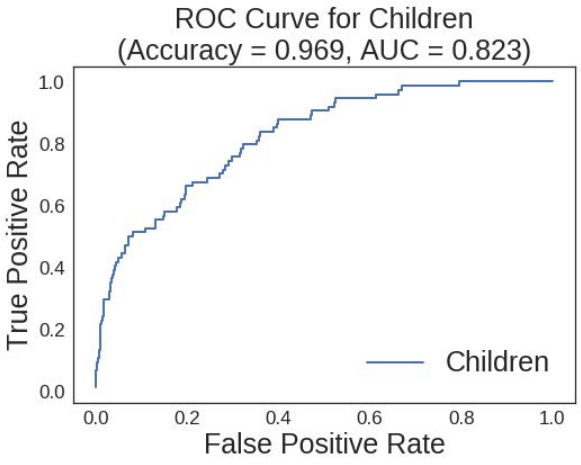


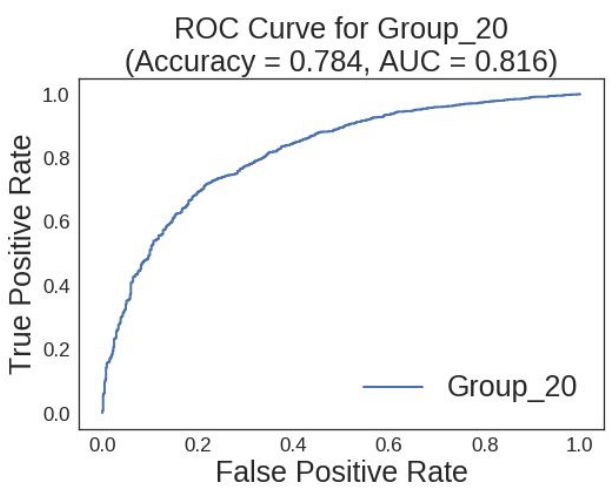


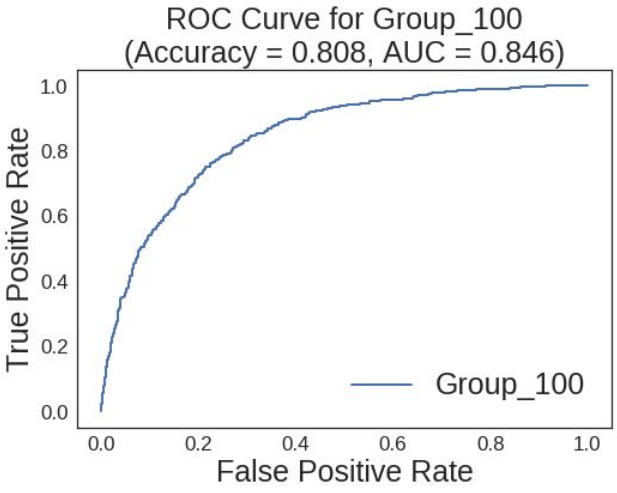


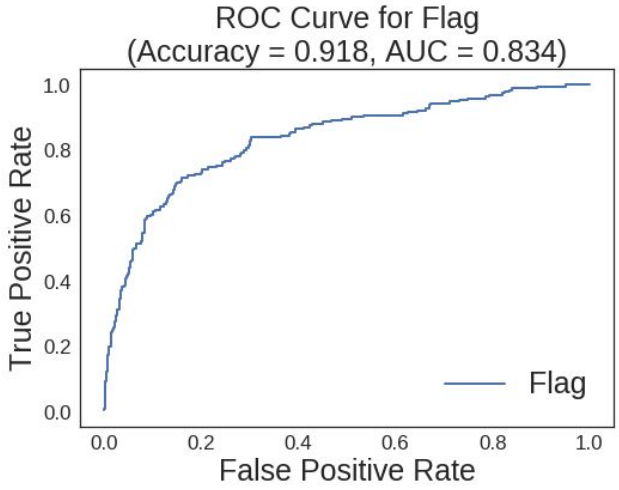


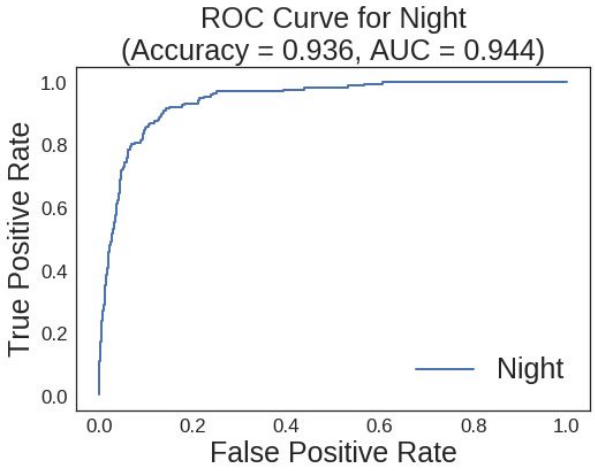


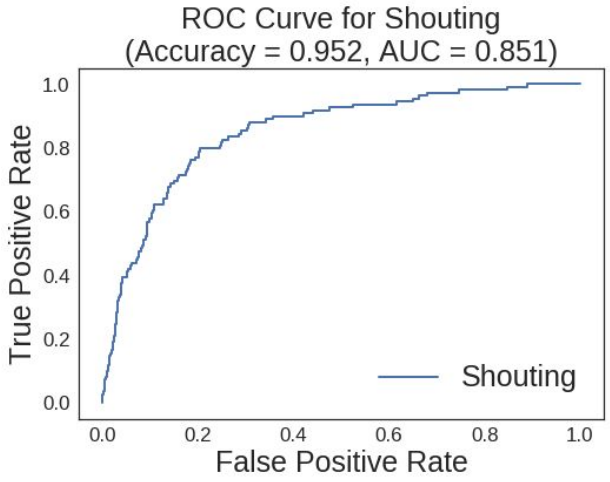


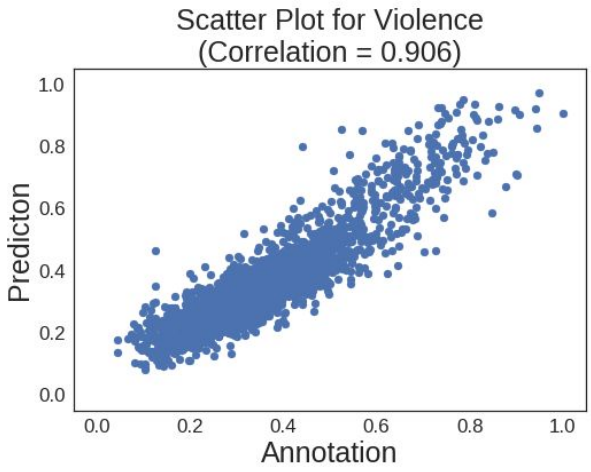

ROC Curve for Group_20
(Accuracy = 0.784, AUC = 0.816)
True Positive Rate
False Positive Rate
Group_20
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curve for Group_100
(Accuracy = 0.808, AUC = 0.846)
True Positive Rate
False Positive Rate
Group_100
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curve for Flag
(Accuracy = 0.918, AUC = 0.834)
True Positive Rate
False Positive Rate
Flag
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curve for Night
(Accuracy = 0.936, AUC = 0.944)
True Positive Rate
False Positive Rate
Night
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curve for Shouting
(Accuracy = 0.952, AUC = 0.851)
True Positive Rate
False Positive Rate
Shouting
0.0
0.2
0.4
0.6
0.8
1.0


Scatter Plot for Violence
(Correlation = 0.906)
Predicton
Annotation
0.0
0.2
0.4
0.6
0.8
1.0

Methods for improving baseline

1. Change the learning rate to 0.001 from 0.01

The lower the value, the slower we travel along the downward slope. We wanted to make sure that we do not miss any local minima, and since we use SGD optimizer in the baseline model, the learning rate is just adapted from the previous iterations and where it converged in the previous iteration.

2. Freezing all weights before the final layer

Rather than training the whole network through a random initialization of weights, we can use the weights of the pre-trained model (and freeze them) and focus on the more important layers (ones that are higher up) for training. This is usually the best method for transfer learning to increase the speed of learning.

3. Freezing weights of four layers before the final layer

If you don't want to modify the weights of a layer, the backward pass to that layer can be completely avoided, resulting in a significant speed boost. For e.g., if half your model is frozen, and you try to train the model, it will take about half the time compared to a fully trainable model. On the other hand, you still need to train the model, so if you freeze it too early, it will give inaccurate predictions. Hence we decided to freeze only four layers.

4. Adam Optimizer instead of SGD

Adaptive Moment Estimation (Adam) is another method that computes adaptive learning rates for each parameter. The authors of this paper show empirically that Adam works well in practice and compares favorably to other adaptive learning-method algorithms.

5. Resnet101 instead of Resnet50

We tried using a residual network with 101 layers instead of 50 layers. This is provided in torchvision's models. We decided to use this to check if additional layers help the training without overfitting.

Visual Results: Compare accuracy values

Accuracy:

Legend:

1: Baseline (resnet50 with lr=0.01)

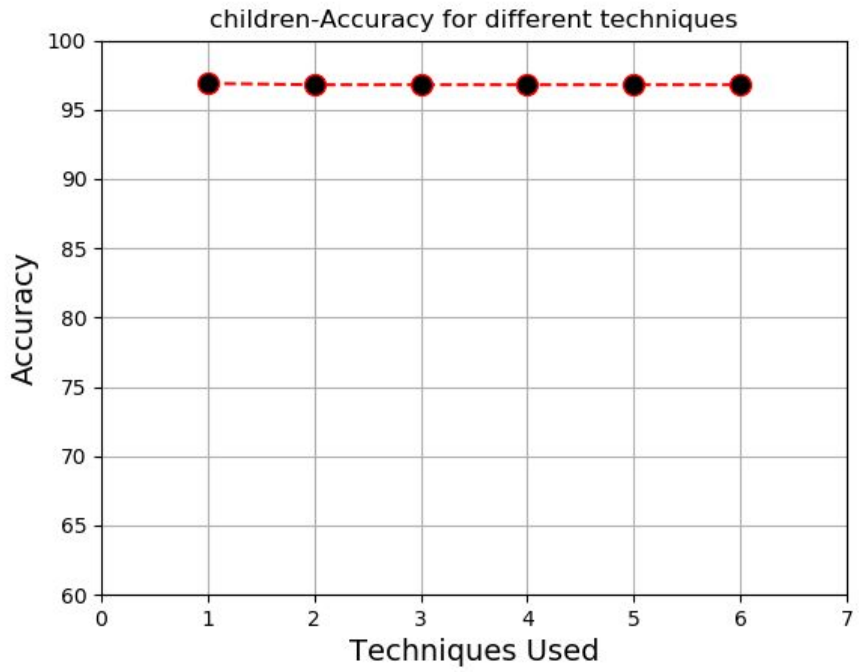
2: Resnet50 with lr=0.001

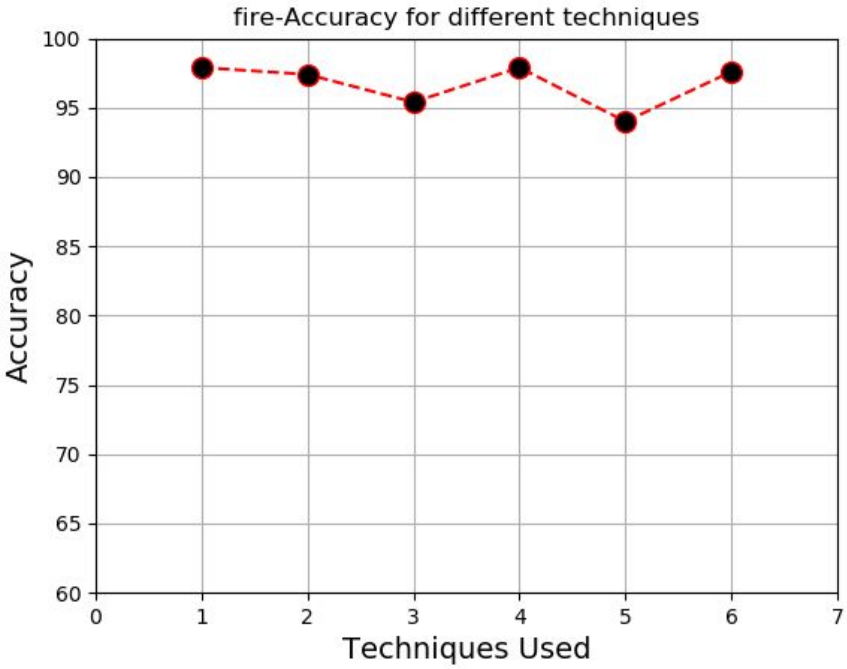
3: Freeze all layers

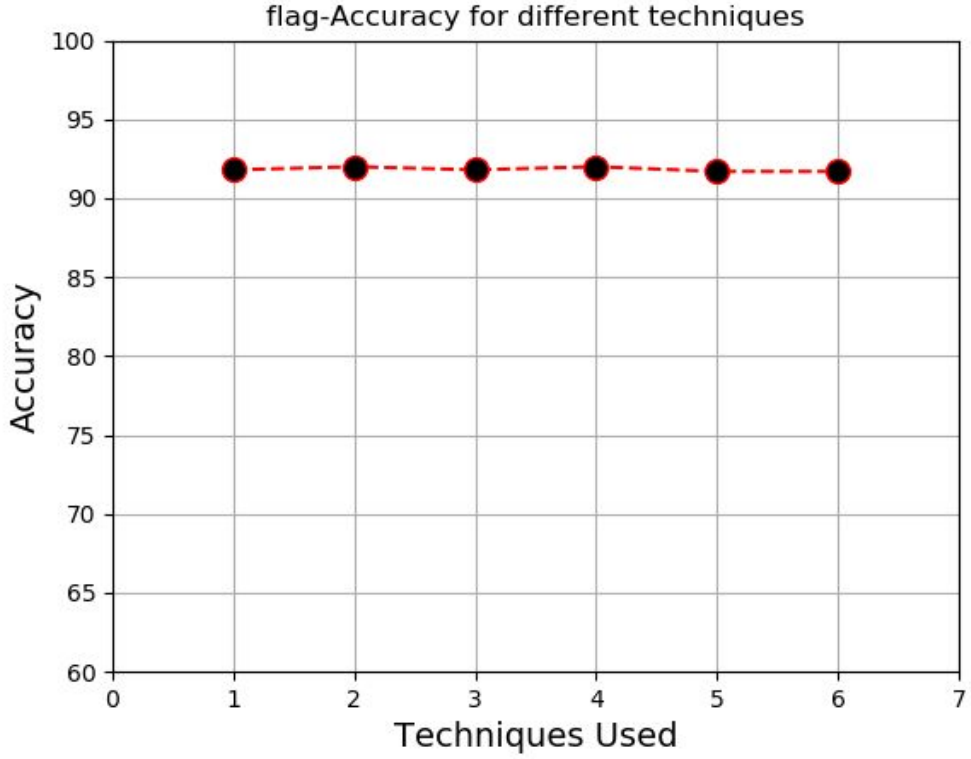
4: Freeze 4 layers

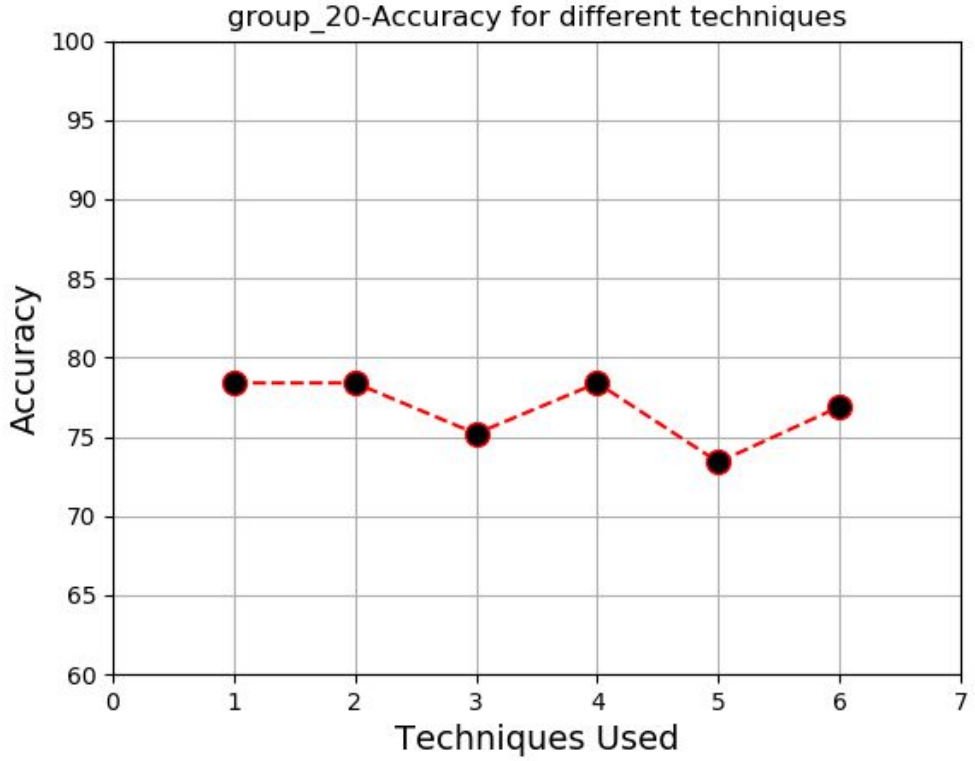
5: Adam Optimizer

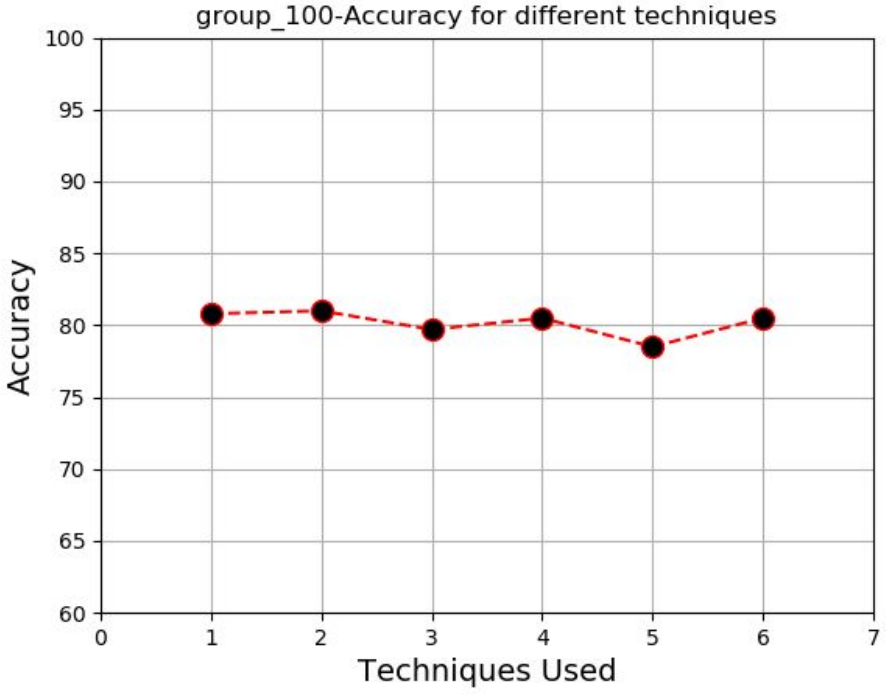
6: Resnet101

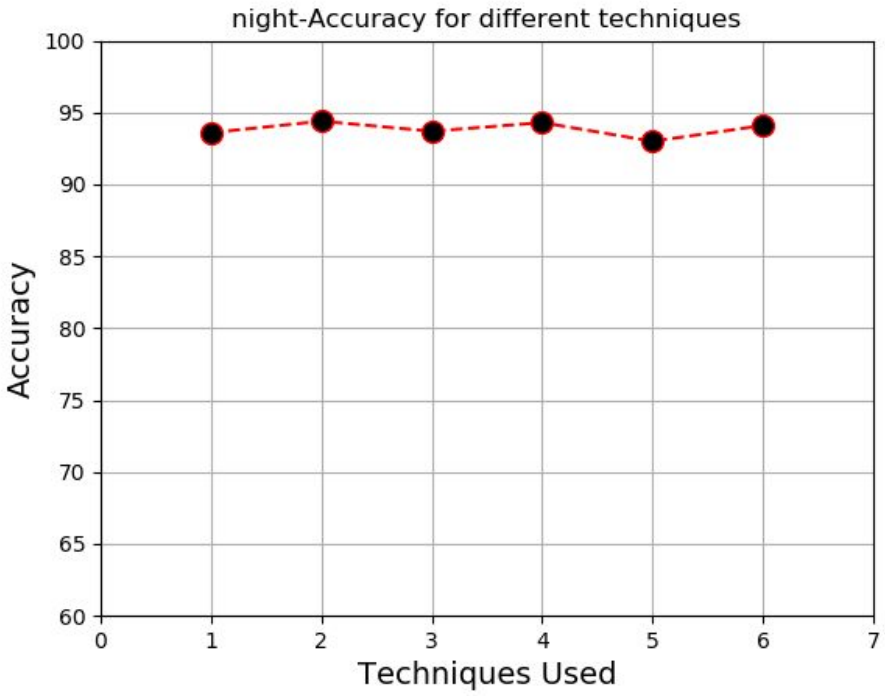



fire-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7


flag-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7


group_20-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7


group_100-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7


night-Accuracy for different techniques
100
95
90
85
80
75
70
65
60
Accuracy
0
1
2
3
4
5
6
7
Techniques Used

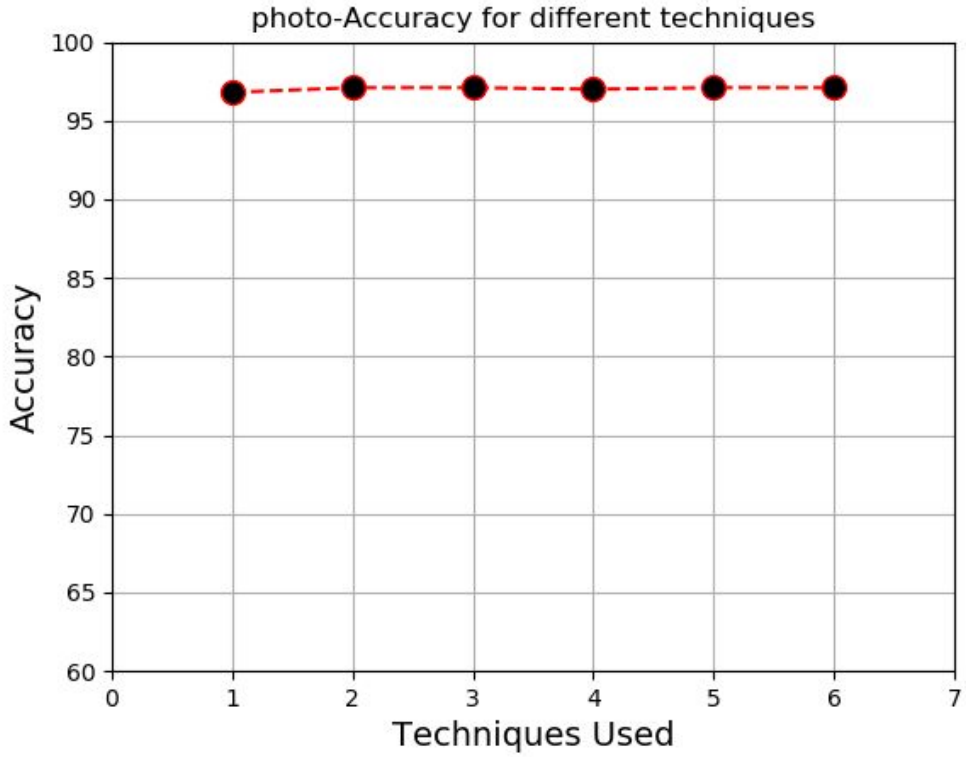
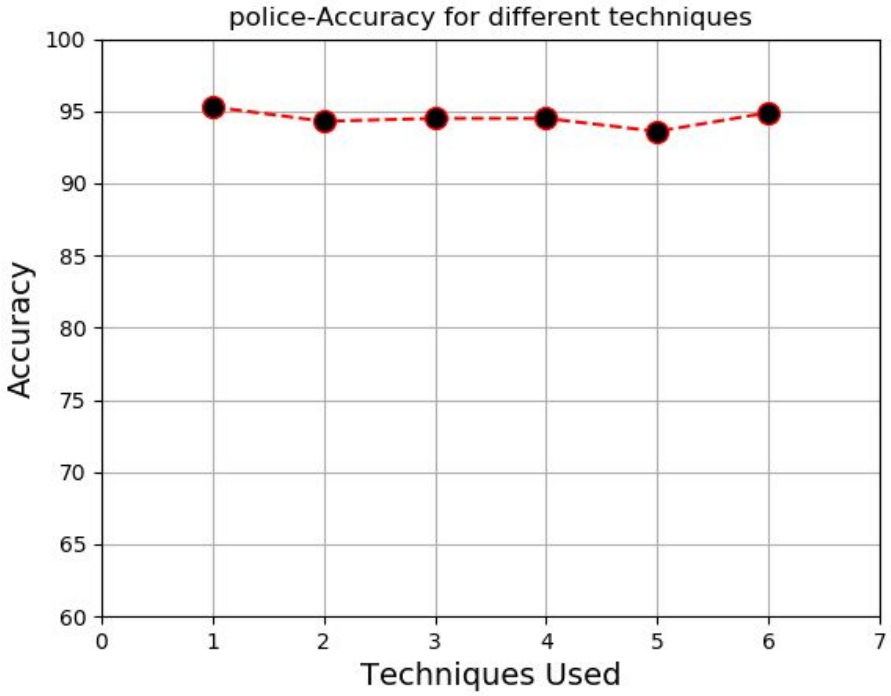
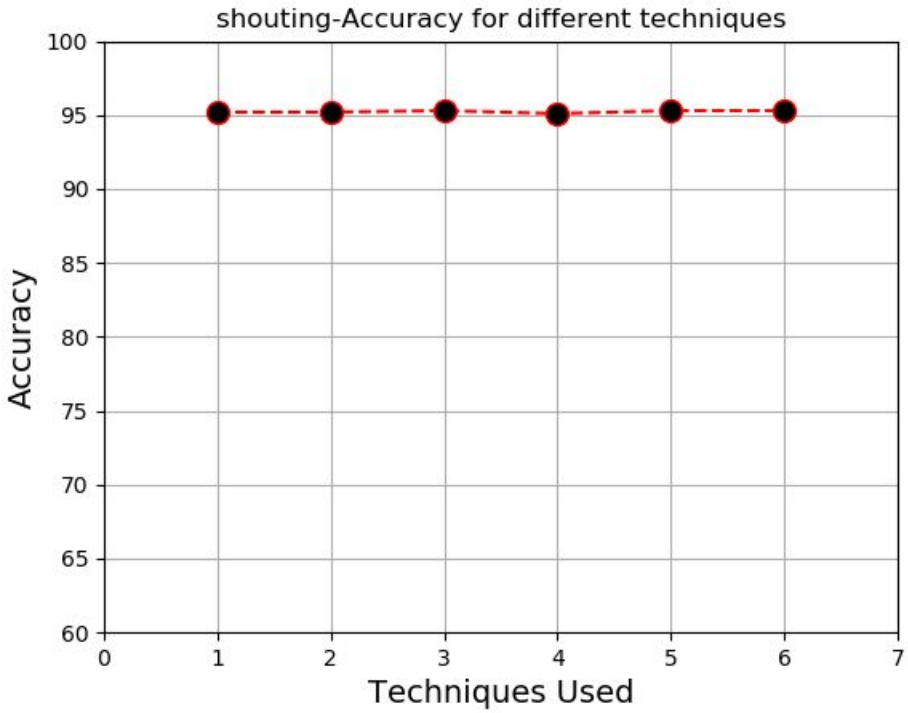
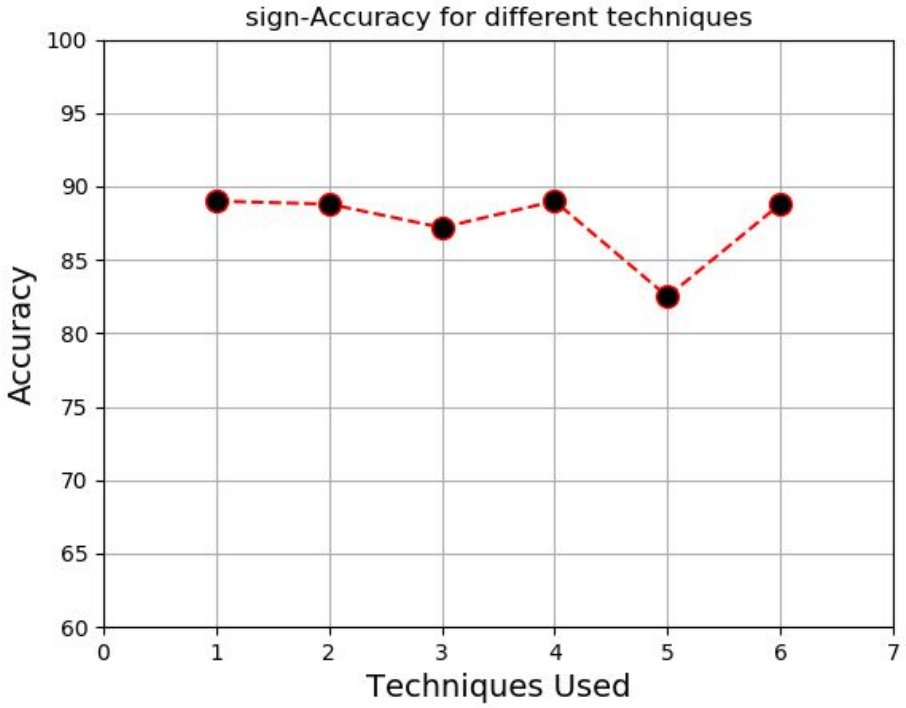
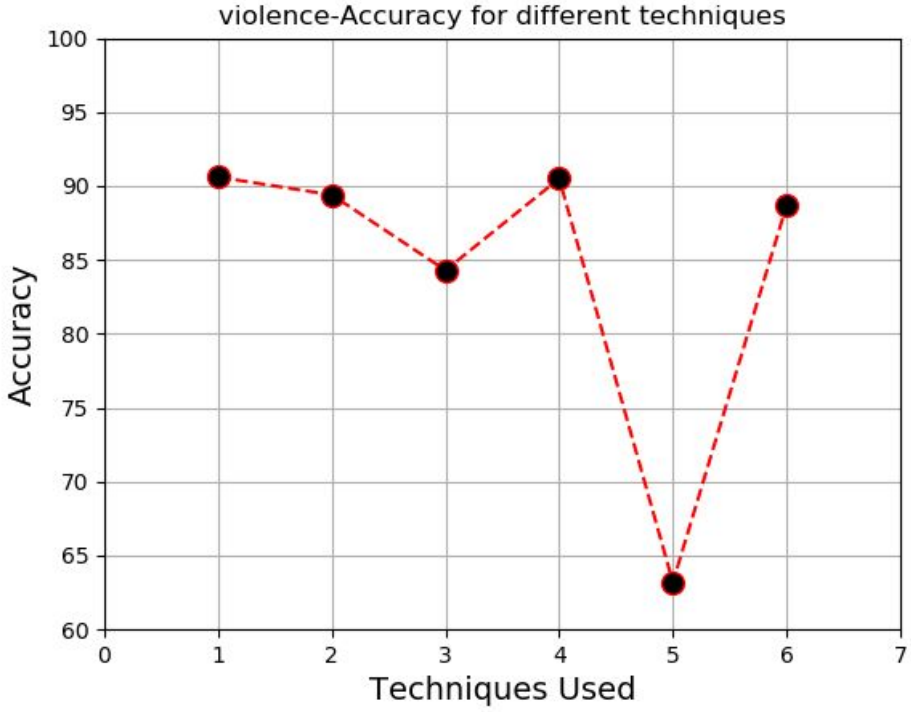


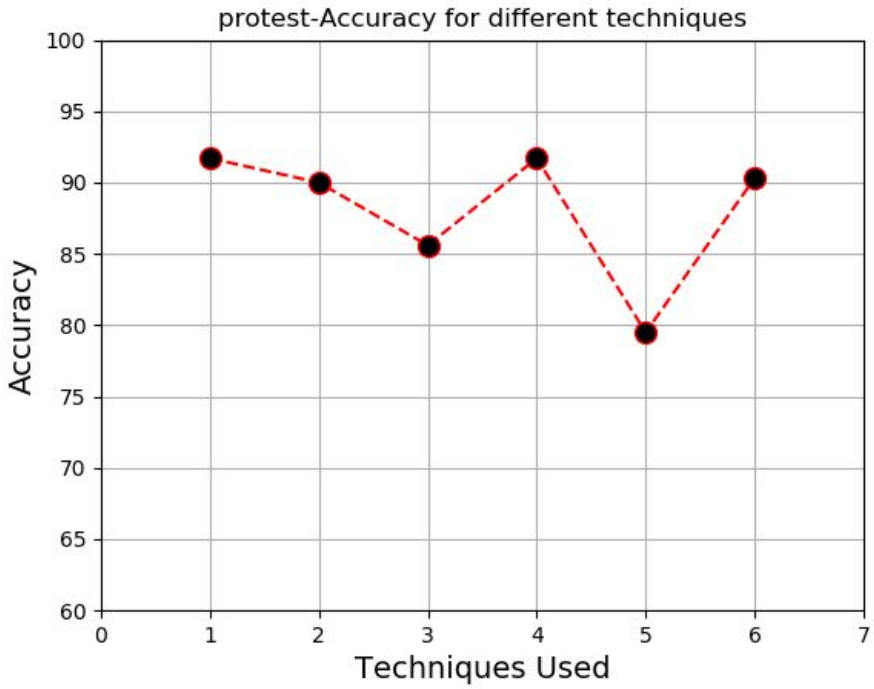
photo-Accuracy for different techniques
100
95
90
85
80
75
70
65
60
Accuracy
0
1
2
3
4
5
6
7
Techniques Used


police-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7

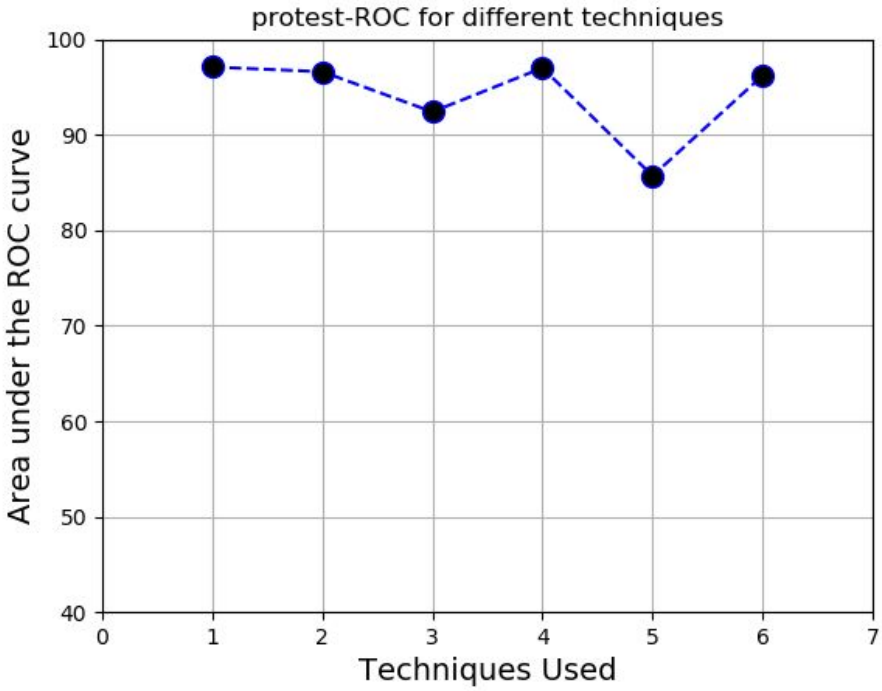

shouting-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7

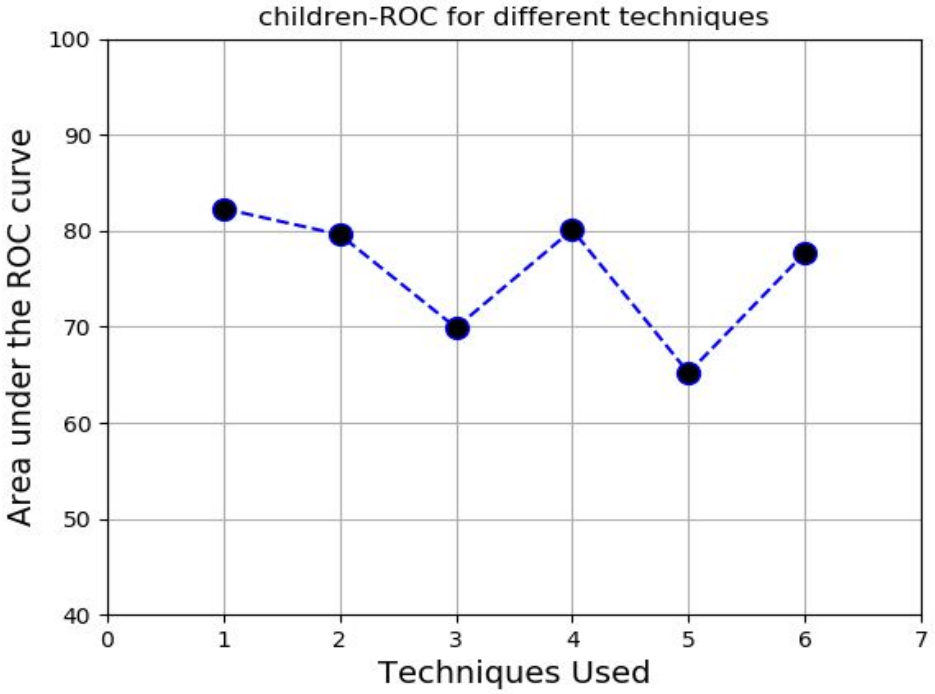

sign-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7

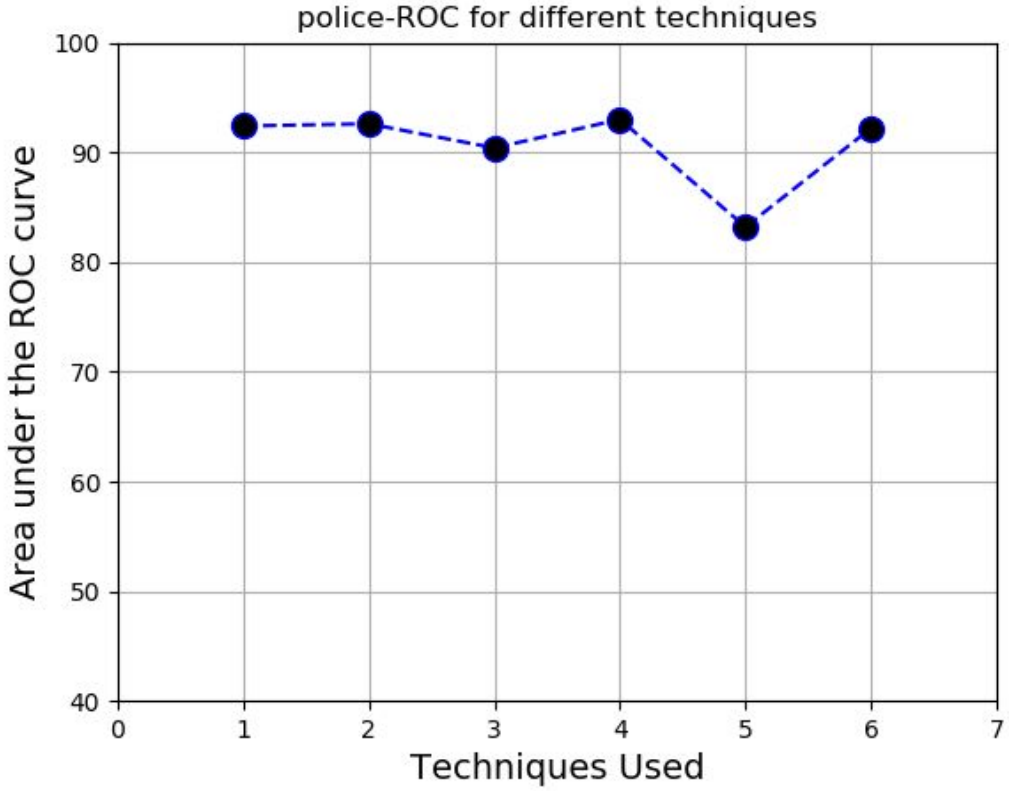

violence-Accuracy for different techniques
Accuracy
Techniques Used
100
95
90
85
80
75
70
65
60
0
1
2
3
4
5
6
7

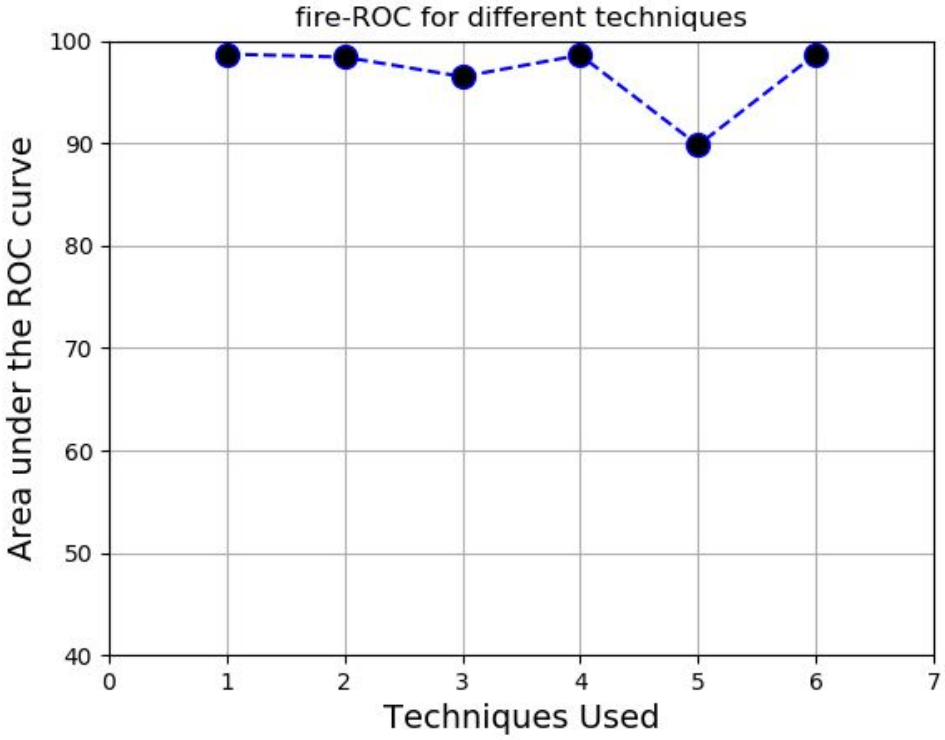


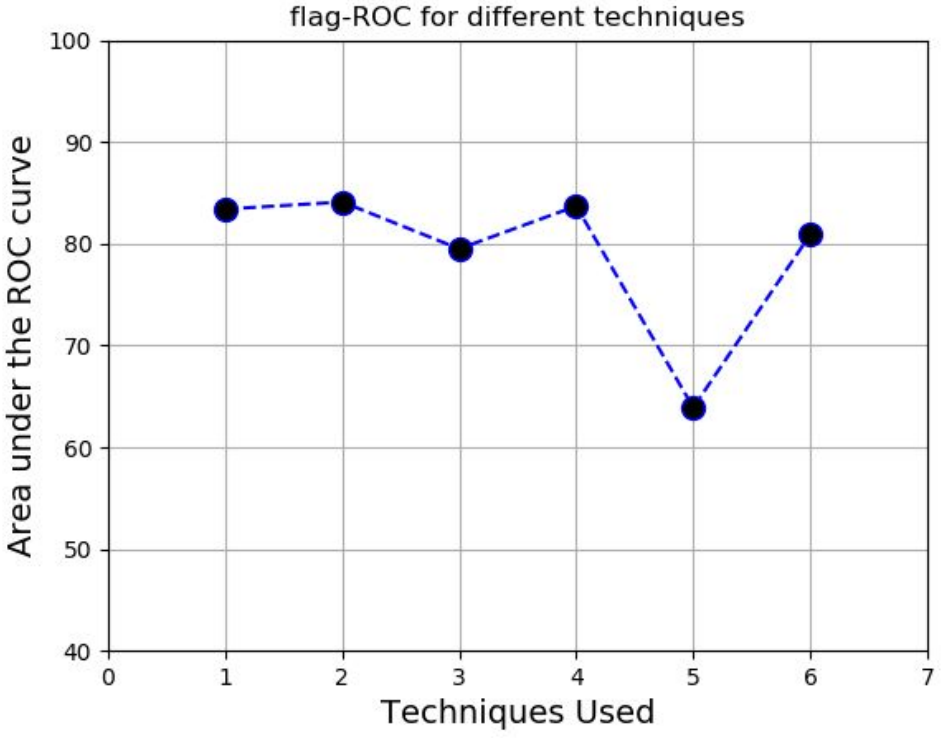
The area under ROC curve:

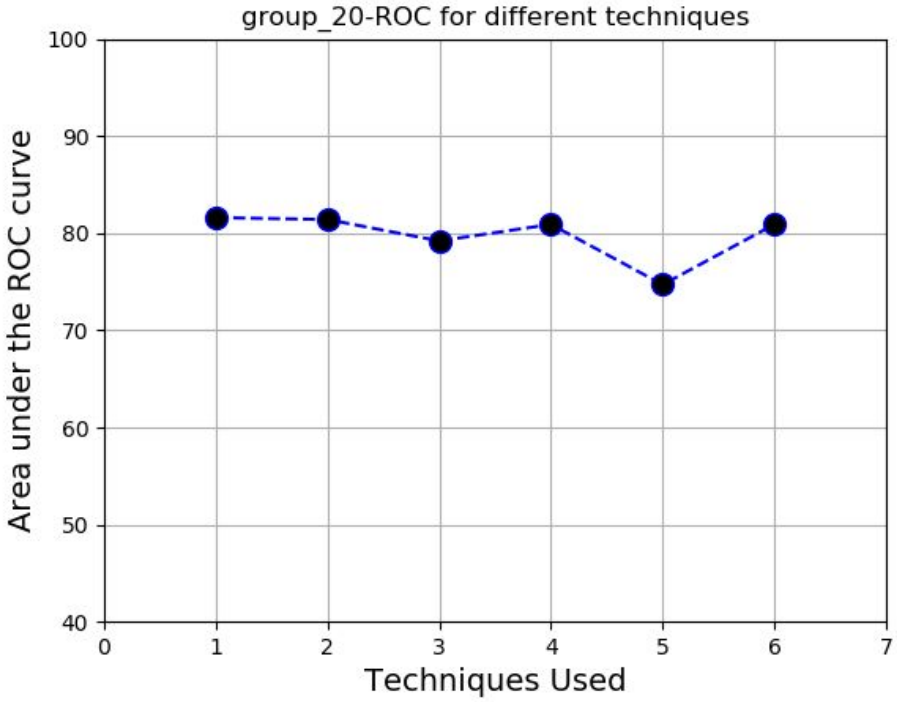


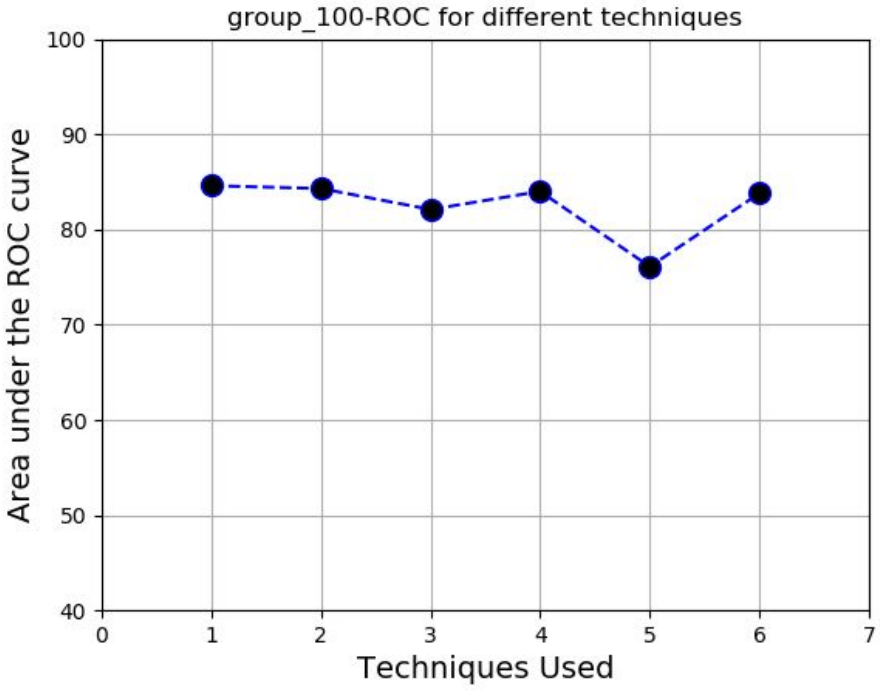

children-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7

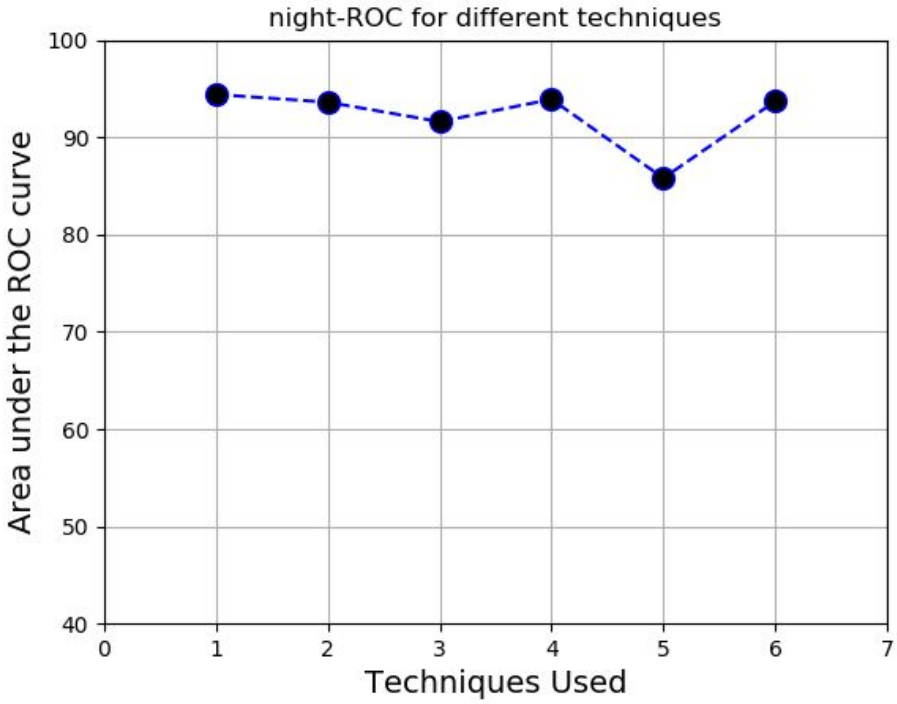

police-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7


fire-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7


flag-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7


group_20-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7


group_100-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7


night-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7

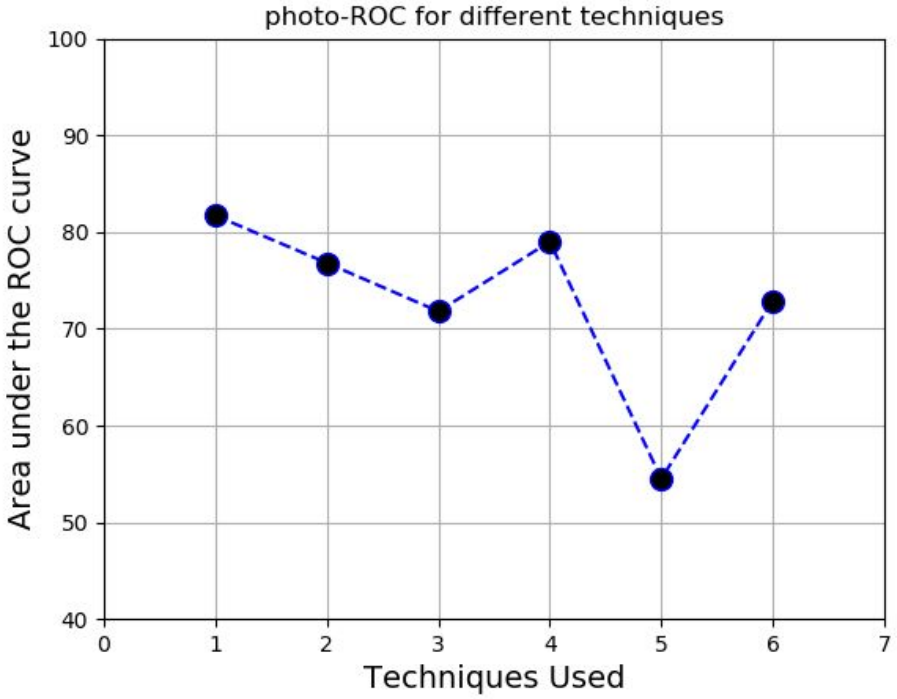
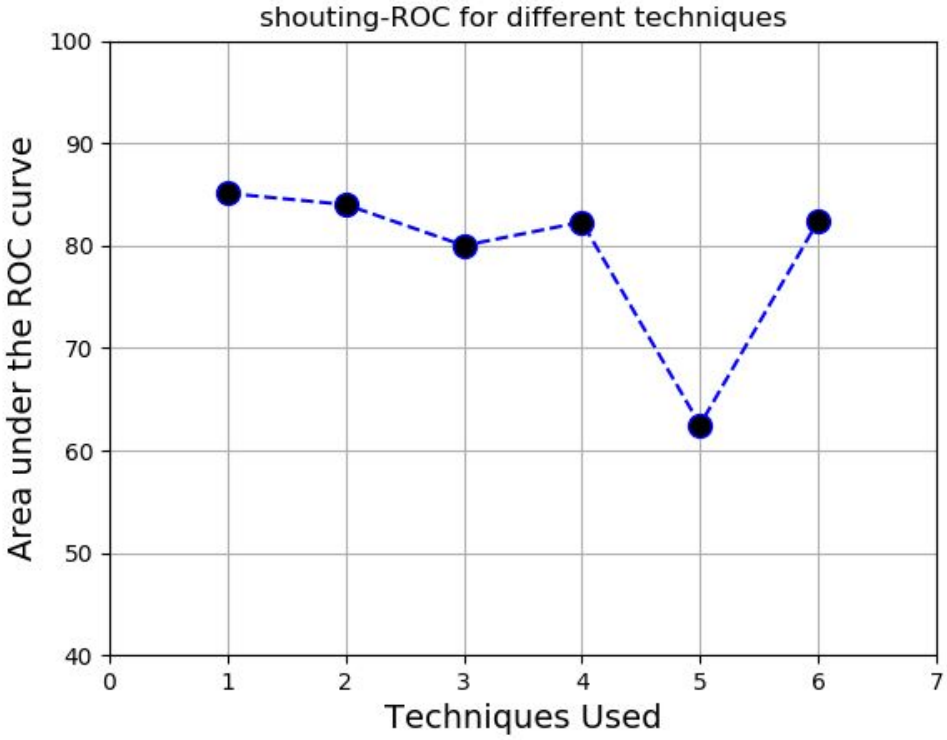
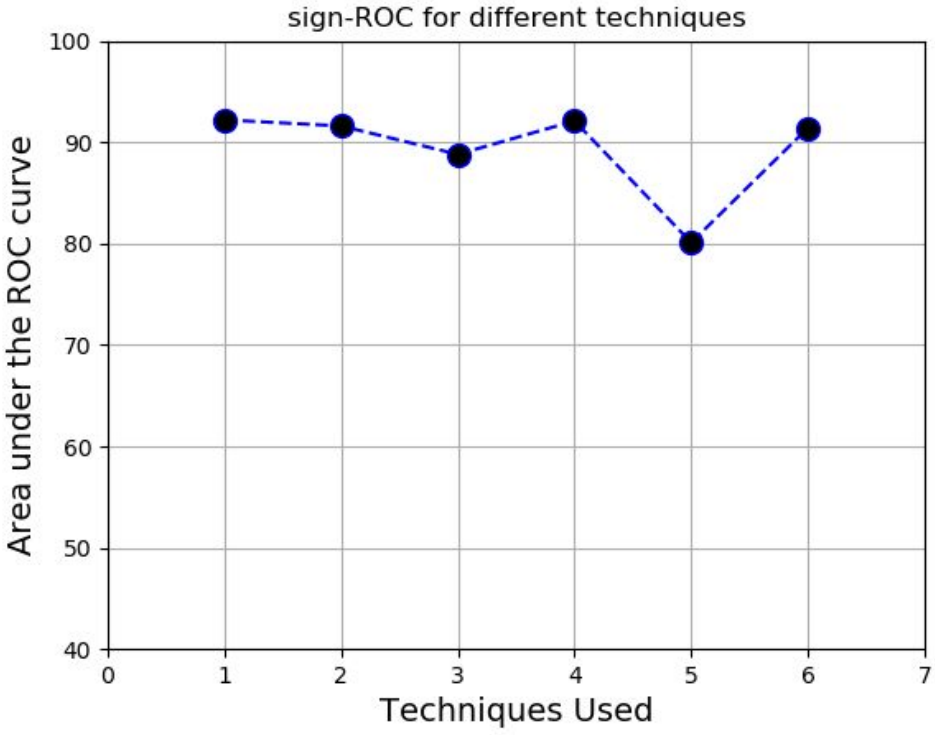


photo-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7


shouting-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7


sign-ROC for different techniques
Area under the ROC curve
Techniques Used
100
90
80
70
60
50
40
0
1
2
3
4
5
6
7

Results:

By changing the learning rate to 0.001 on the base model with resnet50, we were able to improve on the results of the baseline. Model 2 matched the performance of baseline and improved on it on 4 out of 12 categories

For detailed plots and individual code improvements please visit:

https://github.com/paritoshshirodkar/News-Image-Classification

References:

https://ieeexplore.ieee.org/document/6327984

https://medium.com/@tifa2up/image-classification-using-deep-neural-networks-a-beginner-friendly-approach-using-tensorflow-94b0a090ccd4

https://arxiv.org/abs/1709.06204

https://github.com/wondonghyeon/protest-detection-violence-estimation

https://arxiv.org/abs/1412.6980