PROJECT REPORT

Team Members:

1. Ebuka Okpala
2. Shivam Pandit
3. Christin Wilson

Introduction:

Deep Learning is a new technique that enables computer to do tasks that are normal to humans: learn from data and examples Deep Learning is key technology that helped us to think about driverless cars that can automate safe autonomous driving experience. Those driverless cars can determine pedestrian, stop-signs and even any obstructions to decide about the next movement of the vehicle. It is gaining popularity because of its power to control things that requires less human involvement in making decisions. Deep Learning models can learn patterns from the images. Deep Learning is a new technique that enables computer to do tasks that are normal to humans: learn from data and examples. Deep Learning is key technology that us to think about driverless cars that can automate safe autonomous driving experience. Those driverless cars can determine pedestrian, stop-signs and even any obstructions and decide about the next movement of the vehicle. It is gaining popularity because of its power to control things that requires less human involvement in making decisions. Deep Learning models can learn patterns from the images.

Objective:

The first part of the project was about classifying any input image to one of the 9 categories of cyberbullying images like gossiping, isolation, laughing, strangle, stabbing, punching, quarrel, slapping, pulling hair or else a not bully image. The model should predict any given image to one of these 10 categories. The second part of the project was about image detection where any image if found as bullying, it should label the images with predator and the victim. So, only images which are bullying will require to go through image detection to determine predator and the victim.

Network Structure:

To classify bullying and non-bullying in images we used a convolutional neural network and followed the VGG16 architecture. The model used a 3 x 3 filter through the network to increase its depth and has 13 convolutional layers and 5 pooling layers. The number of activation maps starts at 64 and increased by a factor of 2 after each max-pooling layer until it gets to 512.
Process:
We downloaded non-cyberbullying images to increase the image class to 10. The addition of the non-bully images decreased the accuracy but not by a large margin. Training was carried out using mini-batch gradient descent with momentum and learning rate set to 0.001 initially. we started with an epoch of 2 to get a sense of what the loss is and gradually increased the epoch

to 7, and to 74. While training with epoch set to 74 we also set weight decay to 0.0005 and we started with a learning rate of 0.01 and decreased it by a factor of 10 every 7 epoch.
We applied horizontal flips and normalization and subtracted mean from the images to increase the dataset size.

Accuracy:
We achieved highest accuracy of 68% on the image classification after multiple iteration on training. We observed that retraining the model on the weight from the initial training increased the accuracy until 68%. This was accomplished by keeping the learning rate at 0.0001, excluding weight decay and reducing the learning rate by a factor of 10 only a maximum of 3 times in N epoch.

Object detection:
We are using YOLO for image detection. When our model predicts an image as one of the classes of bully, that image is passed to the weights we trained on YOLO and the model draws a bounding box and labels the objects as predictor and victim.

Object detection dataset:
The Annotated dataset is created using LabelImg. We labelled all the images in the dataset with the boundary boxes and generated the txt files as well as the XML files for each of the images. The .txt file is the format supported by YOLO. We did not label the nonbullying images. The classes we used are victim (0) and predator (1).

Code Usages:
In terms of code usage, we rewrote the vgg16 code and borrowed some ideas from the following sources:
https://github.com/pytorch/examples/blob/42e5b996718797e45c46a25c55b031e6768f8440/ima
genet
/main.py https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py

- To run the code input the command: "python main.py -p 1 -i <image path>"

Example: python main.py -p 1 -i my_image.jpg

Reference Papers:
We followed the following paper: very deep convolutional networks for large-scale image recognition