# ADVANCE Labs - Harassment in Images Detection Lab

Copyright © 2021 - 2023.
The development of this document is partially funded by the National Science Foundation's Security and Trustworthy Cyberspace Education, (SaTC-EDU) program under Award No. 2114920. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation. A copy of the license can be found at http://www.gnu.org/licenses/fdl.html.

## 1 Lab Overview

In this lab, you will keep learning about how AI/ML can be used to detect societal issues such as cyberbullying. Cyberbullying is bullying performed via electronic means such as mobile/cell phones or the Internet. The objective of this lab is for students to gain practical insights into online harassment, such as cyberbullying, via images and to learn how to find cyberbullying from images using pre-trained AI/ML solutions to defend against this problem.

In this lab, students will be given a starter code to find cyberbullying and non-cyberbullying images from publicly available datasets. Their task is to follow the instructions provided in the Jupyter Notebook to use a pre-trained AI/ML model on the given dataset. In addition, students will learn the Approach to analyzing the cyberbullying in images using a dataset, there are four steps:
(1) Understand and identify the factors related to cyberbullying in images.
(2) Load the pre-trained model.
(3) Evaluate the pre-trained model with the cyberbullying image dataset
Moreover, this lab covers the following topics:

- Detection of cyberbullying in images
- AI-based classifier models to predict cyberbullying vs. non-cyberbullying in images

## 2 Lab Environment

This ADVANCE lab has been designed as a Jupyter notebook. ADVANCE labs have been tested on the Google Colab platform. We suggest you use Google Colab, since it has nearly all software packages pre-installed, is free to use, and provides free GPUs. You can also download the Jupyter Notebook from the lab website and run it on your own machine, in which case you will need to install the software packages yourself (you can find the list of packages on the ADVANCE website). However, most of the ADVANCE labs can be conducted on the cloud, and you can follow our instructions to create the lab environment on the cloud.

## 3 Lab Tasks

### 3.1 Getting Familiar with Jupyter Notebook

The main objective of this lab is to learn how a pre-trained AI/ML model can be used to detect online harassment through images, such as cyberbullying. Before proceeding to that, let us get familiar with the Jupyter notebook environment.

Jupyter notebooks have a Text area and a Code area. The Text area is where you'll find instructions and notes about the lab tasks. The Code area is where you'll write and run code. Packages are installed using `pip` and need to be preceded with a `!` symbol. Try accessing the lab environment for this task here.

The lab has three areas: one text area and two code areas. Follow the instructions for the three areas, fill them with the instructed content, and add a screenshot to your report.

## 3.2 Cyberbullying Detection in images

In this lab, you will use a pre-trained AI/ML model to detect cyberbullying in images. You will use a dataset of real-world images to train your AI model, evaluate the performance of a pre-trained AI classifier model, and check the result with one random instance from the dataset. You can access the lab by clicking here.

### 3.2.1 Download the pre-trained model, test dataset and the dependencies

In this lab, we provide three publicly available dataset: auxes dataset, poses dataset, and images.You can download the model and dataset as per the lab instruction.

In this lab, you will be using the auxes, poses, and image data set, which consist of real-world cyberbullying images. Run all the code of the lab. Report your results according to each designed task and discuss your idea at the end.

### 3.2.2 How to identify cyberbullying in images

Follow the instructions in the text areas and run the subsequent codes to load example image from a code snippet, as follows. Here is a sample from the lab:

```
testImage = img.imread('/content/cyberbullying_data/cyberbullying_data_splits_clean/
    test/cyberbullying/7.s-s-unshaven-sad-ashamed-man-doing-loser-sign-hand-fingers-his
    -front-funny-depressed-face-expression-s-139158713.jpg')
```

There are 5 factors to measure cyberbullying in images
1. Body-pose
2. Facial Emotion
3. Object
4. Gesture
5. Social Factors

Refer to the table from the lab environment, which shows an analysis of cyberbullying factors in images.

### 3.2.3 Load Dataset

Follow the instructions in the text areas and run the subsequent codes to load your data from a predefined class, as follows. Here is a sample from the lab:

```
class PosesDataset(Dataset)
```

This predefined class checks cyberbullying and non-cyberbullying images from the dataset and includes the generated tokens in your report. You can add a code block to run your code.

Generate a test set from the poses and auxes dataset by following the lab instructions. Here is a sample from the lab:

```
test_set = PosesDataset('cyberbullying_data/cyberbullying_data_splits_clean/test/', '
    cyberbullying_data/cyberbullying_poses/test/', 'cyberbullying_data/
    cyberbullying_data_auxes/test/')
```

### 3.2.4 Define a basic AI model for cyberbullying image detection

After you have loaded the dataset, the next task is to build an AI classifier model. Follow the lab instructions to create the AI model.

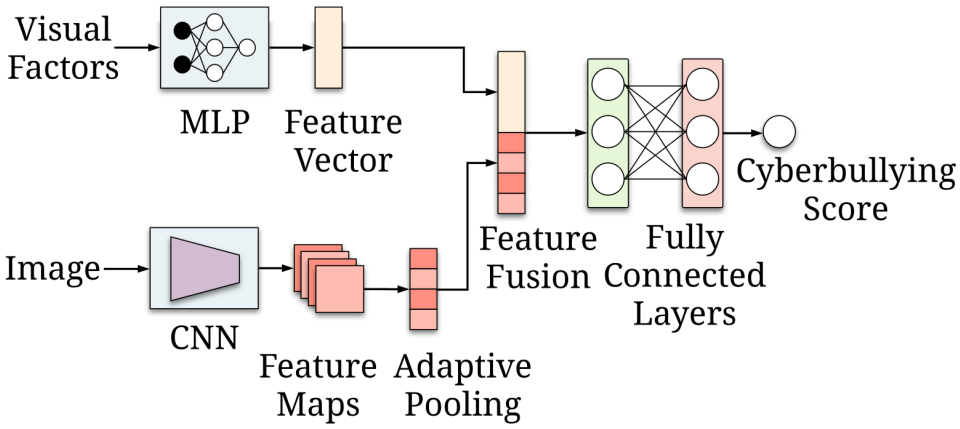



Figure 1: Cyberbullying image detection model.

### 3.2.5 Task 1: Generate the detection results for the basic model with the test dataset

In Task 1, you are supposed to write code to calculate the accuracy of the basic cyberbullying image detection model. Run your model on the test data and report your results here. Use lab instruction:

```
correct, incorrect, total = 0., 0., 0.
...

# TODO: Write code to print out the accuracy of the base model
# Your code here:
```

### 3.2.6 Load the pre-trained model to detect cyberbullying images.

Now, let's load the pre-trained model for cyberbullying image detection. The pre-trained checkpoint has been downloaded, we can load it with the provided code:

```
model = torch.load("auxes_17.pt")
model = model.to(device)
```

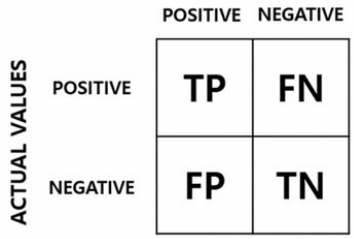
### 3.2.7 Task 2: Write code to generate result report containing: Accuracy, Precision, Recall, and F1-Score

After generating the detection results from the test dataset. Now, let's learn some different evaluation metrics. You are supposed to read the reference first and complete the code to generate a result report containing Accuracy, Precision, Recall, and F1-Score. Please refer to the given description.

```
# TODO: Complete the following code to calculate the accuracy, precision, recall and
    F1 score.
acc =
precision =
```

```
recall =
f1 =

print('The accuracy for test dataset is: {}%'.format(acc * 100))
print('The precision for test dataset is: {}%'.format(precision * 100))
print('The recall for test dataset is: {}%'.format(recall * 100))
print('The f1 score for test dataset is: {}%'.format(f1 * 100))
```




$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

### 3.2.8 Task 3: Write code to compute the confusion matrix

Please look into the given link to understand what is the confusion matrix. https://en.wikipedia.org/wiki/Confusion_matrix Write code to plot the confusion matrix (you are allowed to borrow any Python tools, such as scikit-learn (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html))

### 3.2.9 Task 4: Confusion matrix plotting

After computing the confusion matrix, we can use some Python tools to plot the confusion matrix. Write your own code snippet to plot the confusion matrix we computed in Task 3. (You are allowed to use any Python tool, such as `matplotlib`, `seaboon` to visualize the confusion matrix.)

### 3.2.10 Task 4: Receiver Operating Characteristic (ROC) Curve

An ROC (Receiver Operating Characteristic) curve is a graphical representation used in machine learning and statistics to evaluate the performance of a binary classification model. It illustrates the diagnostic ability of a classifier by plotting two metrics:

- True Positive Rate (TPR) or Sensitivity/Recall: The proportion of actual positives that are correctly identified by the model. It is calculated as:

$$\text{TPR} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

- False Positive Rate (FPR): The proportion of actual negatives that are incorrectly identified as positives by the model. It is calculated as:

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

The ROC curve plots TPR (y-axis) against FPR (x-axis) at various threshold settings. Each point on the ROC curve represents a different threshold value for classifying a positive instance.

Area Under the Curve (AUC): The area under the ROC curve is a single scalar value summarizing the overall performance of the classifier. An AUC of 1 indicates a perfect classifier, while an AUC of 0.5 suggests a model with no discriminative ability, equivalent to random guessing. In this task, you need to write your code snippet to compute and plot the ROC curve and report the AUC value.

### 3.3 Check with one random instance from the test dataset

After generating results from the test dataset, you now need to randomly select an instance in the test dataset using,

```
picture_index  = "11" #@param [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19]
index = int(picture_index)
instance = test_set[index]
```

Follow the lab instructions to plot the images and check the predictions of the model.

## 4 Submission instruction

You need to submit a detailed lab report, with screenshots, to describe what you have done and what you have observed. You also need to provide an explanation for the observations that are interesting or surprising. Please also list important code snippets followed by an explanation. Simply attaching code without any explanation will not receive credits.