

Project Name: Explainable Detection of Online Sexism (EDOS)

**Course Title: Natural Language Processing II**

**Course Code: CSE440**

**Project Team No: 02**

**Group Members:  
Mantaqa Abedin (23241130) (Section: 01)**

**Monowar Zamil Abir (21201187) (Section: 02)**

**Introductions:**

The project aims to develop a machine learning project for two natural language processing tasks. The first task is to detect sexism given a text and the second is to categorize sexism in four different predefined units. The project includes data preprocessing, training, test and result analysis. Besides, pre-trained word embedding is also used in the project.

## **Dataset:**

The dataset contains 14000 rows and 5 columns. The columns ‘text’ represents the features of the machine learning model and both the ‘label\_sexist’ and ‘label\_category’ represent two label columns. ‘label\_sexist’ is the label column for task A and ‘label\_category’ is the label column for task B. For task A there are a total 10602 ‘not sexist’ labels and 3398 ‘sexist’ labels. Out of these 3398 sexist labels 1590 are in ‘derogation’, 1165 are in ‘animosity’, 333 are in ‘prejudiced discussions' and 310 are in ‘threats, plans to harm and incitement’ category.

Opening IDE from the submission folder requires dataset path to be :

trainSet = pd.read\_csv('train\_all\_tasks.csv')

If we upload the .ipynb file on Jupyter Notebook (Anaconda Navigator): The dataset is stored in the same folder as the .ipynb project file and loaded using pandas using this relative path address: “Desktop\\CSE440 Project\\train\_all\_tasks.csv”. We had to add “Desktop\\CSE440 Project\\” before the dataset name since the Jupyter Notebook takes ‘C:\\Users\\User profile name’ as the current working directory and we placed the folder in our homepage.

trainSet = pd.read\_csv('Desktop\\CSE440 Project\\train\_all\_tasks.csv')

## **Methodology:**

1. **Data Preprocessing:**

* **Train-test split:** We have splitted our dataset into training, testing and validation where 80% of it is for training, 10% for testing and 10% for validation.
* **Label Encoding:** We have manually encoded ‘not sexist’ labels into 0 and ‘sexist’ labels into 1. For the labels of task B, we have used ‘LabelEncoder’ from scikit-learn library.
* **Text Tokenization and Padding:** We have used the Keras Tokenizer for tokenizing the training features. We have fitted the training features and transformed the testing and validation features. Padding was used to keep the sequences of uniform length which is 100.

1. **GloVe Embeddings:** To create the embedding matrices, we have used the pre-trained GloVe embeddings. We have used the **glove.6B.300d** for our model. The path for this embedding file has to be set on the ‘glove\_path’ variable which is below the ‘load\_glove\_embeddings’ function.

## 

## **Training:**

* **Task A:**

For task A, we have used a sequential model with dense layers, LSTM and Bidirectional LSTM. All of them have ‘ADAM’ as model optimizer, ‘binary\_crossentropy’ loss function and apply dropout regularization of 0.2 value. The dense sequential model has a layer of 24 units with ReLU activation function and both the LSTM and Bidirectional LSTM uses 128 units. Sigmoid activation function is used for the output layers of the models.

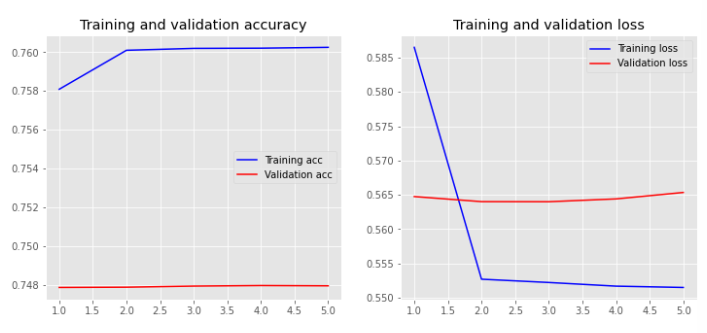
* **Task B:**

For task B, we have used LSTM and Bidirectional LSTM models. Both of them have ‘ADAM’ as the optimizer and ‘sparse\_categorical\_crossentropy’ for loss function. Dropout and units are the same as task A but the output layer has softmax activation.

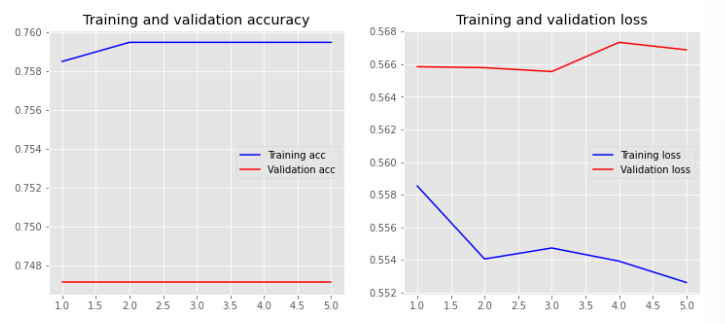
For both tasks, early stopping is used and patience is set at 2.

## **Result and Analysis:**

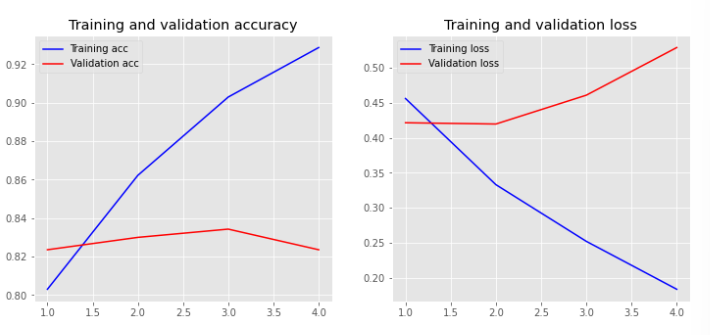
* **Task A:** For the dense model we got 76.03% training accuracy and 75.07% testing accuracy.



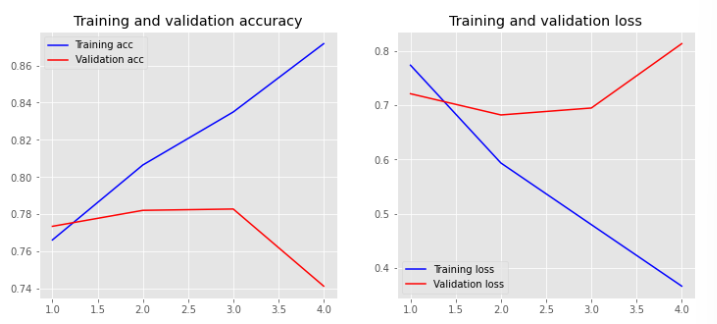
For the LSTM model, the training accuracy decreased to 75.95% and the testing accuracy was 75%.



Last, we got a very good 90.28% training accuracy and 82.93% testing accuracy from the bidirectional LSTM model. We calculated the precision, recall and F1-score for the model where we found the precision is 76.81%, recall is 45.42% and F1\_score is 57.09%.



* **Task B:** For the LSTM model, we found a similar accuracy score as task A. On the other hand, for the Bidirectional LSTM model we got training accuracy of 83.38% and testing accuracy of 78.50%.



We can see that all of the models for both tasks resulted in satisfactory accuracy rate and out of them Bidirectional LSTM model gives best accuracy. Yet, we found a noticeable performance difference between the two classes (sexist and not sexist) when we calculated the precision, recall and F1-score for the Bidirectional LSTM model of task A.

**Class 0 (not sexist):**

Precision: 0.84 (84%)

Recall: 0.95 (95%)

F1-score: 0.89 (89%)

**Class 1 (sexist):**

Precision: 0.77 (77%)

Recall: 0.45 (45%)

F1-score: 0.57 (57%)

The reason behind the lower scores for the ‘sexist’ class can be imbalance of the dataset. Further optimization or adjustments may be needed to address class imbalance and improve the model's ability to detect instances of the minority class ‘sexist’.

(Values changes slightly with each execution of the .ipynb file)

## **Conclusion:**

To conclude, the machine learning project uses a dataset to learn different texts containing sexist or non-sexist content. Furthermore, it categorizes different sorts of sexism from the texts. Different pre-processing of data and models are implemented and a good accuracy is also acquired overall. Additionally, the classification report shows a performance imbalance between two different classes. Lastly, the model provides valuable insights about the text classification task and can be helpful in various domains related to natural language processing.