Project 2 - JIgsaw Unintended Bias in Toxicity Classification

Upskill - Intelligent Machines Cohort Program (Machine Learning Engineer)

Participant Name: **Maksudur Rahman**

Abstract: The Conversation AI team, a research initiative founded by Jigsaw and Google (both part of Alphabet), builds technology to protect voices in conversation. A main area of focus is machine learning models that can identify toxicity in online conversations, where toxicity is defined as anything rude, disrespectful or otherwise likely to make someone leave a discussion. Here’s the background: When the Conversation AI team first built toxicity models, they found that the models incorrectly learned to associate the names of frequently attacked identities with toxicity. Models predicted a high likelihood of toxicity for comments containing those identities (e.g. "gay"), even when those comments were not actually toxic (such as "I am a gay woman").

In this competition, the challenge is to build a model that recognizes toxicity and minimizes this type of unintended bias with respect to mentions of identities. The provided dataset is labeled for identity mentions and another target is to optimize a metric designed to measure unintended bias. As a participant of the competition, I need to develop strategies to reduce unintended bias in machine learning models, and so I can contribute to the entire industry, to build models that work well for a wide range of conversations.

Methodology:

1. **Phase 1:** In the first phase, I imported all the necessary packages and libraries. Then I stored the train data in a dataframe and explored and analyzed the data in order to make sense of the features and the targets.
2. **Phase 2:** In this step, I uploaded the pre-trained word vectors of GloVe as a text file and then built vocabulary out of our train dataset using them. Necessary preprocessing of the texts such as removal of punctuations and special characters were done here. As the values of the target column are float points, I converted them to booleans based on the identity column of the dataset where the values equal to and greater than 0.5 were true while

the others were false. Then I split the train data into 80% train set and 20% validation set. This validation set is used after the training of the model. Also the function that adds padding to the tokenized texts is defined here.

3. **Phase 3:** Here I imported a dataset from a previous competition in order to pre-train the model. All the necessary preprocessing is done in this step.
4. **Phase 4:** In the final phase, I defined all the hyper-parameters for training and split the current train set into another train-test set of 80% : 20% ratio. I created the embedding matrix for the LSTM layer of the model and defined the model. After that I trained the model. In addition, I evaluated the model's performance based on the metrics provided in the competition using the previously separated validation test. Finally, I created the submission file with the model's prediction on the test set.

Result Analysis:

My LSTM model scored around 0.50 on the private scoreboard. After modifying the hyperparameters and the structure of the model, the score significantly improved to the best value which is 0.87911. The model also achieved a decent overall auc score of around 0.88 on the unseen validation set.

Conclusion:

Some improvements can be achieved on the performance of the model. For example:-

1. **Pre-training:** Similar datasets from previous competitions can be used to pre-train the model before the final training on the actual dataset. It may improve the model's score.
2. **Vocabulary Building:** Using word-vector embeddings such as GloVe, Crawl etc. to build vocabulary on the dataset can be a significant step.
3. **Pre-trained Architectures:** State-of-the-art models such as Google Bert and GPT have outperformed other architectures in similar competitions which is an important point to be noticed.