Project 2

Jigsaw Unintended Bias in Toxicity Classification

Abstract:

This problem was posted by the Conversation AI team (Research Institution) in Kaggle competition.

This problem's main focus is to identify the toxicity in an online conversation where toxicity is defined as anything *rude, disrespectful, or otherwise likely to make someone leave a discussion*.

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", "muslim", "black").

The model predicted the comment, "I have a muslim friend", "I'm gay" as toxic.

Unintended Bias

The models are highly trained with some keywords which are frequently appearing in toxic comments such that if any of the keywords are used in a comment's context which is actually not a toxic comment but because of the model's bias towards the keywords it will predict it as a toxic comment.

For example: "I am a gay woman"

Check out the "Identifying-bias-and-possible reason" notebook.

Methodology:

1. Data Cleaning:
    - Training dataset contains “comment text” and “Target” column.
    - At first using a “decontracted” function, comments from the The training and testing datasets are processed so that the words like won’t, aren’t are converted into their base form like will not, are not.
    - Then special characters, numbers are removed from the comments.
2. Exploratory Data Analysis:
    - Word clouds are plotted for toxic and non-toxic comments
    - Vilion and Density plots are plotted for some of the extracted features with target labels.
3. Feature Selection:
    - The correlation of extracted feature with target and some of the other features form the dataset is measured.
    - Features with high correlation has been fed to the model.
4. Modeling:
    - CNN and LSTM model gave the state of the art accuracy without much preprocessing.

Result Analysis:

- CNN model gave the best score which is 0.91381 but took time, around 18mins.
- Single layer LSTM took much time, around 23 mins, but the score was also a bit less. Score on the leaderboard is 0.89538.
- SGD classifier gave 0.84719 score with 13 mins.

Conclusion:

The 0.91381 score is satisfactory as it was achieved without much preprocessing and ensemble of different models. Definitely, the score can be improved by preprocessing the data a bit more, increasing epoch, hyper parameter tuning and model ensemble methods.