NLP final project

Team: Tigran H., Tego C., Dauren B.

Step 1

Our team chose document classification for the final project. The data comes from ***kaggle competition***. The goal of the project is to classify toxic comments which were collected from Wikipedia and were labeled by a human. The target is a vector which contains six categories. This is a multiclass multilabel classification problem.
Data example:

	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0

Figure 1: Data example

Step 2

The generative model that we selected is a Multinomial Naive Bayes model. In order to train the Naive Bayes model, each combination of the labels was assigned to a different class. We have 6 labels, the possible number of combinations is 2^6=64, however, we only have 41 combinations in our data set, which means we have 41 classes. For example, if the comment was labeled as toxic, it will be assigned to class 1. If the comment was labeled as toxic and severe toxic, it will be assigned to class 2. Comments were vectorized using tf-idf technique with 20 000 most frequent words (total vocabulary: 177 603).

To understand the Multinomial Naive Bayes model's predictive quality we compare the model's accuracy, precision and recall with the baseline which was calculated using a shuffled target vector.

	Accuracy	Precision	Recall
Naive Bayes	0.879	0.847	0.879
Baseline	0.805	0.805	0.805

The metrics above do not provide a full picture as the data set is highly imbalanced. Therefore, we used ***confusion matrix*** (normalized by columns) to understand true/false positives/negatives through all classes. The confusion matrix clearly shows that Naive Bayes model can only predict neutral comments

- "class 0" and comments that belong to "class 22" which are `toxic`, `obscene`, `insult`. However, there are a lot of false positive predictions.
For the above-mentioned reasons, it is obvious that the model is not able to generate comments of a corresponding class. So, in order to complete step 4, we simplified the task and changed the type of the problem from multilabel multiclass to binary classification where we classified comments as `neutral` or `abusive`. The data set is still highly imbalanced (~10% abusive comments), so we performed an undersampling of the neutral comments and selected all the abusive ones. We used this model to generate synthetic data in step 4.

The entire process can be seen in ***jupyter notebook***.

Step 3

In this step, we tried five neural networks: four fully connected neural networks with different vectorization techniques and a bidirectional LSTM model. In this step, we the use original data set and train our neural networks as multilabel multiclass classification models for 41 classes.
In the first model, we vectorize our text using tf-idf with full vocabulary and then reduce the dimension of the data to 300 columns using truncated singular value decomposition. The second model employs multi-hot encoding with 10 000 most frequent words. The third model takes bigram multi-hot encoding as input. The fourth model uses a bag of bigrams count instead of multi-hot encoding. The last model uses sequences as input with a fixed length of 900 (90th percentile) in addition this model has an embedding layer.
Neural network models summary:

	Precision	Recall
FCNN (TF-IDF: Dimension Reduction)	0.789	0.585
FCNN (Bag of Words: Unigram Multi-Hot Encoded)	0.709	0.663
FCNN (Bag of Words: Bigram Multi-Hot Encoded)	0.686	0.639
FCNN (Bag of Words: Word Count)	0.729	0.613
Bidirectional LSTM	0.764	0.653

Bidirectional LSTM model's architecture

Step 4

In order to generate synthetic data we use the Naive Bayes model from step two and train it using a simplified data set with two labels (as it was shown, NB cannot handle the original data set with 41 unique classes). Then, we trained our Naive Bayes model using synthetic data. We decided to use the bidirectional lstm model(discriminative) for this part as this model has the best F1-score compared to other models. The results of the two models are:

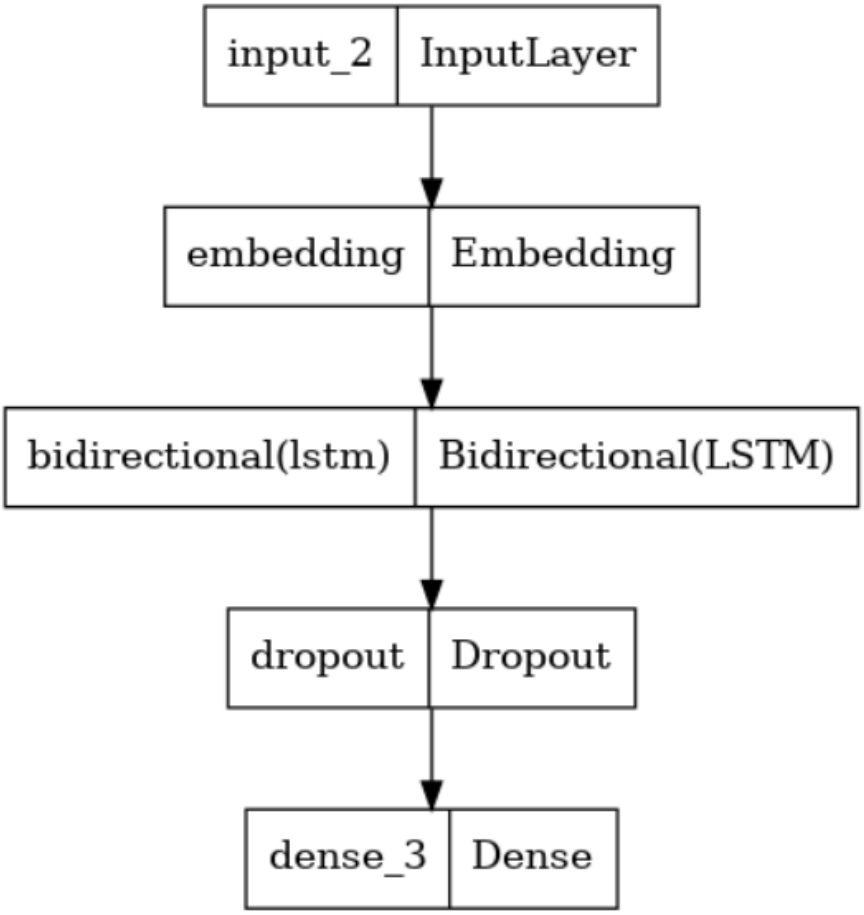

Figure 2: LSTM model

	Accuracy
Naive Bayes	0.999
Biderictional LSTM	0.998

Both models perform very well in classifying abusive comments.

Step 5

We trained neural networks and a Naive Bayes model on the original dataset (train dataset) 143613 comments. The Naive Bayes model was able to classify only a small portion of class 22 (`toxic`, `obscene`, `insult`) and class 12 (`toxic`) ***see NB confusion matrix***, while neural network model does much better on the original dataset. ***LSTM confusion matrix*** shows that the neural network model has fewer false positives in "class 0" and can predict other classes as well. (Precision and recall are different here for LSTM than in the table from step 3, as calculation and averaging strategies are different. To make comparison with NB model we made a comparison with the same weighting scheme):

	Precision	Recall
Naive Bayes	0.847	0.879
Biderictional LSTM	0.937	0.953

Step 6

The dataset is highly imbalanced. The proportion of each label column is:

	0	1
toxic	0.904	0.096
severe toxic	0.990	0.010
obscene	0.947	0.053
threat	0.997	0.003
insult	0.950	0.050
identity hate	0.991	0.009

This is a classification task across multiple classes and labels, however, since the Multinomial Naive Bayes model cannot handle it, we converted it into multiclass task by assigning each unique combination of labels to one class. There are 41 classes in total, including "normal" comments. Although we can extract the distribution of words by class (we extracted most probable words from different classes in Jupiter notebook) from the Naive Bayes model, the model was unable to distinguish between classes, except in some cases for class 22. To complete Step 4, which requires generating synthetic data, we simplified the task to two

classes (Normal and Abusive). In Step 4, both models show good results in solving this simplified classification problem. The only advantage that can be highlighted is that the weights of Naive Bayes model are interpretable.

For the discriminative model, we tried several methods, and the bidirectional LSTM showed the best F1 result, however, this required significant computational resources. In addition, we looked at surrogate interpretation models such as LIME to locally interpret the model's decision. We looked at the embedding matrix using t-SNE and UMAP projections and even though the number of epochs was low (5), we can see that offensive words are grouped in closer proximity to each other than neutral ones.
Although the bidirectional LSTM performs better compared to the Naive Bayes model, it still does not perform well enough to make classifications. This can be explained by a large number of classes and the insufficient number of comments for each class.