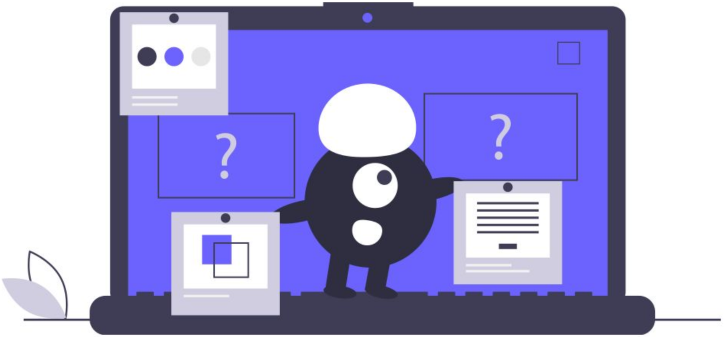
Toxic Comment Classification

Implementation of the Long-Short Term Memory (LSTM) Architecture



Problem Overview

- Detecting harmful speech on Wikipedia Talk Pages.
- An example of a sequence to vector problem.
- Multilabel problem with - 6 possible binary categories.

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0

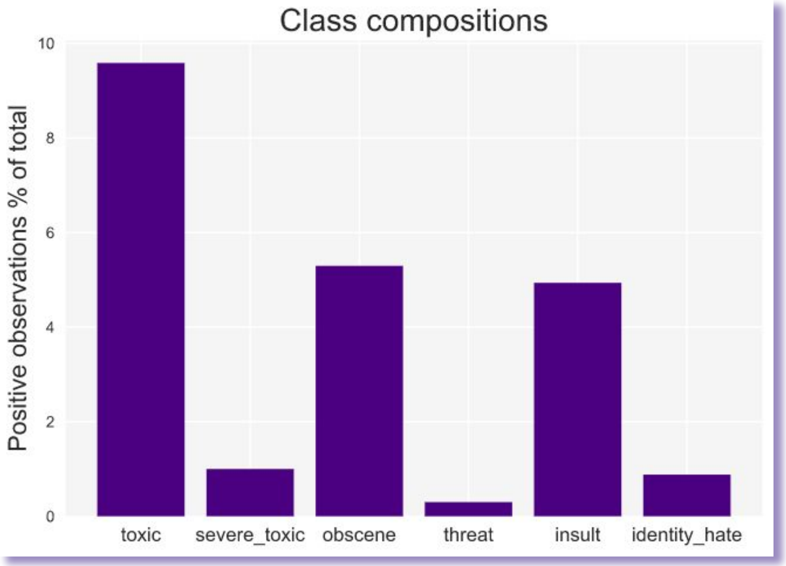
Two Possible Approaches

I. Splitting the problem into **6 binary classification problems** and training a classifier for each class
Used on **naive-bayes & logistic regression**

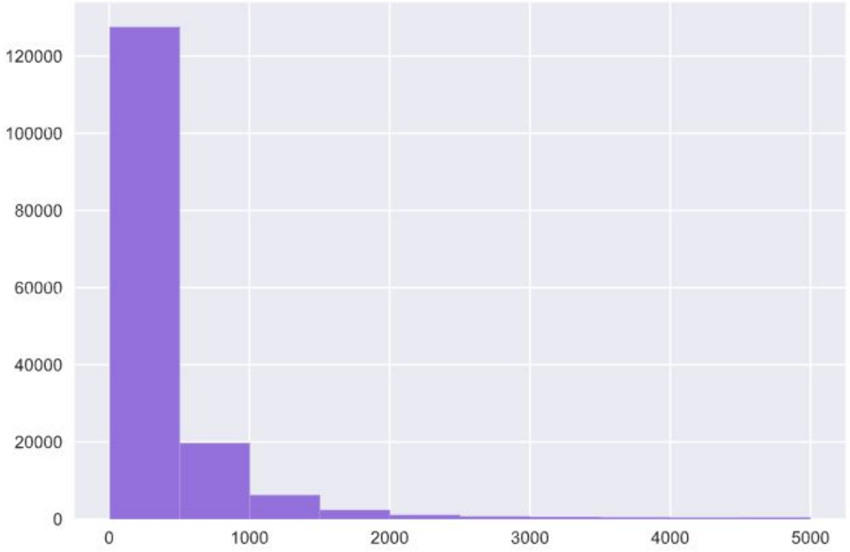
II. Using a multi-label classification algorithm that is trained on all classes together.
Used on a **recurrent neural network**

Exploratory Data Analysis

1. Identified the **imbalance** in classes
2. Identified the **high variance and skewness in comment lengths**



Distribution of comment lengths

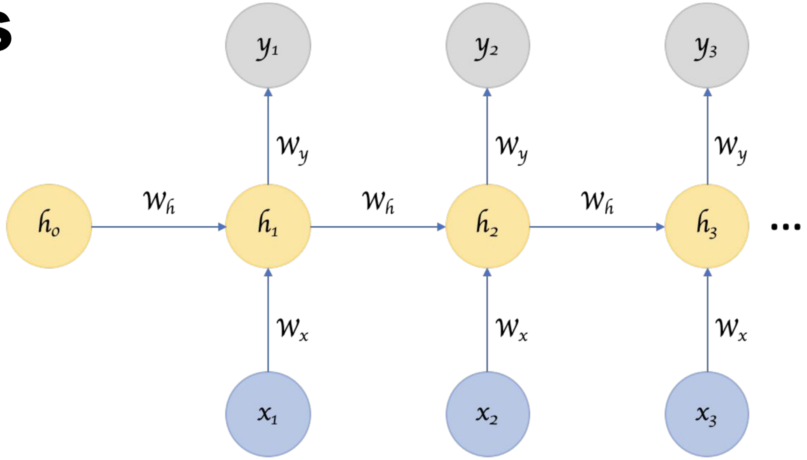


Evaluation Metrics

- ‘Positive’ observations make up a very small proportion in each category.
- Accuracy is an unreliable metric.
- Minimizing false positives is essential.
- Recall, and AUC-ROC are used in this project.

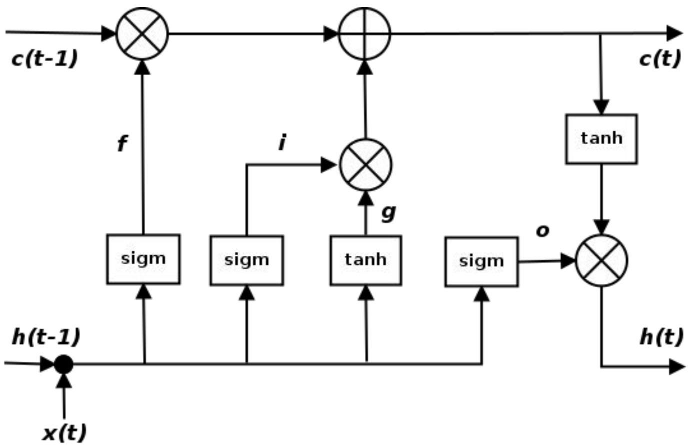
Recurrent Neural Networks

- Recurrent neural networks ideal for sequences like text data
- Yet, suffer from the twin **vanishing** and **exploding gradient** problems
- Vanishing gradient results in ‘short-term’ memory.



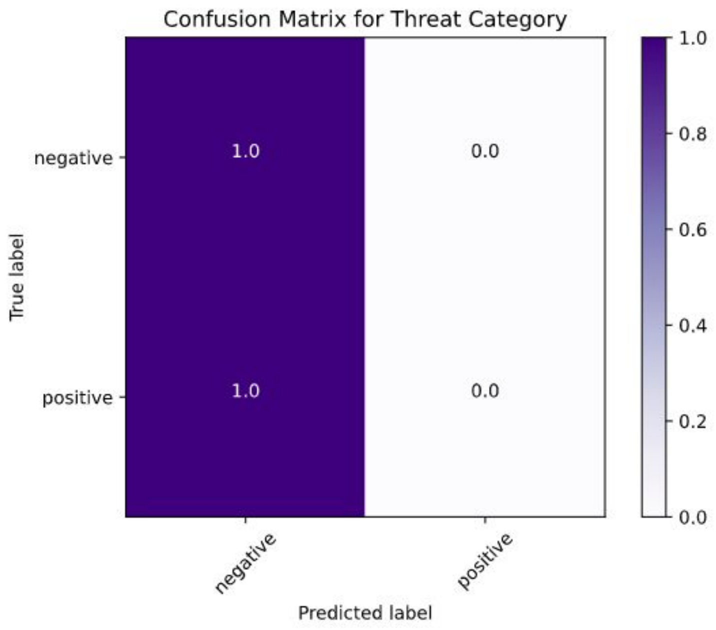
The LSTM Architecture

- The LSTM architecture is a solution to the problem of the vanishing gradient.
- Introduces an additional **‘cell-state’** connection between hidden nodes
- Uses a **‘gate’ mechanism**, to regulate memory.



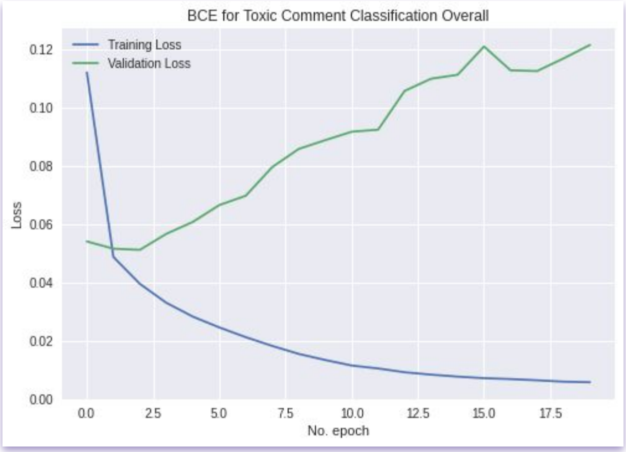
LSTM Model Implementation

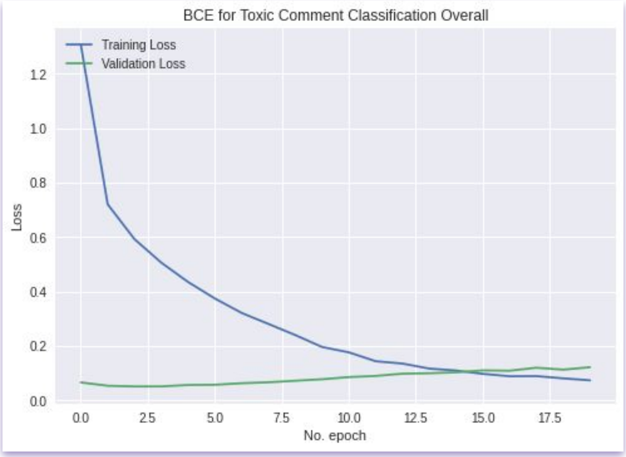
- Used the sigmoid activation function, in combination with the binary cross-entropy loss function.
- Chose the ‘Adam’ optimizer.
- Chose learning rate (0.002) which maximized AUC-ROC using trial-and-error.
- Best **accuracy** was achieved in 4 epochs, **recall** improved when trained over 20 epochs.



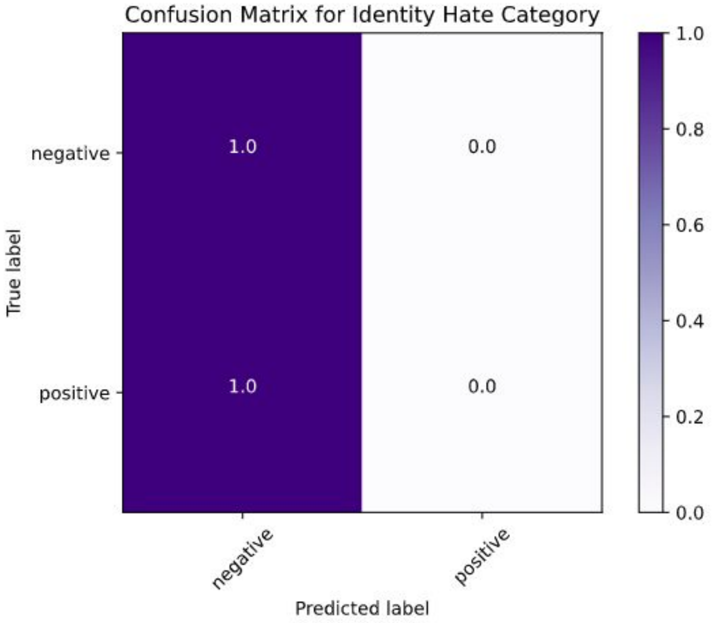
Model Tuning

- Adjusted for the class imbalances:
    - Initialized the biases for each class as the log(pos/neg), where
        - pos = ‘positive’ observations of the class, and
        - neg = ‘negative’ observations of the class
- Recall improved on the toxic, obscene, and threat categories, but worsened in the others.
- Model generalisability was vastly improved, as overfitting was limited.

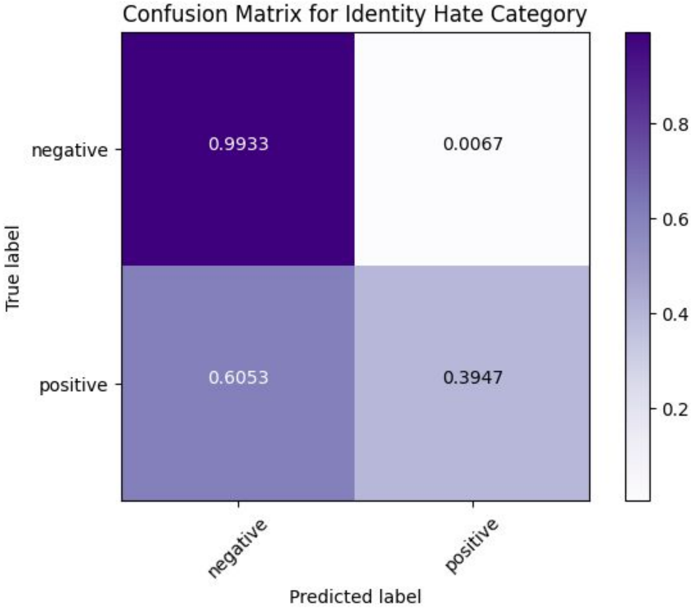




Performance Improvements



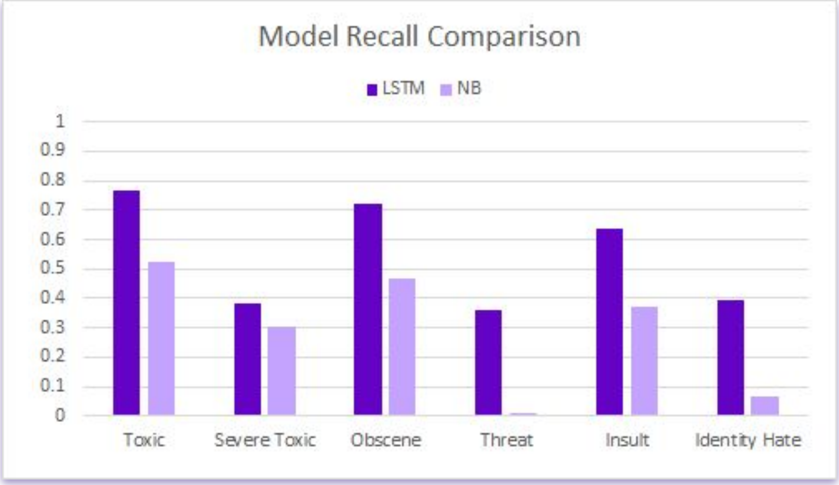
4 epochs - no adjustments to class weights



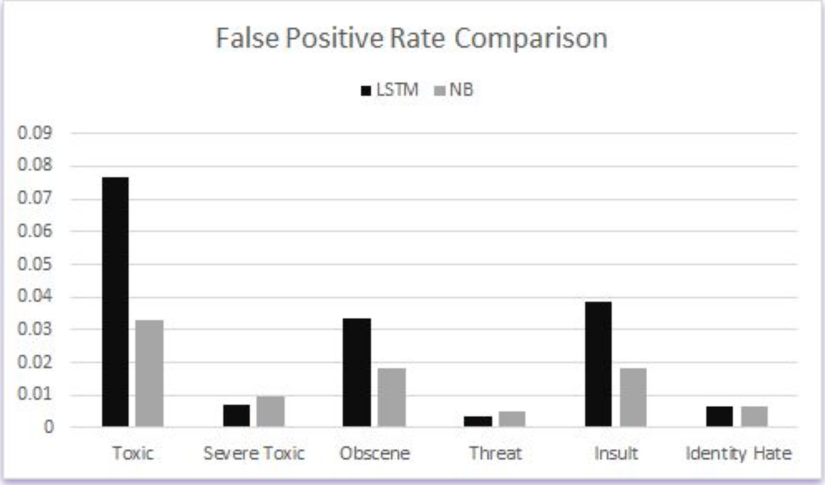
20 epochs - adjusted class weights and initialized biases

Comparison to Naive Bayes

Naive Bayes suffers from weaker recall

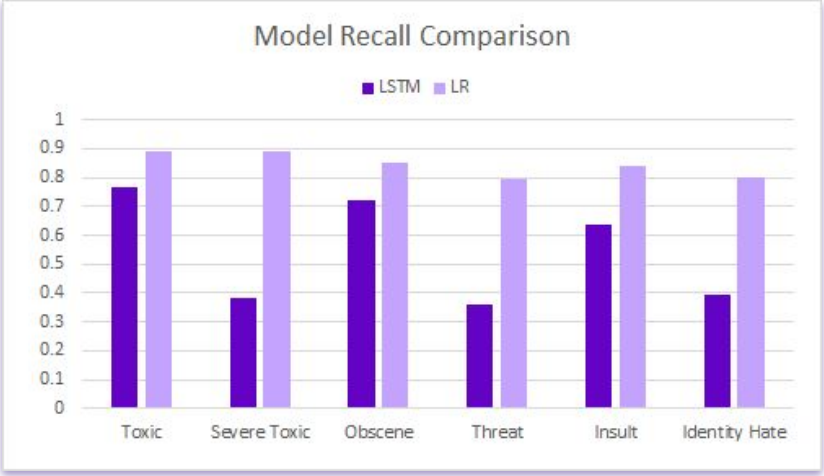


But has a lower false positive rate

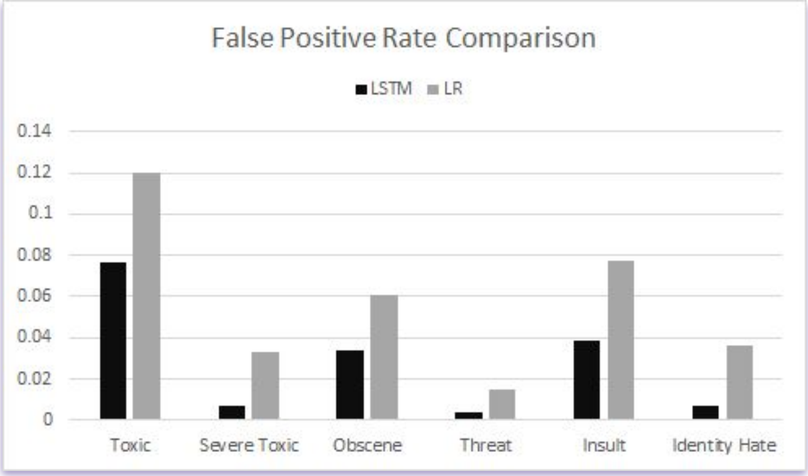


Comparison to Logistic Regression

LR outperforms the RNN in terms of recall



At the expense of a higher FPR



Conclusion

- Model selection depends on the objectives.
- Appears to be a trade off between model recall and FPR.
- **Minimizing false positives** may be more important than **maximizing recall.**
- Improvements could be made by using a custom ‘weighted binary cross entropy’ loss function.
- Alternatively, a combination oversampling and undersampling could help