Toxic Comment Classifcation

CMPT 413 Final Project
Group : outputerror
Group Member: Ao Tang, Shaoqiang Zou

Motivation

- The internet has enabled people to communicate and learn from each other
- Online hate, abuse, toxicity shuns people from opening up and taking full advantage of the opportunities that online communication provides
- Our goal is to use machine learning techniques to classify toxicity from the six types of toxicities

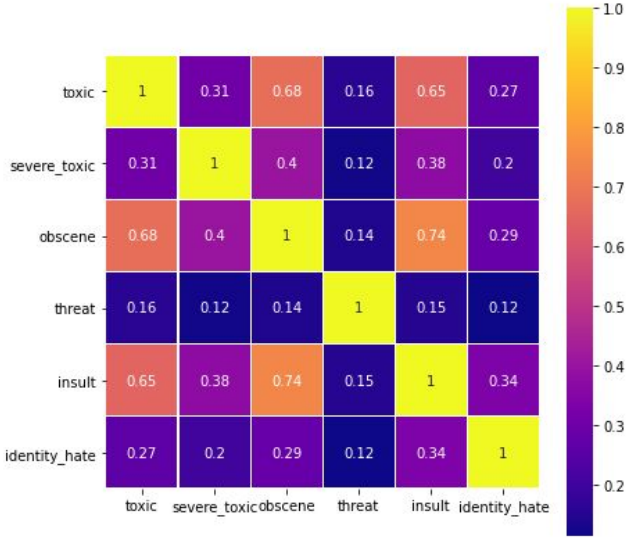
Dataset

- Comments from Wikipedia’s talk page
    - Train: 150k comments; Test: 60k comments
- Six types of toxicities are all boolean labels (0 or 1)

➢ toxic	➢ severe_toxic	➢ obscene
➢ threat	➢ insult	➢ identity_hate

Dateset

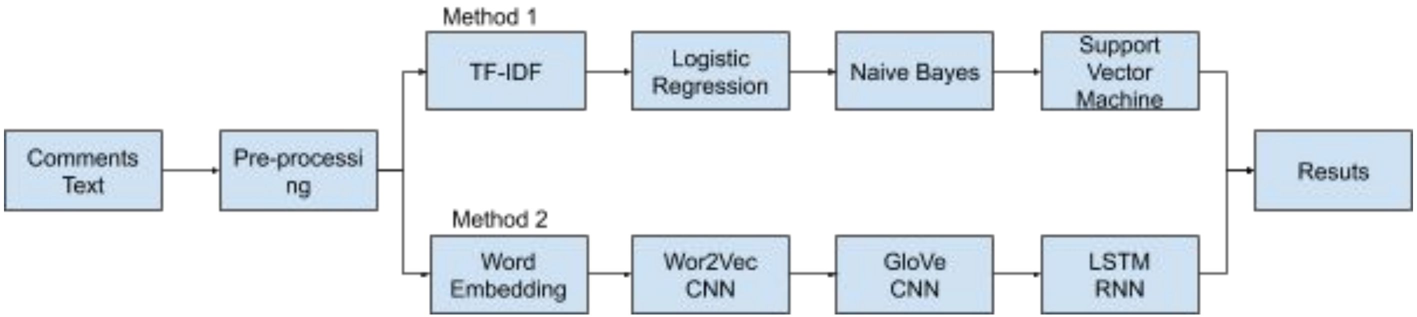
	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	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



Data Pre-processing

1. Translate Non-English word to English word use Google Translate API
2. Convert all character to lowercase
3. Replace emoticons with word
4. Replace Date, Phone Number, and Website Links
5. Remove all numbers and punctuations
6. Normalize repeating characters
7. Remove stopwords

Approch


Method 1
TF-IDF
Logistic Regression
Naive Bayes
Support Vector Machine
Comments Text
Pre-processi ng
Resuts
Method 2
Word Embedding
Wor2Vec CNN
GloVe CNN
LSTM RNN

Method 1: TF-IDF

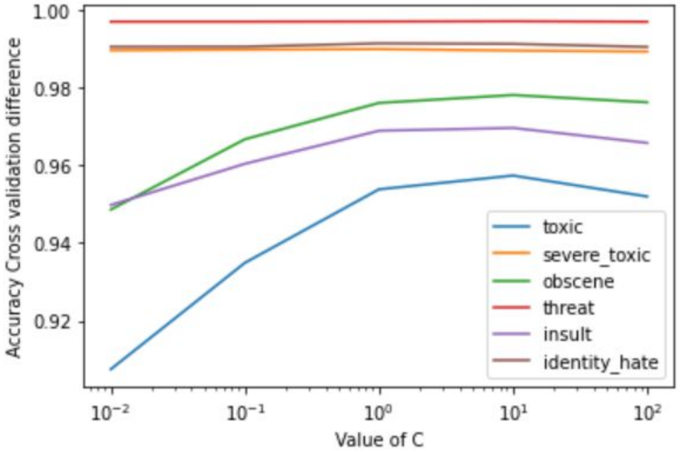
- **Comments in document is transform into Term Frequency/ Inverse document matrix**
- **Every unique word is considered as feature**
- **TF-IDF score can be fed to logistic regression, Naive Bayes, Support Vector Machine**

```
print(tf_vec.get_feature_names())
```

```
['aa', 'aaba', 'aabove', 'aacd', 'aachen', 'aachi', 'aadd', 'aadmi', 'aaffect', 'aafia', 'aaflight', 'aafs',
'aagadu', 'aages', 'aagf', 'aaghh', 'aagin', 'aah', 'aahank', 'aahh', 'aahs', 'aai', 'aaiha', 'aajacksoniv',
'aajonus', 'aakash', 'aake', 'aalborg', 'aalertbot', 'aalexa', 'aaliya', 'aaliyah', 'aaliyahremembered', 'aal
s', 'aalst', 'aamir', 'aand', 'aanda', 'aandahl', 'aandc', 'aande', 'aandm', 'aandr', 'aandw', 'aang', 'aannyy
wwhheerree', 'aany', 'aaot', 'aapropriate', 'aarabs', 'aarau', 'aardman', 'aardsma', 'aardsman', 'aardvark',
'aardvarks', 'aare', 'aarem', 'aargh', 'aarionrhod', 'aarne', 'aaroamal', 'aarohi', 'aaron', 'aaroncrick', 'aa
ronic', 'aaronsw', 'aaround', 'aarp', 'aarrggh', 'aarrow', 'aatc', 'aave', 'aaviksoo', 'aavishkaar', 'aaw', 'a
aww', 'aaye', 'aayege', 'abacha', 'abacination', 'aback', 'abaco', 'abacus', 'abad', 'abadan', 'abaddon', 'aba
de', 'abagnale', 'abalessa', 'abali', 'abanagi', 'abandon', 'abandonded', 'abandoned', 'abandoning', 'abandonm
ent', 'abandonou', 'abandons', 'abanes', 'abang', 'abantecart', 'abaranger', 'abarenoh', 'abase', 'abased', 'a
bassids', 'abatayo', 'abate', 'abaya', 'abba', 'abbas', 'abbasgulu', 'abbasid', 'abbasids', 'abbass', 'abbasse
d', 'abbassi', 'abbassid', 'abbastanza', 'abbau', 'abbe', 'abberations', 'abberline', 'abberrant', 'abbes', 'a
bbey', 'abbeys', 'abbi', 'abbie', 'abbitt', 'abbot', 'abbott', 'abbottabad', 'abbottandcostello', 'abbottsfor
d', 'abbr', 'abbrasive', 'abbrev', 'abbreveated', 'abbreviate', 'abbreviated', 'abbreviatedis', 'abbreviates',
'abbreviating', 'abbreviation', 'abbreviations', 'abbrevs', 'abbriviations', 'abbron', 'abby', 'abbyses', 'abb
ythecat', 'abbywinters', 'abcde', 'abce', 'abcedare', 'abcedere', 'abcmonster', 'abcnews', 'abdali', 'abdalla
h', 'abdallar', 'abdaly', 'abdalyar', 'abdel', 'abdelaziz', 'abdelbaset', 'abdelkader', 'abdi', 'abdicate', 'a
bdication', 'abdielcolberg', 'abdillah', 'abdin', 'abdnor', 'abdolhassan', 'abdolmalek', 'abdomen', 'abdomina
l', 'abdoozy', 'abdoreza', 'abdou', 'abdu', 'abducted', 'abducting', 'abduction', 'abductions', 'abductive',
'abdul', 'abdulaziz', 'abdulhakim', 'abdulhamid', 'abdulkadir', 'abdulla', 'abdullah', 'abdullahi', 'abdulrahm
```

Method 1: Logistic Regression

- Six models for each type of toxicity
- Use cross validation to tuning parameter



Method 1: Logistic Regression

- Used Accuracy for measuring model performance on test data set.
- Our results from logistic are following:

Type of Toxicity	Performance(Accuracy)
Toxic	0.9574
Sever_Toxic	0.9900
Obscene	0.9782
Threat	0.9972
Insult	0.9697
Identity_Hate	0.9915

- We also worked on other models, the average accuracy score are following:

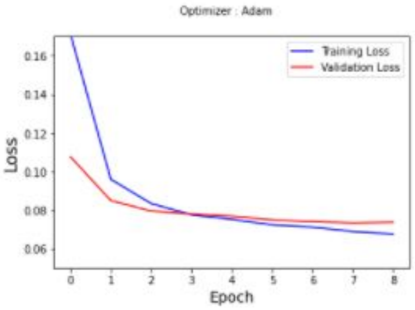
Models	Performance(Accuracy)
Logistic Regression	0.9722
Naive Bayes	0.9708
Support Vector Machine (SVM)	0.9721

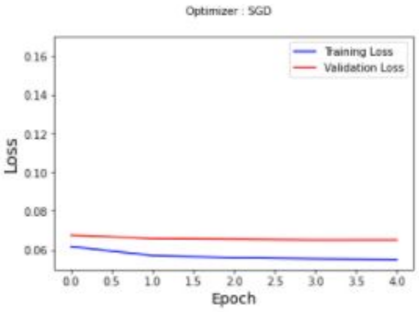
Method 2: Word Emmbedding

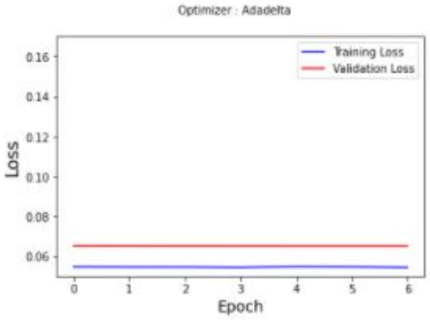
- Wod2vec - Convolution Neural Nets
  - Pre-trained model from Google
  - Contians 300 - dimensional vectors for 3 milion words
- GloVe - Convolution Neural Nets
  - Pre-trained model from Stanford
  - Contians 300 - dimensional vecotrs for 400,000 words
- Keras word embedding - Recurrent Neural Nets

Experiments Performed

- Early stopping using validation set
- Tried different optimisers (Adam, SGD, and AdaDelta)







Model Performance

Average accuracy score:

Model	Embedding	Validation	Test
CNN	word2vec	0.9789	0.9732
CNN	GloVe	0.9789	0.9731
RNN	keras word-embedding	0.9799	0.9722

Conclusion

- TF-IDF: Logistics Regression gives best performance.
- Word Embedding: Word2vec gives best performance.
- Overall, our word embedding method outperformed TF-IDF method.

Dataset Credits

Jigsaw/Conversation AI. 2017.
https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data