Toxic Comment Classification and Unintended Bias
Chady Ben Hamida, Victoria Ge, Nolan Miranda
{chadybh, vge, mirandan}@stanford.edu

Motivation

- Internet users find it much easier to propagate harmful stereotypes and toxic commentary in comment sections
- Unfortunately, this promotes an unhealthy environment online, and toxic commentary often ropes in language regarding minorities to construct insults

Problem Statement

- Construct a model that can accurately classify toxicity of unseen comments
- Find the bias with respect to the mention of certain identities

Data

- ~1.8 million data points taken from Civil Comments platform in 2017 hosted by Kaggle
- Each data point contains comment text and a classification label (from "very toxic" to "not toxic") across multiple graders aggregated into a score of 0.0 to 1.0
- Data points also binarize references to mentions of identity (e.g., "homosexual_gay_or_lesbian", "christian", "black")

Models

- **Naive Bayes** with Laplace smoothing: multivariate bernoulli model for bag-of-word count
- **Logistic regression**: Minimize cross-entropy loss with regularization and normal cost function on tf-idf
- **CNN**: used keras to create CNN of three convolutional layers. Interlaced with three MaxPooling layers to reduce dimensionality, speed up runtime, and to mitigate overfitting
- Employed dropout as a method of regularization and measured loss using categorical cross-entropy (softmax + cross-entropy)
- Ran each CNN for 7 epochs with batch size 128

Features

- Bag of words: vectorizes comments in context of the total vocabulary to model term frequency
- TF-IDF: vectorizes comments in context of the total vocabulary to model relative term importance
- GloVe and fastText: downloaded pre-trained embeddings for more compact vectors gathered from complex NNs
- Identities: binarized mentions of various identity markers

Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Embeddings</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>(Bag-of-words)</td>
<td>0.9020</td>
</tr>
<tr>
<td>Logistic Reg</td>
<td>(TF-IDF)</td>
<td>0.9474</td>
</tr>
<tr>
<td>CNN</td>
<td>GloVe 50d</td>
<td>0.9390</td>
</tr>
<tr>
<td>CNN</td>
<td>GloVe 100d</td>
<td>0.9450</td>
</tr>
<tr>
<td>CNN</td>
<td>fastText 300d</td>
<td>0.9484</td>
</tr>
</tbody>
</table>

Discussion

- Pre-trained word embeddings added comparably high improvements in accuracy, but higher dimensions increased runtime for little gain
- Our best CNN performed with 94.84% accuracy with word embeddings of 300 dimensions (fastText)
- Logistic regression performed very well, with a higher accuracy than a few of the CNN models
- While complex models can learn the problem decently, the simple model (logistic regression) ran about 10 times faster with similar results
- bsn_auc (Background Negative, Subgroup Positive) and bpsn_auc (Background Positive, Subgroup Negative) are both metrics for unintended bias

Future

- We will consider improving the accuracy of our algorithm by using recurrent neural networks, namely, BLSTM which would allow us to integrate both past and future context in our model
- We can consider translating our comments to other languages then back to English as a way to increase the dataset and make it more generalized

References