Deep Learning Based Sarcasm Detection

Rohan Bais {rbais@stanford.edu}, Daniel Do {dktd@stanford.edu}
Stanford University

Problem

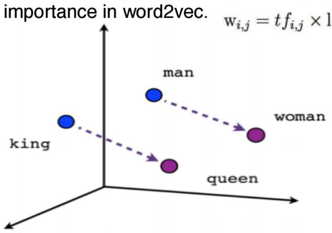
- Sarcasm Detection is a relatively untouched subject in NLP due to its difficulty. With the advent of deep learning, we wanted to see if binary sarcasm classification can be detected through neural networks. We built 4 models using CNNs, RNNs, and standard neural networks (with and without TF-IDF) to try and see the implications for each of these types of networks. Using 1.3 million sentences as our dataset and word embeddings as features, we were able to see that RNNs performed best followed by CNNs and lastly our baseline.

Data

Used a dataset of 1.3 million sarcasm-labeled Reddit comments from Kaggle. Consisted of comments, parent comments, username, subreddit,
and time for each labeled comment. Approximately half of comments are sarcastic, and half are not sarcastic.

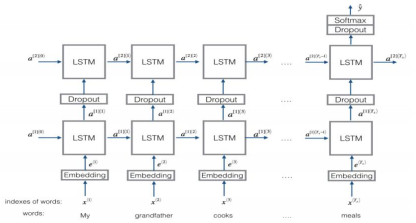
Features

- We used words as our only feature since we were interested in seeing if our model could predict sarcasm using only words.
- We used the comments from the available features and trained a Word2Vec model on the training set's words.
- Derived a TF-IDF weighting scheme to amplify certain words' importance in word2vec. $w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$



Models

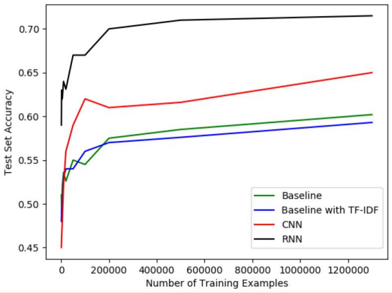
- **Baseline:** 2-layer 25 unit neural network, with sentence vector as an average of individual word vectors (with and without TF-IDF weights)
- **CNN:** Neural network convoluting over a matrix of stacked word2vec's,1-max pooling into a fully connected layer.
- **RNN:** 2-layer 128-cell LSTM RNN using word2vec as an embedding matrix, with dropout in between layers before softmax.


$\hat{y}$
Softmax
Dropout
$a^{[2]\langle 0\rangle}$
LSTM
$a^{[2]\langle 1\rangle}$
LSTM
$a^{[2]\langle 2\rangle}$
LSTM
$a^{[2]\langle 3\rangle}$
....
$a^{[2]\langle T_x-1\rangle}$
LSTM
$a^{[2]\langle T_x\rangle}$
Dropout
$a^{[1]\langle 1\rangle}$
Dropout
$a^{[1]\langle 2\rangle}$
Dropout
$a^{[1]\langle 3\rangle}$
Dropout
$a^{[1]\langle T_x\rangle}$
$a^{[1]\langle 0\rangle}$
LSTM
$a^{[1]\langle 1\rangle}$
LSTM
$a^{[1]\langle 2\rangle}$
LSTM
$a^{[1]\langle 3\rangle}$
$a^{[1]\langle T_x-1\rangle}$
LSTM
$e^{\langle 1\rangle}$
Embedding
$e^{\langle 2\rangle}$
Embedding
$e^{\langle 3\rangle}$
Embedding
$e^{\langle T_x\rangle}$
Embedding
indexes of words:
$x^{\langle 1\rangle}$
$x^{\langle 2\rangle}$
$x^{\langle 3\rangle}$
$x^{\langle T_x\rangle}$
words:
My
grandfather
cooks
....
meals

Results

- Used approximately a 90/5/5 train/dev/test split on 1.3 million comments which is 1.2 million, 50k, 50k comments approximately.

Method	Train Acc.	Test Acc.	Precision	Recall	F-1
Standard NN	65.7%	60.2%	60.4%	58.7%	59.6%
NN w/ TFIDF	65.4%	59.3%	58.6%	59.1%	58.8%
CNN	66.7%	65.4%	68.2%	57.6%	62.5%
RNN	75.8%	71.9%	77.7%	69.5%	73.4%



Discussion

- TF-IDF didn't significantly improve performance on the baseline, so important words are not necessarily indicative of sarcasm.
- CNN did better than baseline in the long run and had less variance but still had high bias.
- Convolutions capture better relationships with consecutive words than previous two with average vectors.
- CNN, Baseline, and Baseline with TF-IDF had low recall, indicative of trouble distinguishing between sarcastic and not sarcastic for all sarcastic sentences.
- LSTM RNN is a cut above the rest, with higher training and test accuracy and higher precision/ recall than previous methods, RNN properly captures the sequential nature of NLP the best, as expected, so it was most suited despite high bias.
- Overfitting on some methods due to specific sarcasm being related to subreddits, like political vs gaming, and makes it hard to generalize.

Future Work

- Use other non-word-based features in the data and compare it to the sole word2vec models and other word vector representations (GloVe).
- Try attention model to compare with LSTM.
- Hyperparameter search with a deeper number of layers for all 3 neural networks.

References

[1] Y. Kim "Convolutional Neural Networks for Sentence Classification" in *EMLP 2014*, Ithaca, NY, 25 August 2014

[2] T. Mikolov. "Distributed Representations of Words and Phrases and their Compositionality", NIPS, 2013, pp. 1 - 5