Predicting Audience Reactions to a Speech

Samuel Lurye
Stanford University
slurye@stanford.edu

March 17, 2019

Abstract

The purpose of this project is to create a model to predict a sentence-level audience reaction to a written speech. This model could be immensely useful to a politician or executive because it provides a heuristic for the effectiveness of the speech. This project applies a recently developed, computationally efficient, context-encoding model- originally intended for medical document classification- to the CORpus of tagged Political Speeches. The organizational complexity of speeches is overcome by processing large amount of context to effectively classify audience reaction. Model experiments indicated that increasing context size is of limited importance in accurately predicting audience responses. Overall, the model achieved moderately successful classification results.

1 Introduction

Speeches are often intended to provoke an emotion or action in their audience, so a predictive analysis ahead of delivery can be invaluable to successfully driving people towards the intended goal. Audience reaction is the simplest indication of more complex internal feelings: a person typically cheers or claps when they are pleased, and conversely boos when they are displeased. The model in this project intends to predict how specific arrangements words influence human emotion in the context of political speeches, a powerful tool for politicians and influencers in general.

Historically, a project with similar scope was limited by two factors: lack of sufficient labeled data, and an efficient neural architecture to capture of idiosyncrasies of political speeches. The former limitation was overcome by the creation of the CORpus of tagged Political Speeches (CORPS) in 2010. The latter limitation stems from the unique structure of political speeches. Standard challenges of understanding spoken word are compounded by unconstrained references and quick topical shifts.

This project seeks to successfully overcome

the organizational and semantic complexities of political speeches by implementing the Context-LSTM-CNN model to incorporate lots of leading and lagging context into its audience-reaction predictions.

2 Related Work

2.1 Sentence Classification

The underlying problem type of this project-sentence classification- has been the focus of NLP research from the outset. There have been a plethora of architectures suggested, each tailored made to the unique flavor of classification problem at hand. This section describes earlier models for reaction-based sentence classification.

Initially outlined by Yoon Kim in 2014, the CNN based approach for sentence classification has long been standard. Yoon achieved a high of 88.1% accuracy using a 1 convolutional layer on a simple positive/negative classification task, and set the standard for classification architectures to come. [Yoon, 2014] However, convolution did not maintain long distance dependencies in sentence and paragraph structure. [Song, 2018] Sainath et al. combined the traditional CNN architecture with an LSTM, showing that a combined model decreased word-error rate by as much as 6% on speech-search tasks. [Sainath, 2015]

Lee and Dernoncourt demonstrated the effectiveness of including context in improving accuracy of classifying short texts. [Lee, 2016] However, their approach would incur significant computational costs if large history was incorporated into the model. [Song, 2018] Instead, Song et. al decided to use FOFE encoding first put forward by Zhang et al, maintaining performance en-par with traditional RNN architectures without increasing computational complexity with size of context. [Zhang, 2015]

2.2 Political Speech Analysis

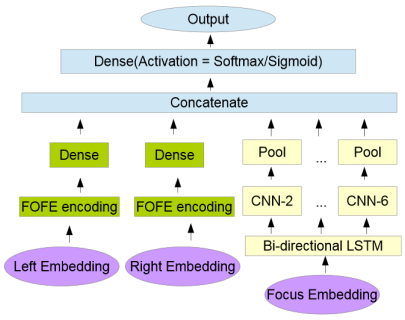
Strapparava et al, the creators of the CORPS, ran a variety of experiments on the data-set to examine NLP's ability to judge persuasiveness in political speeches. The experiment showed that a model can be trained to separate Republican and Democrat speeches. Following up on this model, the paper outlines a basic classifier used to predict audience reaction grouped into three classes: positive-ironical, neutral and micro. The classifier achieves 0.625, 0.658 and 0.641 precision respectively, leading the team to claim that the "results show that this could be a viable way of studying the persuasive power of discourse." [Strappava]

3 Approach

This project implements three different architectures to show a development of sophistication and performance as each architecture improves on the flaws of the previous design.

1. Two-layer CNN (Baseline)
2. LSTM-CNN
3. Context-LSTM-CNN

Figure 1: C-LSTM-CNN Model Overview



Neither architecture [1] or [2] will be given context, but rather trained to predict audience reaction based solely on the features itself.

The C-LSTM-CNN architecture [3] [Figure 1], outlined in the Song et. al paper [2], is the main focus of this paper. The C-LSTM-CNN architecture is the most innovative as it efficiently incorporates large amounts of context without linearly increasing computational costs.

3.1 Pre-processing

3.1.1 Data Distribution

In order to simplify the problem, all available tags were grouped into four primary categories: Applause, Laughter, Boos, Audience Participation. Per Figure 2, it is evident that that there is mis-distribution of tags between the four classes. In order to not throw away valuable data, the data was artificially sorted so that trainging/dev/test set all maintained the same data distribution.

Moreover, the tags represent 0.0084 density in the speeches. This means that the vast majority of sentences do not have a (notable) audience reaction. In order to not over-train on null-response sentences, 45,000 null-response sentences were chosen at random to include in the final distribution. Each sentence- with its leading and lagging context- was treated as an independent data-point. Speech information was not encoded as input.

3.1.2 Lemmatization and Padding

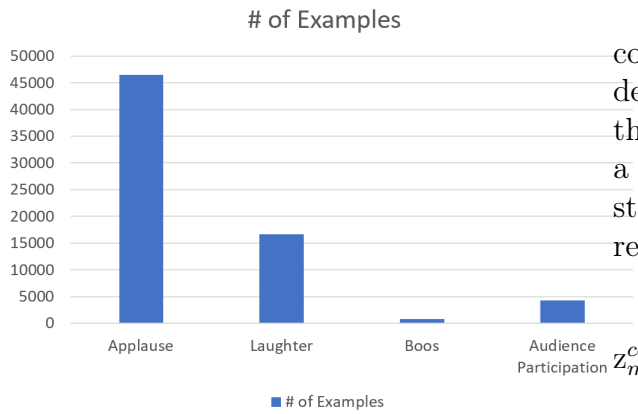
Nltk.stem.WordNetLemmatizer was used in order to transform inflected words into their original root word so that we can pull the correct wort embedding from the Word2Vec library.

The sentences were set to a max length of 30 words. If the sentences were shorter, they were padded with a PAD token. If a sentence within the context was longer than 40 words, it would be split into two sentences and treated accordingly.

3.2 LSTM-CNN Encoding

The right-hand side is a standard LSTM-CNN pairing common in many sentence classification tasks. We convert each word in a target sentence into an embedding using Word2Vec and feed through a bi-directional LSTM to get forward and backward hidden states for each cell. These states are concatenated and passed as inputs to five independent CNN layers, each with a different kernel size (varying from 2 to 6). The CNN independent layers with max-pooling are de-

Figure 2: Class Distribution



signed to extract features from every part of the sentence and catch different sized inter-dependencies.

3.3 FOFE Encoding

The main innovation in this architecture is the use of FOFE encoding. The embedding z for a sentence $(x_1, x_2, \ldots x_U)$ is initialized to $z_1 = x_1$, and then calculated recursively for $u \in 2 \ldots U$ as

$$|\mathrm{z}|_U = \alpha * z_{(u-1)} + x_u \quad (1)$$

The parameter α is the forgetting factor. This puts heavy bias on sentences more local to the target sentence while keeping the importance of all words within the sentence the same. The dense layer provides a hidden layer that can be concatenated with the encoding of the target sentence and then classified.

For the FOFE encoding, we start by encoding each sentence into z^{sent} with a slowly-decreasing α_{sent}. These encodings are then themselves encoded into the embedding using a rapidly decreasing α_{sent}. This is calculated starting with $z_1^{cont} = z_1^{sent}$ and is calculated recursively for $m \in 2 \ldots |C_{left}|$ where

$$\mathrm{z}_m^{cont} = \alpha_{sent} * z_{(m-1)}^{cont} + z_m^{sent} \quad (2)$$

Although the model architecture is not original to this project, all the pre-processing code and architectures are implemented from scratch. Additionally, all the hyper-parameter and architectural changes (described in experiments section) are original.

3.4 Model Training

The scores are transformed to a conditional probability distribution of labels by applying softmax over the scores for each crowd reaction label. The networks are trained with stochastic gradient descent to minimize the negative log likelihood. In the end, the model outputs one label of audience reaction based on which conditional probability is higher.

4 Experiments

4.1 Data

CORpus of tagged Political Speeches (CORPS). The political speeches are tagged

Figure 3: Data Statistics

Total number of speeches:	3,618
Total number of speakers:	197
Total number of words:	7,901,893
Total number of tags:	66,082
Tag density (μ):	0.0084
PF-density (μ):	0.0062
I-density (μ):	0.0020
NF-density (μ):	0.00015
Temporal range (μ):	from 18/05/1917 to 16/09/2010

with audience reactions, like "Applause" or "Boos." The statistics of the data set are included above [Figure 2]:

60% of the data set was set aside for training, 20% for validation and 20% for test. This ratio was chosen because our data set is not that large. Data was provided courtesy of the Carlo Strapparava and is not publicly accessible. A link can be provided on request. [Guerini, 2013]

4.2 Evaluation Metrics

In order to evaluate individual model performance, we used the following metrics:

$$Accuracy = \frac{true\ positives}{All\ Examples} \tag{3}$$

$$Precision = \frac{true\ positives}{true\ positives + false\ positives} \tag{4}$$

4.3 Experiment Details

Summary of different experiments can be found in Table 1.

Table 1: Hyperparameter Tuning

Hyperparameter	Choice	Tested
LSTM Hidden Size	100	50-300
CNN Stride	2	1-5
Learning Rate	0.001	0.001-0.01
Dropout Rate	0.3	0-1
CNN Pooling	Max	Max-Mean
CNN Output Channels	10	5,10,20,50

We experimented with a couple of LSTM hidden sizes. 100 was the optimal trade-off between information capture and training efficiency. We believe that increasing hidden state size could capture final details about sentence structure, but would take 2x-3x time to train. Decreasing learning rate did increase training time 1.4x but was justified for the consistent training. Given the complexity of the architecture, it became clear that it was robust in isolation. Adding a large dropout probability only inhibited learning.

Aside from the hyper-parameter tuning, the key experiment was the amount of context that was fed into the FOFE encoder.

4.4 Results

All results can be found in Table 2. The CNN baseline model performed as expected, relative to a baseline set out in Strappava. The LSTM-CNN model was a minor improvement on the baseline model, but did not show the significant gap that the architecture improvement would suggest.

Figure 4: Context vs. Accuracy

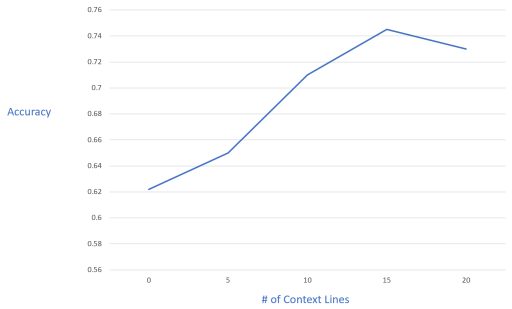


Table 2: Model Test Accuracy and Precision

Model	Accuracy	Precision
CNN (baseline)	0.604	0.62
LSTM-CNN	0.622	0.701
C-LSTM-CNN (best)	0.745	0.76

We ran four different version of the C-LSTM-CNN model, varying the total leading and lagging context used. The values in the table represent the optimal context of our experiment, 15 lines. The other test-accuracy values are plotted in Figure 4 and are discussed in the analysis section.

5 Analysis

5.1 Error Analysis

5.1.1 Bad Context

Sentence: "He would emphasize that I was height-challenged to begin with."
Leading Context: NULL
Lagging Context: "You're a hard act to follow. Thank you, Mr. President. Ladies and gentlemen, I am filled with immense pride as I stand here in our Capitol City today and take part in this national celebration on teaching and teachers..."
Tag: Laughter
Prediction: Applause

The model recognized that this line has some audience reaction, but it mis-classified the type. This example it difficult for the model because there is no leading context (it is the beginning of the speech) and uses indicator words like "emphasize" that may throw the model off.

5.2 Per Class Accuracy

Overall, the model has moderate success in accomplishing the desired task. The model showed a 0.141 improvement in accuracy over the baseline, which is a notable improvement. After optimizing the hyper-parameters, it became clear that the skewed distribution of the training data did not contain sufficient examples of Boos and Audience Participation classes to effectively train, and so it overfitted on Null and Applause classes [Figure]. Upon further reflection, there seems to be little pre-processing improvement to be made, and the bottleneck remains the under-representation of some data classes.

5.3 Context Saturation

The most surprising outcome of the model can be seen in Figure 4. For the optimal set of hyperparameters, there appears to be a saturation point of 15 lines of context. Although not intuitive, the existence of a sat-

uration point is prudent. Initially, raising the amount of context increases the amount of predictive information that the FOFE encoding can capture, increasing accuracy numbers. However, there comes an inflection point where further context doesn't possess predictive value and instead acts to dilute the value of the FOFE encoding.

6 Conclusion

Although the peak accuracy of 0.745 would not suggest that our model is a practical tool for a politician or leader in its current iteration, it is a notable result for the task. The model has moderate ability to predict how a human- a complex composition of signals- will do based on a set of characters. This project shows that there is potential for further exploration in word-based prediction of human reaction. Future work would depend on the availability of more data and the development of an even more effective encoding methodology.

7 References

[1] Guerini, M., Giampiccolo, D., Moretti, G., Sprugnoli, R., Strapparava, C. (2013). The New Release of CORPS: A Corpus of Political Speeches Annotated with Audience Reactions. Lecture Notes in Computer Science Multimodal Communication in Political Speech. Shaping Minds and Social Action, 86-98.

[2] Lee, Ji Young and Franck Dernoncourt. 2016. Sequential short-text classification with recurrent and convolutional neural networks. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

[3] Sainath, Tara et al. 2015. Convolutional, long short-term memory, fully connected deep neural networks. In IEEE International Conference on Acoustics, Speech and Signal Processing.

[4] Song, Xingyi et al. "A Deep Neural Network Sentence Level Classification Method with Context Information" ACM, 2018.

[5] Strapparava, C., Guerini, M., Stock, O. (n.d.). Predicting Persuasiveness in Political Discourses.

[6] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In The 2014 Conference on Empirical Methods in Natural Language Processing.

[7] Zhang, Shiliang et al. 2015.The fixed-size ordinally-forgetting encoding method for neural network language models. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conferenceon Natural Language Processing.