Deep Learning Based Sarcasm Detection Using Contexts

Bhargav Ganguly
Department of Industrial Engineering
Purdue University
West Lafayette, IN
bganguly@purdue.edu

Abstract

At a high level of abstraction, sarcastic text conveys a meaning which is often interpreted as the metaphorical version of the original statement. In this regard, several learning models have been proposed which take into account the underlying relation of the words in the statement to develop a forecasting methodology to identify sarcasm. This work is an attempt towards building an understanding of how some of well known Deep Learning based computational methods in Sarcasm detection work in practice.

1 Introduction

In every language, inherent sarcasm heavily depends on the figurative nature of the context and the conversational discourse that it follows also reveals significant cues. The non-literal nature of the interlinked contextual data has traditionally made it a difficult task for any kind of sentiment analysis model to solve the problem. Furthermore, sarcasm which is heavily dependent on background knowledge of the linked topic also makes this task a non-trivial and rather highly challenging.

In literature, a significant amount of study has focused primarily on hybrid deep learning based computational networks to develop a formal realisation of the intuitive connection of the words and the utterance. In our work, we look at some of those techniques which are rather focused on the rich amount of data contained in online discussion forums, particularly Reddit.

2 Related Work

In Sarcasm detection, our analysis delves deeper into the works of [Ilic et al., 2018], [Ghosh et al., 2017] which are considered *State-of-the-Art* among the LSTM based methods. And our experiments are fully based on the Reddit discussion forum dataset which has been prepared using actual posts as well as the comments that came up as part of the conversation threads [Khodak et al., 2017]. [Ghosh et al., 2017] builds on the idea of an attention based architecture which seperately attends to both the actual comment and the conversation discourse. And the LSTM architecture largely attempts to develop the connection between the two texts. [Ilic et al., 2018] describes a rather simpler approach with just a single BiLSTM which processes the concatenation of the post itself with the response thread followed by feed forward fully connected layers to classify sarcasm. Emperically it has been found that lexical and syntactic hints such as quotes, interjections, all-caps text etc. can be obtained from utterances and interjections which are critical aspects of sarcastic content [Bharti et al., 2015].Deep Learning approaches such as convolutional network followed by LSTM architecture (CNN-LSTM-DNN) proposed by [Ghosh and Veale, 2016] has also been found to efficiently handle the problem. A rather naive approach with **k-nearest neighbor** based learner model with extra focus

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

on high frequency words and important features has also been shown to produce modest results [Tsur et al., 2010]. [Wu et al., 2018] has used a combination of several stacked LSTMs followed by multi-task learning approach to produce results that are indeed *State-of-the-Art* in sarcasm detection literature.

3 Approach

3.1 Word Embedding Vectors

Our word vector generation process is based on **Embeddings from Language Model** or ELMo [Peters et al., 2018]. ELMo is used to generate 512-dimensional rich vector for each word in the tokenized sentences of the parent context and the subsequent child response text. ELMo already contains a pre-trained bi-directional LSTM architecture bolstered by a subsequent CNN layer, whereby it takes in a purely character-based input and generates feature for each word in the sentence as output while taking into account the neighborhood information. As a matter of fact, the rationale behind using this architechture is strengthened by it being trained on 800M tokens of news crawled data [Chelba et al., 2013]. In our implementation of [Ilic et al., 2018], we have concatenated the pair of responses associated with each post to generate a 1024-dimensional word averaged feature vector. For [Ghosh et al., 2017], our models leverages a 512-dimensional vector of the actual post and the concatenated 1024-dimensional vector for the pair of responses as described before.

3.2 Models

Bi-directional Vanilla LSTM model In [Ilic et al., 2018], a single Bi-directional LSTM model with two layers stacked one upon the other is used. The input is simply is the response vector. In this model, the outputs of the last layer is used as the intial input of the next stacked layer.

Bi-directional Conditional LSTM model In [Ghosh et al., 2017], the approach $LSTM_{conditional}$ consists of a combination of two double layered Bi-LSTMs same as described above where the $LSTM_{reply}$ receives the last hidden output and memory cell output of $LSTM_{originalpost}$ as part of the initial input. The conditional encoding model is heavily inspired from the *conditional encoding* architechture first proposed for solving the problem of directional relation between sentence fragments in [Rocktäschel et al., 2015].

Attention Layer We have dot product based Attention model for our implementation which is widely used in theory as well as practice. They are described by the below equations as explained in [Yang et al., 2016]:

$$u_i = tanh(W_s h_i + b_s) \tag{1}$$

$$\alpha_i = \frac{exp(u_i^T u_s)}{\sum_{j \in [d]} u_j^T u_s} \tag{2}$$

$$v = \sum_{i \in [d]} \alpha_i h_i \tag{3}$$

This attention layer is particularly important in rewarding training sentences which have enough syntactic cues hinting toward degree of sarcasm. Here, u_s acts as an additional context vector that can be later used to quantify the importance of the various chunks within the context/response sentence. And, v summarizes the contribution that each of this chunk has to the overall non-literal semantic significance of the sentence.

Feed-Forward Layers Beyond the attention layers, a couple of fully connected linear feedforward layers have been used to ultimately generate 2 outputs for softmax based classification.

4 Experiments

Data In our experiments, we have used 100,000 datapoints from the training dataset made available in Self-Annotated Reddit Corpus (SARC) dataset [Khodak et al., 2017] with 9:1 training to validation size ratio for the purpose of training our models. And, we have subsequently tested our model

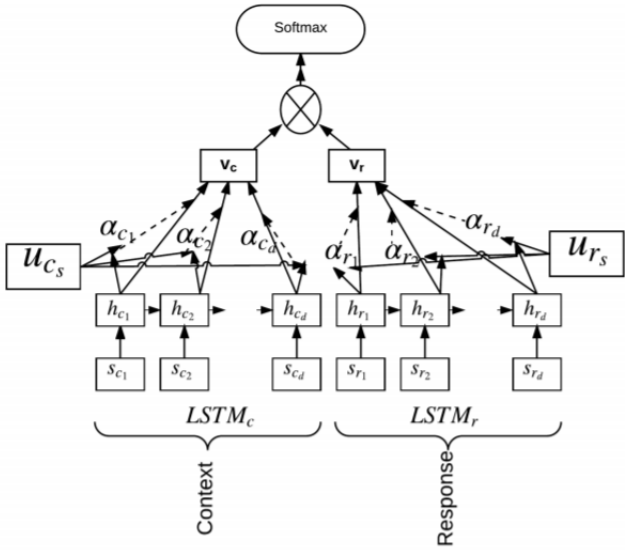


Figure 1: $LSTM_{conditional}$: Sentence Level Attention based LSTM model combining context and response as proposed in [Ghosh et al., 2017]

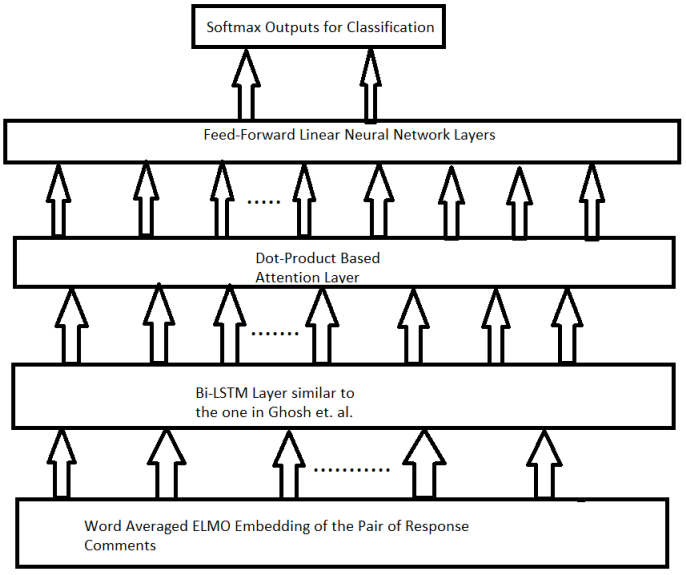


Figure 2: $LSTM_{reply}$: Attention based LSTM model with only response as input

on 10,000 datapoints from the test set. This dataset has 1.3 million comments with both balanced and unbalanced versions of training and test sets, and thus provides a broader scope to develop an understanding of the task as compared to other previous datasets which are atleast 10 times smaller in term of data quantity.

Pre-processing We have used the standard **Nltk** tokenizer in Python to tokenize each of the training sentence-response pair. The tokenizer essentially removes all the uninteresting stop words and keeps only the words including the interjection characters which are critical to identification of sarcasm.

Evaluation and Comparison Method We have trained our model on binary cross-entropy loss and have used also used accuracy as the metric to do a comparative performance analysis of our models.

Hyperparameter details We manually tried 20 different combinations of the hyperparameters on our training data to identify reasonable values for golden set of experiments. All the LSTM layers are trained with dropout rate of 0.2. For each of the layers of the Bi-LSTMs, the hidden unit size has been kept same at 256. And, the sequence length is fixed at 16. The Learning rate is kept constant at 0.001 across all the 20 epochs. We have used batch size of 500 with Adam optimizer for mini-batch gradient descent.

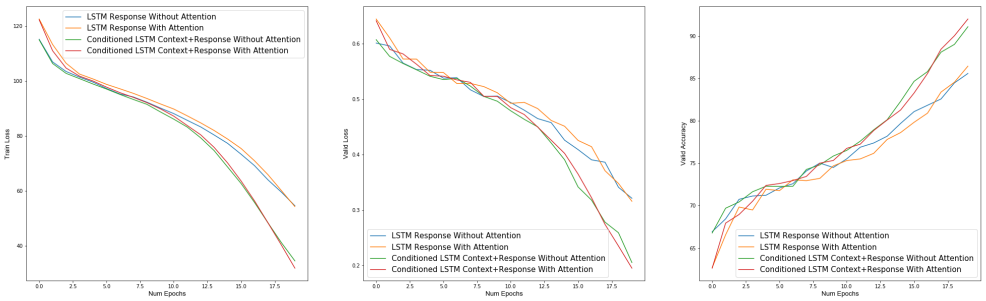
4.1 Results

The results of our experiments are in the table below:

Model	Attention	Valid Accuracy	Test Accuracy
$LSTM_{reply}$	NO	85.6	70.57
$LSTM_{conditional}$	NO	91.11	68.74
$LSTM_{reply}$	YES	86.46	71.35
$LSTM_{conditional}$	YES	92.02	69.39

Table 1: Performance statistics based on best validation epoch model

The training/validation curves are presented below. We can observe that during training with the mentioned parameters, the $LSTM_{conditioned}$ with Comment-response pair outperforms the $LSTM_{response}$ model in terms of training loss, validation loss and accuracy. However, the performance on unseen test set does not give conclusive evidence regarding better generalization ability of the conditioned architechture over the vanilla architechture using Bi-LSTM. Also, we note that in this particular setting, our implementation achieves minimal gains with addition of the dot product based attention architechture. However, we have been able to almost completely replicate the results in the original papers.



(a) Training Loss (b) Validation Loss (c) Validation Accuracy

5 Conclusion

In this work, we have explored two approaches towards identifying sarcasm by modelling the contextual information around it. The work is limited by the scope of the amount of data processed during training, hence the results point towards no clear conclusion. However, based on the current analysis, our conjecture is that the conditional model has an underlying overfitting nature which is clearly identifiable in it's much higher validation accuracy with no improvement in the test performance. Going ahead, it will also be interesting to see how word embedding models such as GLoVe [Pennington et al., 2014] would perform in this setting.

References

Santosh Kumar Bharti, Korra Sathya Babu, and Sanjay Kumar Jena. Parsing-based sarcasm sentiment recognition in twitter data. In *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1373–1380, 2015. doi: 10.1145/2808797.2808910.

Ciprian Chelba, Tomás Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. One billion word benchmark for measuring progress in statistical language modeling. *CoRR*, abs/1312.3005, 2013. URL `http://arxiv.org/abs/1312.3005`.

Aniruddha Ghosh and Tony Veale. Fracking sarcasm using neural network. In *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*, pages 161–169, 2016.

Debanjan Ghosh, Alexander Richard Fabbri, and Smaranda Muresan. The role of conversation context for sarcasm detection in online interactions. *CoRR*, abs/1707.06226, 2017. URL `http://arxiv.org/abs/1707.06226`.

Suzana Ilic, Edison Marrese-Taylor, Jorge A. Balazs, and Yutaka Matsuo. Deep contextualized word representations for detecting sarcasm and irony. *CoRR*, abs/1809.09795, 2018. URL `http://arxiv.org/abs/1809.09795`.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. A large self-annotated corpus for sarcasm. *CoRR*, abs/1704.05579, 2017. URL `http://arxiv.org/abs/1704.05579`.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL `https://www.aclweb.org/anthology/D14-1162`.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*, 2015.

Oren Tsur, D. Davidov, and A. Rappoport. Icwsm - a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *ICWSM*, 2010.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Junxin Liu, Zhigang Yuan, and Yongfeng Huang. THU_NGN at SemEval-2018 task 3: Tweet irony detection with densely connected LSTM and multi-task learning. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 51–56, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/S18-1006. URL `https://www.aclweb.org/anthology/S18-1006`.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489, 2016.