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

Preprint. Under review.

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. The rationale behind using this architechture is strengthened by it being trained on 800M tokens of news crawled data [Chelba et al., 2013]. 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 pipeline consisting of a single Bi-LSTM network followed by aggregation and classification layers has been proposed for the sarcasm detection task. We note that, in our implementation of [Ilic et al., 2018], we have merged the set of contexts and responses associated with each post to create the concatenated vector of words. Words within the input sentences are transformed to contextualized embeddings via pre-trained ELMO model. Subsequently, the stream of embeddings is fed to a BiLSTM network consisting of 2,048 hidden units. In the next step, we perform aggregation of the hidden states thus generated by the LSTM network via max-pooling. Finally, the resulting down-sampled output of the max-pool layer is sent to a 2-layer feed-forward network consisting of 512 units in each layer. This output is finally sent for final binary classification.

Bi-directional Conditional LSTM model 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]. In [Ghosh et al., 2017], the same idea is extended to sarcasm detection task due to the intrinsic causal nature of the relationship between the context associated with the original post and the subsequent set of responses. We also note that, in constrast to the Vanilla Bi-LSTM network, in conditional Bi-LSTM we rather use sentence level contexts rather than word level. More specifically, we prepare the sentence level embedding using average of the word embeddings, and feed the stream of context/reply sentences to their respective networks for prediction. Henceforth, we represent the context as c and the associated response as r for a given example. Furthermore, the set of sentence vectors are denoted by $\{s_{c_i}\}$ and $\{s_{r_i}\}$ respectively.

In conditional LSTM architecture, $\{s_{c_i}\}$, $\{s_{r_i}\}$ are fed to separate Bi-LSTM networks i.e. $LSTM_{context}$ and $LSTM_{response}$ respectively. And, the last output representation vector of the context stream is used to initialize the memory cell of the $LSTM_{response}$. Final prediction is performed via softmax based binary classification layer on the last output of the response network.

Bi-directional LSTM model with Attention In this model, we introduce sentence level attention layer on top of the Bi-LSTM network for the context and the responses. We incorporate the widely used dot product based Attention model into our prediction pipeline. More specifically, the attention layer for the context stream $\{s_{c_i}\}$ consisting of d sentences is described by the following equations as explained in [Yang et al., 2016]:

$$u_{c_i} = tanh(W_s h_{c_i} + b_s) \tag{1}$$

$$\alpha_i = \frac{exp(u_{c_i}^T u_{c_s})}{\sum_{j \in [d]} exp(u_{c_j}^T u_{c_s})},\ v_c = \sum_{i \in [d]} \alpha_i h_i \tag{2}$$

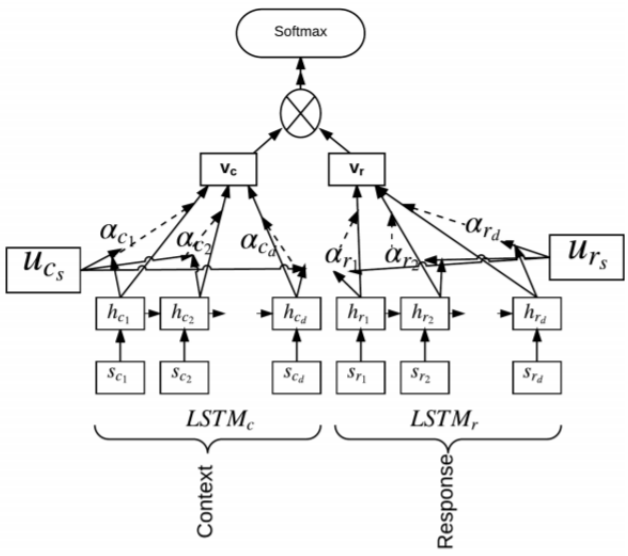


Figure 1: Sentence Level Attention based LSTM model combining context and response as proposed in [Ghosh et al., 2017]

The output of the $LSTM_{context}$ is first squased to $[-1, 1]$ as described by (1). Next, each sentence s_{c_i} is assigned weights for the final context representation vector v_c according to (2). This attention layer is particularly important in rewarding training sentences which have enough syntactic cues hinting toward degree of sarcasm. Here, u_{c_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, finally, v_c summarizes the contribution of each of sentences in the stream to the overall non-literal semantic significance of the context. We note that the exact same approach is used to generate the representation v_r for the response stream. Subsequently, we concatenate v_c and v_r and feed it to a 2-layer feedforward network for final binary clasification. We highlight that in attention based network we remove the connectivity between the $LSTM_{context}$ and $LSTM_{response}$ and introduce the attention layers. The architecture is summarized in Fig. 1.

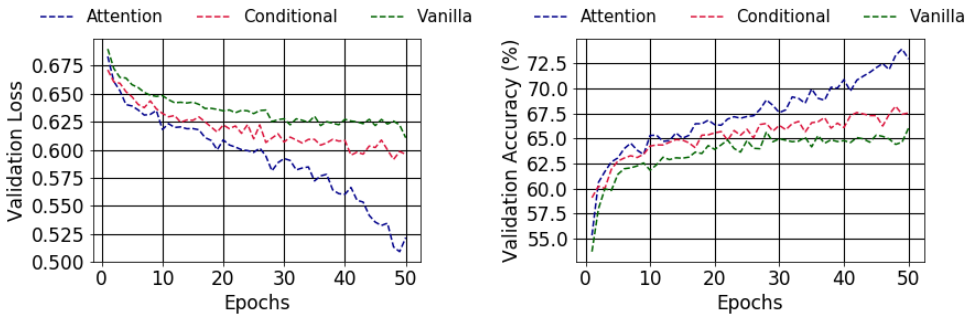
4 Experiments

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 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 In our experiments, the models are trained against cross-entropy binary classification loss. Furthermore, we conduct model comparisons using accuracies on test dataset and averaged over validation datasets in each epoch.

Hyperparameter details We determine the best set of hyperparameters by performing line search on them and using validation accuracy as metric for comparison. 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



(a) Validation Loss (b) Validation Accuracy

0.001 across all the 50 epochs. We have used batch size of 500 with Adam optimizer for mini-batch gradient descent.

4.1 Results

The performance on the test dataset for the different models is summarized below:

Model	Test Loss	Test Accuracy (%)	Test F1-Score
Vanilla Bi-LSTM	0.627	64.61	0.633
Conditional Bi-LSTM	0.617	66.9	0.646
Attention Bi-LSTM	0.601	70.55	0.682

Table 1: Performance statistics based on best validation epoch model

We first observe that performance on validation and test datasets indeed improve as model architecture is gradually improved. However, we note that Vanilla model is only slightly outperformed by the conditional Bi-LSTM. Thus, increasing model complexity via inclusion of causal depedency structure and replacing context representation with sentences rather than at word level leads to only limited gains. However, the performance gain is appreciable with the incorporation of the attention layers. Thus, it is crucial highlight that the attention model indeed brings out the inherent syntactic relationship between the individual sentences in both the context and the associated response, thereby leading to improved generalization ability and noticeable performance improvements even in the absence of any conditional connectivity in the architecture.

5 Conclusion

In this work, we explore three approaches towards identifying sarcasm by modelling the contextual information around it. We describe the Vanilla, Conditional and Attention Bi-LSTM approaches and experimentally evaluate each of the methods on the SARC dataset. In our experiments, we highlight the efficacy of the attention layers and their ability to efficiently handle the semantic connections between the sentences that comprise the context and the responses. Going ahead, it will be worthwhile demonstrating how different word embedding models such as GLoVe [Pennington et al., 2014] would perform in this setting. Moreover, how word level context/response representations behave in the presence of attention layers is another interesting future direction to explore.

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.