Final Project Report

Natural Language Processing II

CSE 440

By

Mazharul Islam Rakib (ID - 20101408, Section - 01)

Topic: Intended Sarcasm Detection In English

September 04, 2023

Submitted To: Dr. Farig Yousuf Sadeque

1 Introduction

Sarcasm is a complex form of expression that relies not only on the textual content but also on context, tone, and even historical social interactions. While it is easy for humans to understand sarcasm in many cases, machines find it significantly more challenging. The automatic detection of sarcasm in written English text has emerged as a compelling problem in the realm of Natural Language Processing (NLP). Accurate sarcasm detection can lead to more nuanced sentiment analysis, better machine-human interactions, and improved text mining techniques.

Intended Sarcasm Detection specifically focuses on identifying instances where the speaker or writer aims to convey sarcasm deliberately. This is often characterized by a disparity between the literal meaning of the text and the context in which it is placed. For example, the sentence "Oh, great, another flat tire" could be considered sarcastic if the context reveals that having a flat tire is an undesirable event for the speaker.

In this project, we aim to develop algorithms that can effectively identify intended sarcasm in English texts with remarkable accuracy. Our approach combines lexical, syntactic, and semantic features, along with contextual clues, to create a more accurate and reliable sarcasm detection model.

2 Data Collection

In a novel approach to sarcasm detection, we have curated datasets where the labels for sarcastic or non-sarcastic nature of texts are provided directly by the authors themselves. This strategy enhances the authenticity and reliability of the data. For every text labeled as sarcastic, we also procure a rephrased version from its author that conveys the same meaning but without the use of sarcasm.

In addition to author-provided labels and rephrases, each text entry is further annotated by linguistic experts. They categorize the text according to the ironic speech classifications defined by Leggitt and Gibbs (2000), including sarcasm, irony, satire, understatement, overstatement, and rhetorical questions.

Thus, each text in our datasets carries a multi-dimensional annotation:

- A label indicating whether the text is sarcastic or non-sarcastic, as provided by the author.
- A rephrased version of the text that removes the sarcastic tone but keeps the intended meaning, also provided by the author.
- A label that identifies the category of ironic speech it falls under, as determined by linguistic experts.

This richly annotated dataset serves as the foundation for developing and validating our sarcasm detection algorithms. The dataset used in this study has been sourced from a GitHub repository under the path '/Intended-Sarcasm-Detection-In-English'.

3 Model

We utilize the BERT (Bidirectional Encoder Representations from Transformers) model for sequence classification tasks, which is a highly effective model pre-trained on a large corpus of text. The specific implementation we employ is ‘BertForSequenceClassification‘, which is a version of the BERT model fine-tuned for classification tasks. The model is loaded with the pre-trained weights from the ‘’bert-base-uncased”‘ variant, which is trained on English text data and does not distinguish between uppercase and lowercase characters.

For the task at hand, we configure the model to have two output labels, corresponding to the binary classification of text as either sarcastic or non-sarcastic. The configuration is specified as follows:

```
model = BertForSequenceClassification.from_pretrained
('bert-base-uncased', num_labels=2)
```

For optimization, we employ the Adam optimizer, which is known for its effectiveness in training deep neural network models. We set the learning rate to 1×10^{-5}, a value that is commonly used for fine-tuning BERT models on specific tasks. The optimizer is initialized as follows:

```
optimizer = torch.optim.Adam(model.parameters(), lr=1e-5)
```

The choice of Adam and the specified learning rate aims to strike a balance between training speed and model performance, allowing for efficient fine-tuning of the pre-trained BERT model on our specific task of intended sarcasm detection.

4 Implementation

The implementation uses Python with essential libraries such as pandas, Transformers, PyTorch, and scikit-learn for the tasks of data loading, tokenization, model training, and evaluation. The code is run on Google Colaboratory.

4.1 Data Importing and Preprocessing

The training data is loaded from a CSV file and contains columns for the tweet text and sarcasm labels. Similarly, the test data is read from another CSV file.

- The text from the training data is tokenized using the BERT tokenizer.
- Input IDs are padded to a maximum sequence length.
- Attention masks are created based on the input IDs.

4.2 Model and DataLoader

- A BERT sequence classification model is initialized with two output labels.
- The DataLoader constructs mini-batches from the training and validation datasets.

4.3 Training

The model is trained using the Adam optimizer with a learning rate of 1×10^{-5}.

- The model is trained for 3 epochs.
- For each batch, the optimizer's gradients are zeroed, the forward pass is computed, and the backpropagation is performed based on the loss.

4.4 Testing and Evaluation

- The test data is also tokenized and converted to DataLoader format.
- The model's performance is evaluated based on its accuracy, precision, and recall on the test dataset.
- A confusion matrix is also calculated for further analysis.

The implementation efficiently combines various Python libraries to train and evaluate a sarcasm detection model based on the BERT architecture.

5 Analysis and Result

In this study, we evaluated the performance of the BERT-based sarcasm detection model on two different tasks, Task A and Task B.

5.1 Accuracy

For Task A, the model achieved an accuracy of 86.7%, which indicates a fairly high capability in distinguishing sarcastic comments from non-sarcastic ones in the given dataset. In comparison, the model's accuracy for Task B was 85.6%, also showcasing a strong performance.

5.2 Precision and Recall

The model achieved a precision of 0.1632 and a recall of 0.2611 in Task B. Though the precision is relatively low, it means that among the samples that the model predicted as sarcastic, only 16.32% were actually sarcastic. The recall value suggests that the model correctly identified 26.11% of all actual sarcastic comments in the dataset.

5.3 Confusion Matrix

The confusion matrix for Task A and B is as follows:

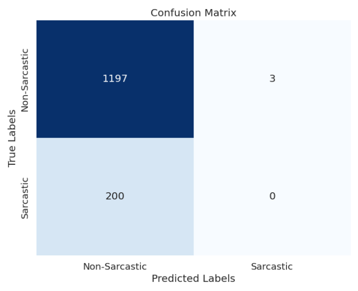

Figure 1: Confusion Matrix for Task A

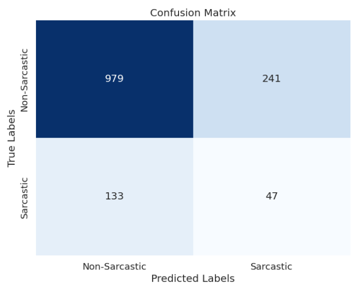

Figure 2: Confusion Matrix for Task B

Here, the true negatives are 1197 and 979, false positives are 3 and 241, false negatives are 200 and 133, and true positives are 0 and 47. The higher number of true negatives suggests that the model is more cautious in predicting sarcasm, which is also reflected in its precision and recall values.

6 Limitations

In our analysis, we observed several limitations that should be addressed for future work:

1. **Low Precision and Recall**: The model exhibits low precision and recall, which suggests a substantial number of false positives and false negatives. High precision or recall is crucial for applications that cannot afford these errors.
2. **Imbalanced Data**: The dataset used might be imbalanced with more non-sarcastic examples compared to sarcastic ones, potentially leading to a biased model.
3. **Limited Context**: Text-based data may not capture the full context in which sarcasm occurs, such as tone of voice or situational context, affecting the model's performance.
4. **Training Time**: The use of BERT models can be computationally expensive and time-consuming, particularly for large datasets.
5. **Generalization**: The model is trained on tweets, making it uncertain how well it would generalize to other forms of text.
6. **Computational Requirements**: The BERT architecture has high computational and memory requirements, making it less suitable for resource-constrained environments.

7. **Language and Cultural Bias**: If the training data is skewed towards a specific demographic, cultural or linguistic bias could be a concern.

7 Conclusion

Both the accuracy metrics and the confusion matrix provide valuable insights into the model's performance. While the model performs well in terms of accuracy, there is room for improvement in its precision and recall. Further tuning and feature engineering may help to address these aspects.

8 References

1. iSarcasmEval GitHub Repository. Available at: `https://github.com/iabufarha/iSarcasmEval`

2. Sarcasm Detection Using Bidirectional Encoder Representations from Transformers and Graph Convolutional Networks. Available at: `https://www.sciencedirect.com/science/article/pii/S1877050922024991`