CMPSC 497 Midterm Project – Sarcasm Detection with Deep Learning

Chih-Hsuan Sun

I. Problem Definition and Dataset Curation

Problem Definition

Sarcasm detection in textual data is a complex variant of semantic classification - what we have done in class before - and a challenging Natural Language Processing (NLP) problem, as it often relies on implicit meaning, tone, and context. In this project, I try to classify tweets into four categories: figurative, irony, sarcasm, and regular. This classification can help understand user sentiment, improving chatbot responses as well as content moderation.

Dataset Curation

The dataset used in this project contains tweets annotated into **four classes**: **Figurative, Irony, Sarcasm,** and **Regular.**

The dataset I chose is "Tweets with Sarcasm and Irony". The training set (train.csv) is further split into **training (80%) set** and **validation (20%) set**, and the model is tested on the separate **test set** (test.csv). The dataset is preprocessed to remove stopwords, punctuation, URLs, and unnecessary symbols, ensuring cleaner input for the model. The final dataset distributions are:

- **Training Set**: 81,408 samples
- **Validation Set**: 20,352 samples
- **Test Set**: 8,119 samples

Class distributions in the dataset are presented in Table 1.

Training set		**Test set**	
Class/Label	**Percentage**	**Class/Label**	**Percentage**
figurative	29.15%	figurative	25.18%
regular	25.52%	regular	22.9%
irony	28.68%	irony	26%
sarcasm	16.64%	sarcasm	25.93%

Table 1: Distribution across each labels in train and test sets.

II. Word Embeddings, Algorithm, and Training Details

Word Embeddings

Two types of pre-trained word embeddings, as well as self-trained embeddings were explored:

1. **Self-trained (128D):** Randomly initialized embeddings in the first layer.
2. **Word2Vec (Google News 300D)**: Extracted from the word2vec-google-news-300 model.
3. **GloVe (50D)**: Extracted from the glove-wiki-gigaword-50 model.

Algorithm and Model Architecture

The model used for results is a stacked LSTM network (built with Tensorflow) with L2 regularization, optimized for sarcasm detection. The architecture consists of:

1. **Embedding Layer:** Maps words to a 50-dimensional vector space (trainable).
2. **Bidirectional LSTM (128 units):** A BiLSTM layer with L2 dropout (0.5).
3. **Batch Normalization:** Stabilizes training by normalizing activations across batches.
4. **Fully Connected Layer (32 units, ReLU activation):** Extracts high-level features.
5. **Dropout (0.5):** Helps prevent overfitting.
6. **Output Layer (4 classes, Softmax activation):** Predicts one of the four integer encoded categories (figurative, irony, sarcasm, regular).

The printed parameters and the overall architecture are shown in Figure 1.

Model: "sequential_14"

Layer (type)	Output Shape	Param #
embedding_14 (Embedding)	(None, 34, 50)	250,000
bidirectional_9 (Bidirectional)	(None, 256)	183,296
batch_normalization_9 (BatchNormalization)	(None, 256)	1,024
dense_28 (Dense)	(None, 32)	8,224
dropout_14 (Dropout)	(None, 32)	0
dense_29 (Dense)	(None, 4)	132

Total params: 442,676 (1.69 MB)
Trainable params: 442,164 (1.69 MB)
Non-trainable params: 512 (2.00 KB)

Figure 1: Architecture and parameters of the BiLSTM based model

Other models, including regular LSTM model, multi-layer LSTM as well as CNN based models are also tested. However, since I did not see significant improvements in the performance comparing to the presented model, I will only include the results from the presented model (with GloVe embeddings).

Training Details

- **Loss Function**: Sparse Categorical Cross-entropy.
    - Since the classification problem has four mutually exclusive classes (figurative, irony, sarcasm, regular), and the labels are integers but not one-hot encoded, Sparse Categorical Cross-entropy should be appropriate here.
- **Optimizer**: Adam (Learning Rate: 0.001, Clip Value: 1.0)
- **Batch Size**: 32
- **Epochs**: 15
    - 15 epochs may seem low, but since my training stabilizes shortly within 15 epochs, this should be enough.
- **Early Stopping:** Stops after val performance degrades (Patience:3).
- **Hardware:** The training was done with T-4 GPU runtime on Google Colab.

III. Results and Presentation

Model Performance

The model was trained and evaluated using accuracy and loss metrics:

Dataset	Accuracy	Loss
Training Set	62.32%	0.776
Validation Set	59.39%	0.832
Test Set	60.19%	0.8124

Table 2: Classification performance across each set with the BiLSTM model and GloVe embeddings.

Accuracy and Loss Trends

Training and validation accuracy/loss curves are plotted as in Figure 2 - 4, with self-trained embeddings, wor2vec, and GloVe, respectively. Figure 4 shows the best results I have so far (stabilized and smooth training curves with highest val accuracy).

The results indicate that although the training curves look normal across all embeddings, only with GloVe can the model reach a stable performance within 15 epochs. While 60% may not appear high, this is the best accuracy I can achieve, and I believe it should be feasible given the difficulty of the problem and the dataset. With the other two embeddings, however, the model seems to overfit, leading to unstable or degrading validation performance.

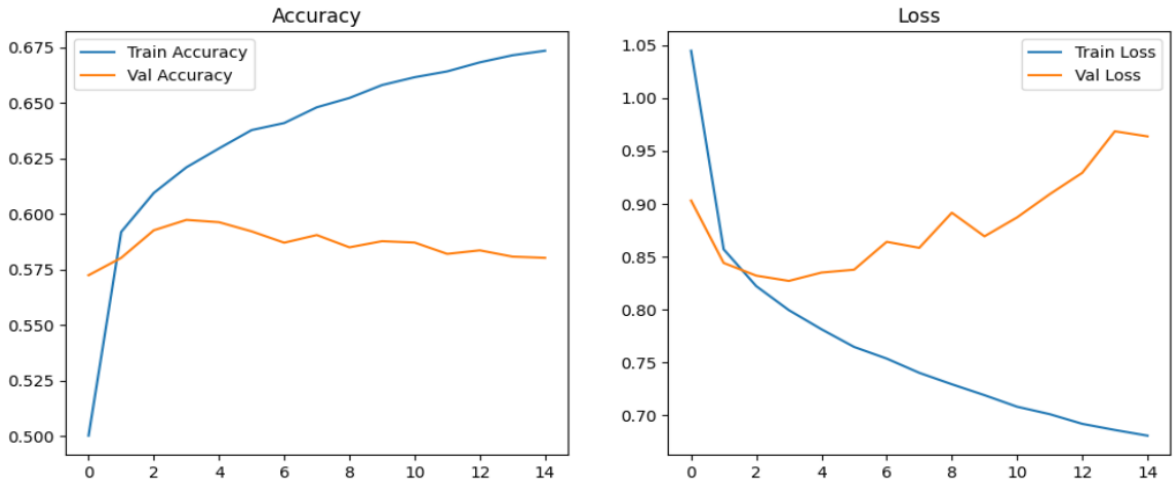

Figure 2: Learning curves with self-trained embeddings on sarcasm detection.

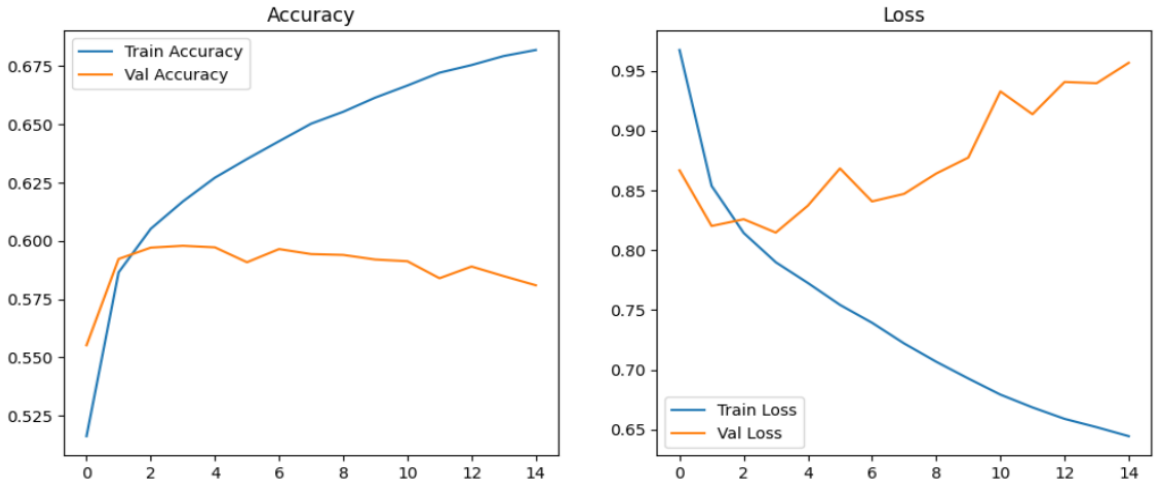

Figure 3: Learning curves with Word2Vec embeddings on sarcasm detection.

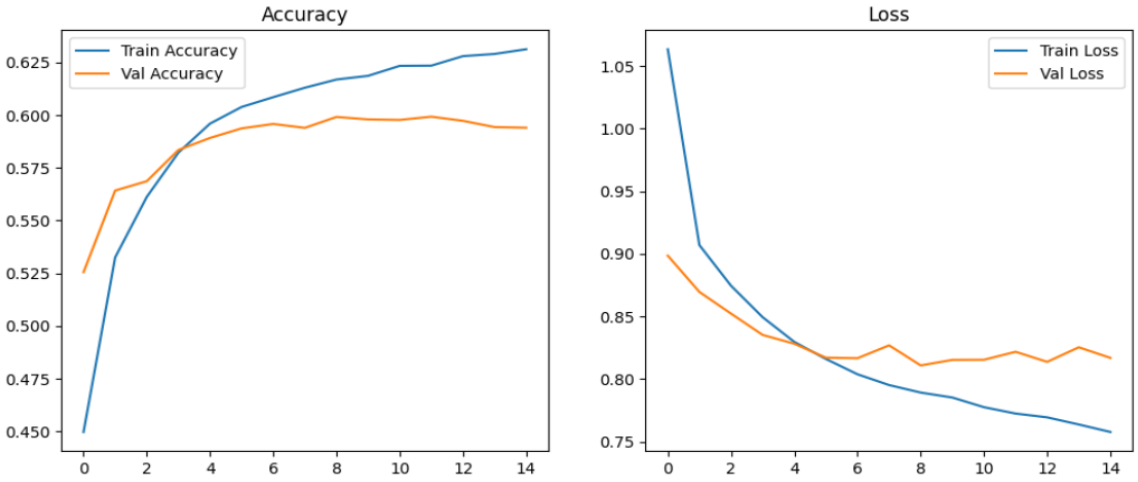

Figure 4: Learning curves with GloVe embeddings on sarcasm detection.

Example Predictions

Sample predictions from the test set are included in Table 2, randomly chosen for demonstration.

Tweet	True Label	Predicted Label
michael clarke moves himself down the order to keep away from the new ball comes in after 8 deliveries #ashes2015 #bbccricket	Irony	Irony
fact checking a hillary clinton email claim #christian #news	Regular	Regular
photo fiasco dnc mistakes elderly poles for us vets #politics	Regular	Regular
thats the spirit!	Sarcasm	Sarcasm
trump says he would get along with mexico? #gop2016	Figurative	Irony
maybe they can hire john harbaugh next year	Figurative	Sarcasm
i dunno though didn't he work for the dhss and we all know everyone on benefits has got more money than soft mick	Figurative	Sarcasm
host wordpress #wordpress #education #inspirational #lol #funny	Regular	Regular
nice try with the cheap shot #unreliablewebsite #kys	Sarcasm	Sarcasm

Table 2: Prediction examples from the test set.

The model performs well in detecting sarcasm but occasionally confuses figurative and irony/sarcasm classes.

IV. In-Depth Analysis and Experiments

Experiments Conducted

a) **Comparison of Embeddings**:

- GloVe 50D provided stable and effective performance with smooth training curves.
- Word2Vec 300D failed to stabilize the validation performance throughout the training, with no obvious accuracy improvement over GloVe, suggesting overfitting problems. This could be a result of the high dimensionality.
- Self-trained embeddings (128D) also failed to stabilize, which could be due to the sarcasm dataset not able to provide enough meaningful samples to learn semantic embeddings, making it biased towards the training set.

b) **Impact of LSTM Layers**:

- More layers are not necessarily better, as it produced practically the same results.

- Bidirectional LSTM can help stabilize the training, but no significant improvement on the accuracy.
- Increasing the LSTM size can speed up the convergence.

c) **Dropout and Regularization**:

- L2 regularization can sometimes further destabilize the performance.
- Including recurrent dropout on LSTM will significantly slow down training due to its effect on GPU efficiency and neuron masking.

d) **Misclassification Analysis**:

- Figurative tweets were frequently misclassified, often as irony or sarcasm, likely due to overlapping linguistic patterns and lack of explicit markers. This could imply there is potential ambiguity in label definitions, as figurative expressions sometimes share common characteristics with irony and sarcasm.
- The model may rely too much on vocabulary patterns rather than deeper contextual meaning, leading to misclassification.
- Despite having a supposedly sufficient number (29%) of figurative examples in the dataset, the model struggles to differentiate them. We may need better contextual understanding.
- Some low-confidence correct predictions may have contributed to the higher test loss, despite the improved accuracy comparing to validation.

V. Lessons & Experience Learned

There were some challenges I encountered in the project. The major one being preprocessing data. The original dataset is very noisy with lots of URLs and unintelligible words and symbols, causing the model to perform poorly. On the other hand, after further inspection, I found that the data (tweets) itself contains hashtags such as #sarcasm or #irony, which is a big red flag as it may cause data leakage, and due to this discovery I had to re-work on my whole experiment. It made me realize the importance of text cleaning and data preparation in NLP based preprocessing.

I have listed a few key takeaways from the project as follows.

Key Takeaways

- **Data Preprocessing Matters**: Removing stopwords, URLs, and unnecessary characters was a tedious process but significantly improved model performance. The data itself should also be closely inspected to ensure the integrity of the experiment.
- **Hyperparameter Tuning is Crucial**: Small adjustments in LSTM size, dropout, L2 regularization, and optimizer settings can also affect performance as well as computation efficiency. We should choose them carefully.

- **Training is costly**: Even with Colab Pro subscription, training the model for this project costed me a few dollars. I will have to think about looking for other GPU resource options now.
- **Sarcasm is Context-Dependent**: Some misclassifications occurred due to missing context that the model couldn't infer from short tweets.
- **More Training Data Could Help**: The dataset's sarcasm examples were limited (I realize tweets may not be the best source to classify sarcasm), which may have constrained generalization.

Conclusion

This project successfully implemented a deep learning approach for sarcasm detection in tweets. The BiLSTM model with GloVe embeddings achieved 60% accuracy on the test set. The dataset I worked on may be more challenging than others since tweets are often very noisy and difficult to interpret even for normal humans, and I took a step further and removed hashtags that directly match the keywords in the labels, preventing data leakage while also increasing the difficulty. There are potential further improvements, such as using transformers (e.g. BERT) and contextual attention mechanisms, to be considered to enhance sarcasm detection accuracy.