Group No - 10
Group members:
Md. Tawsifur Rahman - 20141027
Siam Sadman Azad - 20141002
Sayma Akter Lubna - 20301450
Md Mahbub Alam Prithibi - 20101243

Leveraging GloVe Embeddings and LSTM Networks for Sarcasm Detection in News Headlines

Abstract:

Sarcasm detection in textual data remains a challenging yet pivotal task in natural language processing. This paper presents an approach utilizing GloVe word embeddings and LSTM (Long Short-Term Memory) networks to detect sarcasm in news headlines. Leveraging a dataset curated from news sources, we explore the effectiveness of pre-trained GloVe embeddings to represent the semantic and contextual information within headlines. The LSTM architecture, known for its ability to capture sequential dependencies, is employed to discern the intricate linguistic nuances associated with sarcastic expressions. The study involves preprocessing the headline dataset, integrating GloVe embeddings into an LSTM-based model, and evaluating its performance in distinguishing sarcastic from non-sarcastic headlines. Experimental results showcase the efficacy of the proposed methodology, demonstrating promising accuracy and providing insights into the robustness of employing deep learning techniques for sarcasm detection in news headlines. The findings underscore the potential of leveraging sophisticated neural architectures and pre-trained embeddings in deciphering the subtle linguistic cues embedded within textual data, particularly in the realm of news sarcasm detection.

Introduction:

In the era of vast digital content, the detection of sarcasm in textual data stands as a crucial yet intricate challenge in natural language understanding. With the proliferation of news headlines across online platforms, the discernment of sarcastic intent within these succinct and often nuanced expressions has garnered significant attention. Sarcasm, characterized by an incongruity between literal and intended meanings, presents a formidable hurdle in automated

sentiment analysis and textual comprehension due to its subtlety and context-dependent nature.

This paper delves into the realm of sarcasm detection specifically within news headlines, a domain replete with linguistic intricacies and diverse contextual cues. Leveraging advancements in deep learning methodologies, particularly GloVe (Global Vectors for Word Representation) embeddings and LSTM (Long Short-Term Memory) neural networks, we aim to unravel the subtleties of sarcastic language embedded within news headlines.

The use of pre-trained word embeddings, such as GloVe, enables the conversion of textual data into continuous vector representations that encapsulate semantic and contextual information. Complementing this, LSTM networks, renowned for their capability to capture sequential dependencies, offer a promising avenue to discern the implicit sarcasm embedded within headlines by comprehending the underlying linguistic structures.

This study seeks to address the scarcity of research dedicated to sarcasm detection within the news domain while exploring the efficacy of state-of-the-art deep learning techniques. Through an empirical investigation involving the curation and preprocessing of a dataset comprising news headlines, the integration of GloVe embeddings, and the design of an LSTM-based sarcasm detection model, we aim to elucidate the effectiveness of these methodologies in accurately identifying and distinguishing sarcastic from non-sarcastic headlines.

The ensuing sections delve into the methodologies employed, presenting the experimental setup, results, and discussions that underpin the effectiveness and implications of leveraging GloVe embeddings and LSTM networks for the challenging task of sarcasm detection within news headlines.

Literature review

In Abaskohi et al [2022], the authors explored various models, including SVM-based, BERT-based, Attention-based, and LSTM-based, along with different data augmentation strategies such as generative-based and mutation-based approaches. They attempted to expand the training set by augmenting the iSarcasm dataset with Sentiment140 (Go et al. [2009]) and Sarcasm Headlines Dataset (Misra

and Arora [2019]), but notably did not augment with SARC, our chosen dataset.

The authors found that fine-tuning RoBERTa for sentiment classification, without data augmentation and further fine-tuning only on the iSarcasm dataset, achieved the best performance with an F1 score of 0.414. However, this fell short of the RoBERTa model fine-tuned for sarcasm detection by Hercog et al. [2022]{b2}, which utilized a training set comprising iSarcasm and a subset of SARC, achieving a higher F1 score of 0.526 on iSarcasm.

The iSarcasm dataset (Oprea and Magdy [2019]) is unique as it consists of tweets labeled by the authors themselves as sarcastic or non-sarcastic, focusing on intended sarcasm. This labeling approach differs from datasets that label perceived sarcasm, and models trained on such datasets performed poorly on iSarcasm. The dataset comprises 777 sarcastic and 3,707 non-sarcastic tweets.

The Self-Annotated Reddit Corpus (Khodak et al. [2017]) is a dataset of Reddit comments labeled for perceived sarcasm, using the "/s" token convention. However, SARC is noisy due to false negatives (sarcastic comments without "/s") and false positives (non-sarcastic comments with "/s"). We use a balanced subset of SARC as our starting point, attempting to prune it while maintaining similar performance. The pruning process involves adopting the student-teacher setting for perceptron learning described by Sorscher et al. [2022].

Sorscher et al. [2022] explored ways to select training samples to achieve a reduction in test error better than scaling laws. They showed that optimal pruning strategies depend on the amount of initial data, and exponential scaling is possible with respect to pruned dataset size by choosing an increasing Pareto optimal pruning fraction based on the initial dataset size. These results apply not only to training from scratch but also to fine-tuning, which aligns with our setting.

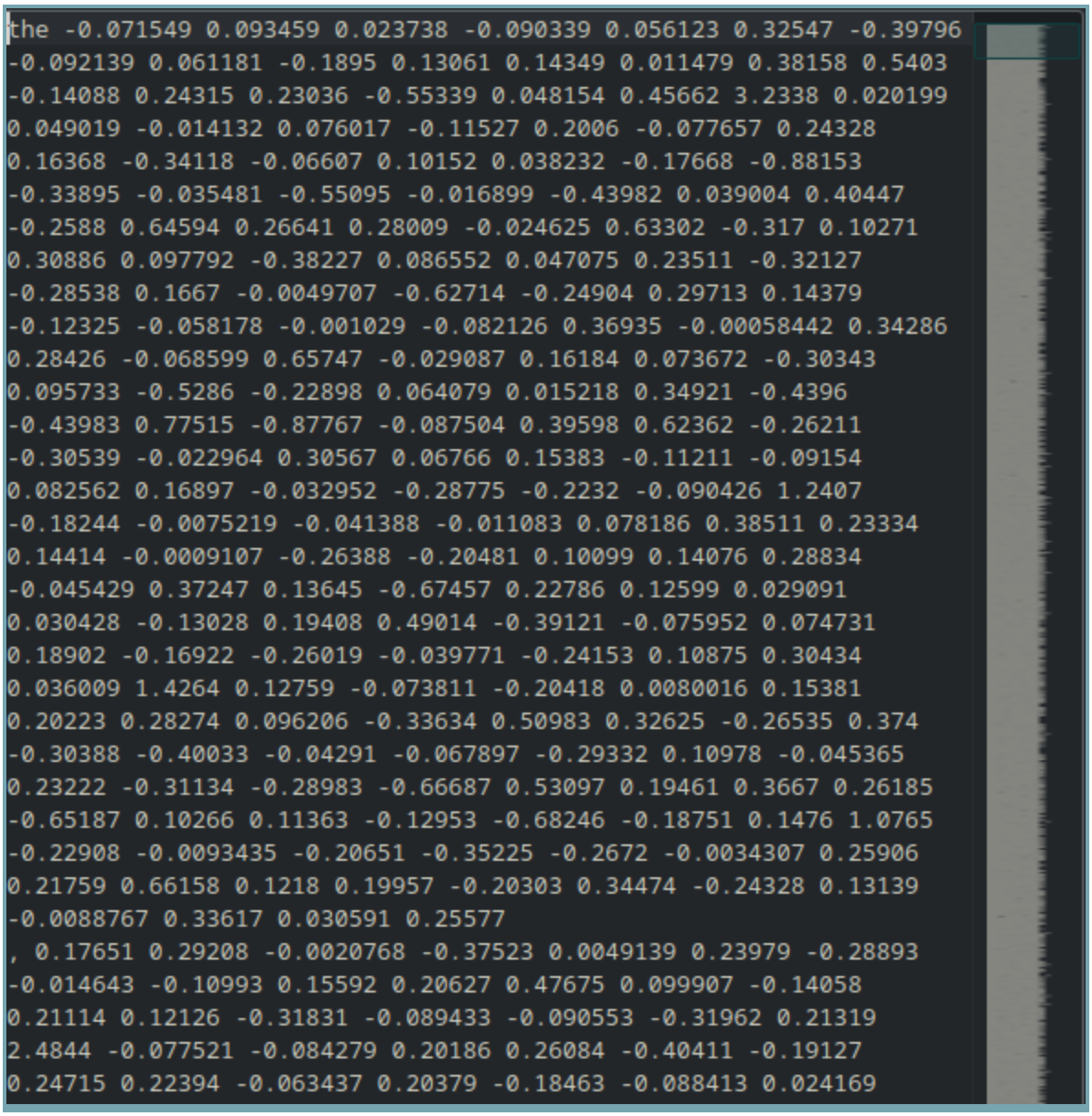
Abaskohi et al. [2022] used Cross Entropy Loss, and to address class imbalance in the iSarcasm dataset, we consider Weighted Cross Entropy Loss. Squared Hinge Loss, known for good performance in classification tasks, is also explored. Janocha and Czarnecki [2017] compare various losses and recommend using squared hinge loss when operating in a data-scarce regime, as it converges faster and exhibits more robustness to noise in training set labeling and input space. This robustness is particularly beneficial when augmenting

with SARC, a noisier dataset than iSarcasm labeled using distant supervision.

GloVe Embeddings

Word Embeddings:
Word embeddings represent words as dense vectors, where similar words are closer to each other in the embedding space. They are learned from large text corpora using various techniques like Word2Vec, GloVe, or FastText. For instance, in a well-trained word embedding space, words like "king" and "queen" might be closer in distance compared to words like "dog" or "cat" because they share similar contexts and relationships.

Loading GloVe Embeddings File: glove.6B.200d.txt




```
EMBEDDING_FILE = '/content/drive/MyDrive/sarcasm
detection/glove.6B.200d.txt'

embeddings = {}
for o in open(EMBEDDING_FILE):
    word = o.split(" ")[0]
    # print(word)
    embd = o.split(" ")[1:]
    embd = np.asarray(embd, dtype='float32')
    # print(embd)
    embeddings[word] = embd
```

```
# create a weight matrix for words in training docs
embedding_matrix = np.zeros((num_words, 200))

for word, i in tokenizer.word_index.items():
      embedding_vector = embeddings.get(word)
      if embedding_vector is not None:
            embedding_matrix[i] = embedding_vector
```

Reading Embeddings:
The code initializes an empty dictionary embeddings to store word embeddings.
It reads each line of the GloVe file, splits it by spaces, and extracts the word as well as its embedding vector.
The word is stored as a key in the embeddings dictionary, and its embedding vector (as an array of floats) is stored as the value.
Creating Embedding Matrix:

It initializes an embedding matrix embedding_matrix with zeros.
Then, for each word in the word index generated by tokenizer.word_index.items(), it checks if the word exists in the pre-trained GloVe embeddings.
If the word is found, its pre-trained embedding vector is retrieved from the embeddings dictionary and placed into the embedding_matrix at the corresponding index obtained from tokenizer.word_index.
This code essentially loads pre-trained word embeddings from a file, matches them with the words in your dataset using a tokenizer, and creates an embedding matrix that can be used in a neural network or machine learning model for tasks like sentiment analysis, text classification, etc.

Remember, the success of using these pre-trained embeddings in a model depends on various factors such as the similarity between the vocabulary of your dataset and the pre-trained embeddings, as well as the specific task you're working on.

Model:

```
lstm_out = 196

# define the model
model = Sequential()
model.add(Embedding(num_words,
```

```
                    embedding_size,
                    embeddings_initializer=Constant(embedding_matrix),
                    input_length=maxlen,
                    trainable=False))
model.add(SpatialDropout1D(0.4))
model.add(Bidirectional(LSTM(lstm_out, dropout=0.2,
recurrent_dropout=0.2)))
model.add(Flatten())
model.add(Dense(1, activation='sigmoid'))
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 25, 200)           5531400

 spatial_dropout1d (Spatial  (None, 25, 200)           0
 Dropout1D)

 bidirectional (Bidirection  (None, 392)               622496
 al)

 flatten (Flatten)           (None, 392)               0

 dense (Dense)               (None, 1)                 393

=================================================================
Total params: 6154289 (23.48 MB)
Trainable params: 622889 (2.38 MB)
Non-trainable params: 5531400 (21.10 MB)
_________________________________________________________________
None
```

Embedding Layer:

The model starts with an Embedding layer. This layer converts input sequences into dense vectors of fixed size (embedding_size).
It uses pre-trained word embeddings loaded into the embedding_matrix.
The parameter num_words specifies the size of the vocabulary.
trainable=False means the embeddings (weights) loaded from GloVe will remain fixed and not be updated during training.

Spatial Dropout:

SpatialDropout1D is a type of dropout that helps prevent overfitting in neural networks by randomly setting a fraction of input units to zero.
0.4 represents the dropout rate, where 40% of the input units will be randomly dropped during training.

Bidirectional LSTM (Long Short-Term Memory):

LSTM is a type of recurrent neural network (RNN) designed to capture long-term dependencies in sequential data.
Bidirectional indicates that the LSTM processes the input sequence both forwards and backwards, capturing information from both past and future states of the sequence.
lstm_out defines the dimensionality of the output space of the LSTM layer.

Flatten Layer:

This layer flattens the output from the Bidirectional LSTM into a 1D array.
It prepares the output for the subsequent Dense layer.

Dense Layer:

The model ends with a Dense layer with a single unit and a sigmoid activation function.
It produces a binary classification output (0 or 1) for the binary classification task (e.g., sentiment analysis, binary categorization).
The sigmoid activation function squashes the output between 0 and 1, interpreting it as a probability.

Summary:

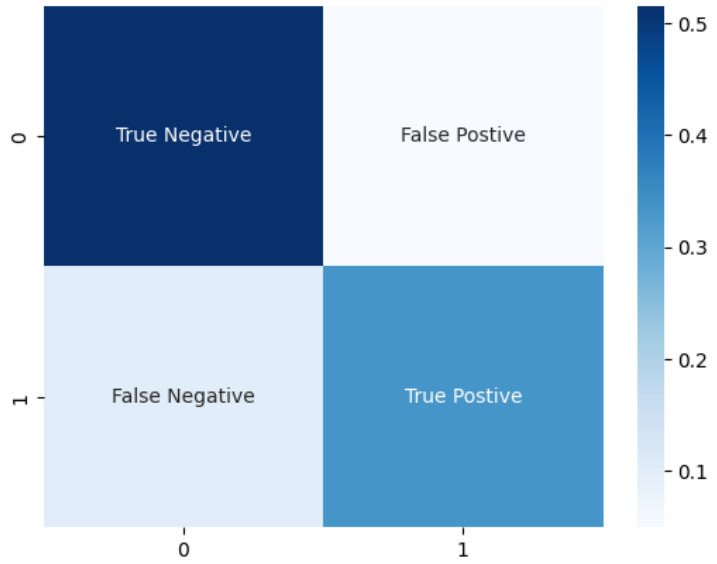
The model architecture consists of an Embedding layer with pre-trained GloVe embeddings, followed by a Spatial Dropout layer to prevent overfitting. Then, a Bidirectional LSTM layer captures contextual information bidirectionally, and a Flatten layer reshapes the output for the final Dense layer, which produces a binary classification output using a sigmoid activation function. This type of architecture is commonly used in NLP tasks for sequence classification, especially when dealing with text data.

Results:

BernoulliNB:

Classification Report:

category	precision	recall	f1-score	support
0	0.84	0.91	0.87	3017
1	0.87	0.77	0.82	2325
accuracy			0.85	5342
macro avg	0.85	0.84	0.84	5342
weighted avg	0.85	0.85	0.85	5342

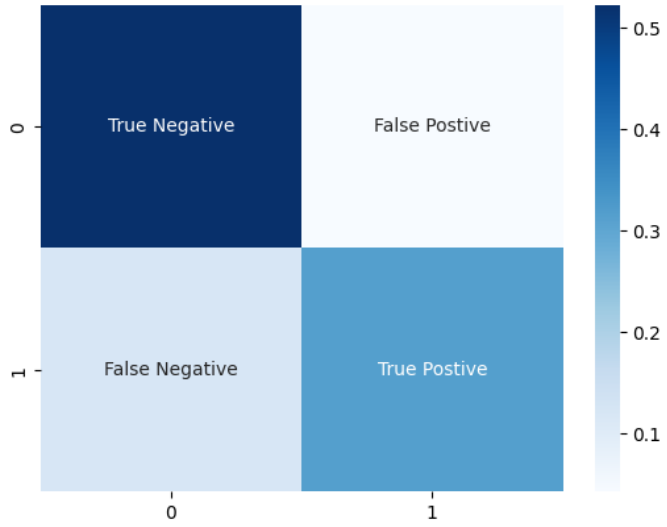


MultinomialNB

Accuracy Score:0.8371

Classification Report:

category	precision	recall	f1-score	support
0	0.81	0.93	0.87	3017
1	0.88	0.72	0.79	2325
accuracy	0.84	5342		
macro avg	0.85	0.82	0.83	5342
weighted avg	0.84	0.84	0.83	5342

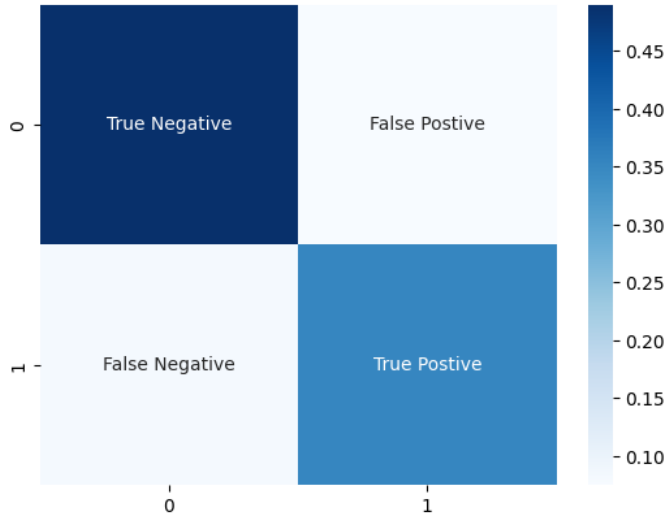


Logistic Regression:

Accuracy Score: 0.842755522276301

Classification report:

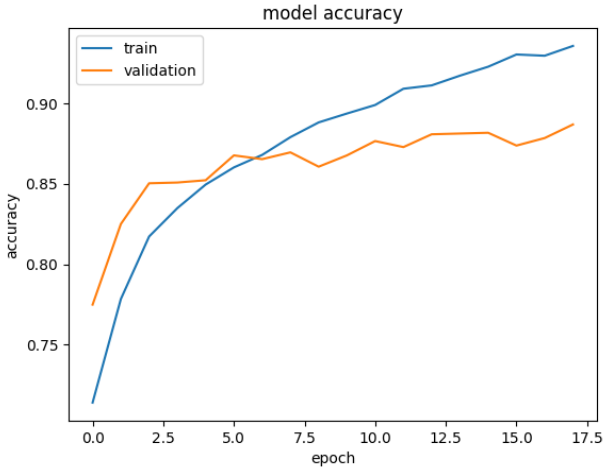
category	precision	recall	f1-score	support
0	0.86	0.87	0.86	3017
1	0.82	0.81	0.82	2325
accuracy	0.84	5342		
macro avg	0.84	0.84	0.84	5342
weighted avg	0.84	0.84	0.84	5342

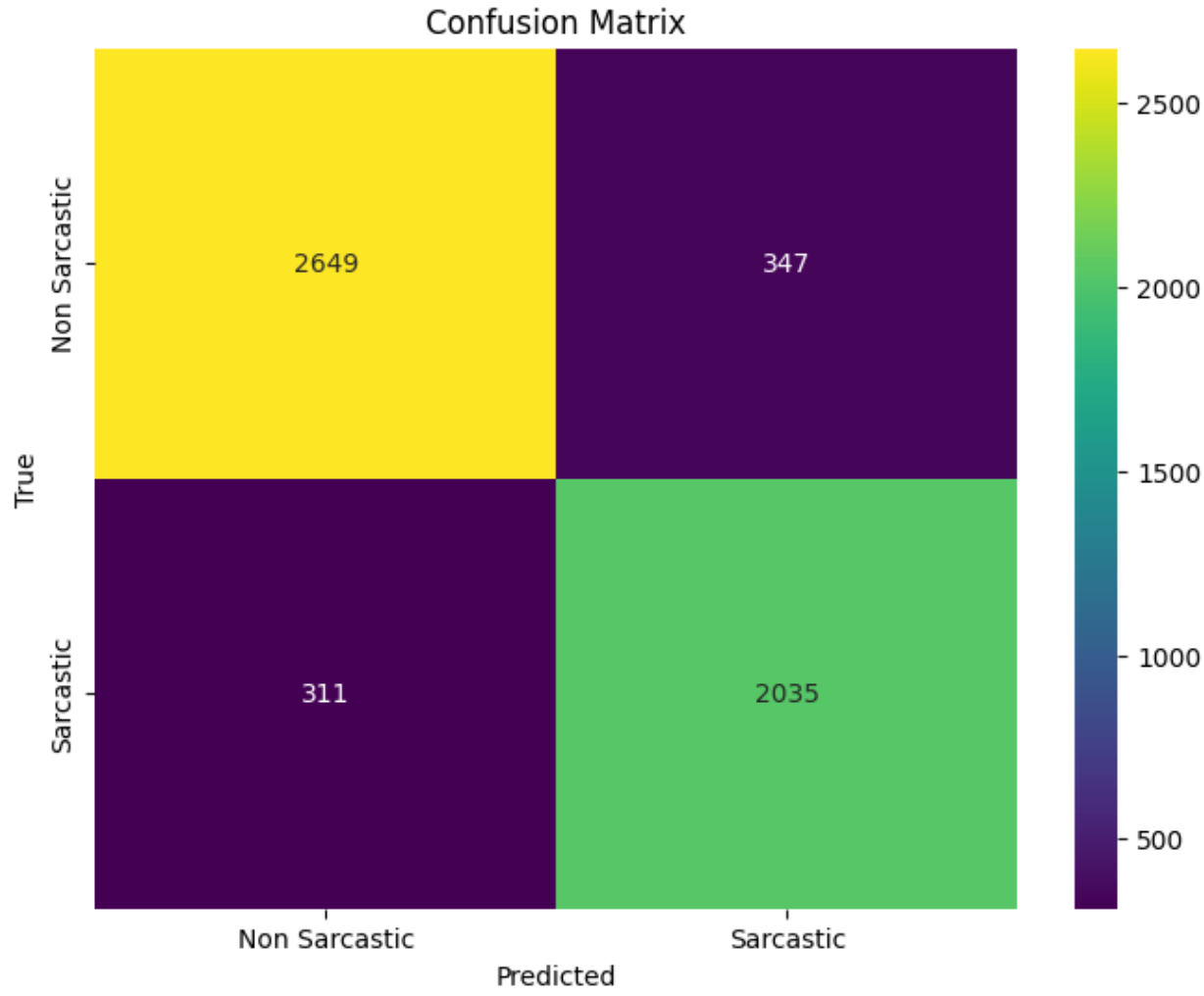


LSTM:

Test accuracy: 0.8768

Test loss: 0.3057





Limitations:

Despite the promising results and advancements made in sarcasm detection within news headlines using GloVe embeddings and LSTM networks, several limitations merit attention. Firstly, the reliance on pre-trained embeddings, while beneficial in capturing semantic relationships, may not encapsulate domain-specific nuances present in news headlines, potentially limiting the model's adaptability to specialized contexts.

Additionally, the inherent subjectivity and context-dependency of sarcasm pose challenges, as the same phrase might exhibit sarcastic intent in one context while being literal in another. The model's performance might be affected by such variations in contextual understanding.

Moreover, the dataset used in this study, while carefully curated, might lack comprehensiveness across diverse news sources, potentially introducing biases or overlooking certain linguistic patterns prevalent in specific domains or regions.

Future Work:

To address these limitations and further enhance the efficacy of sarcasm detection in news headlines, future research avenues present intriguing opportunities. One potential direction involves domain-specific fine-tuning of embeddings or utilizing transfer learning techniques to adapt pre-trained models to the specific linguistic nuances inherent in news headlines.

Exploring multimodal approaches that combine textual data with additional contextual cues, such as metadata, images, or user interactions with headlines, could offer a more comprehensive understanding of sarcastic intent.

Additionally, employing advanced techniques in contextual embeddings or transformer-based models might improve the model's ability to capture intricate linguistic nuances and contextual variations more effectively.

Conclusion:

In conclusion, this study has demonstrated the viability of employing GloVe embeddings and LSTM networks for sarcasm detection within news headlines. Through the integration of pre-trained word embeddings and the utilization of LSTM architectures, our model showcased promising performance in discerning the nuanced sarcastic expressions present in news headlines.

While acknowledging the limitations inherent in the current approach, the findings underscore the potential of leveraging deep learning methodologies for sarcasm detection in textual data. The study contributes to the evolving landscape of natural language understanding by shedding light on the complexities associated with sarcasm detection, particularly within the domain of news headlines.

As we venture into more nuanced and sophisticated techniques, the pursuit of improving sarcasm detection remains a dynamic and promising area, promising enhanced comprehension of language nuances and contributing significantly to advancing natural language processing applications in diverse domains.