Fake News Detection

Aaron Feng, Angelina Zhang, Sam Qiu, Peter Liu
Halıcıoğlu Data Science Institute, Department of Mathematics,
Department of Computer Science, University of California, San Diego (UCSD)
{zhf004, ruz039, ziqiu, pel006}@ucsd.edu
GitHub Link

December 4, 2024

1 Introduction

In this project, we explore fake news detection using text-based classification methods, aiming to understand how machine learning models can effectively identify fraudulent news articles. We began by experimenting with baseline models using Term Frequency-Inverse Document Frequency (TF-IDF), a method covered in class that calculates word frequencies and weights them by importance. For our final model, we implemented Convolutional Long Short-Term Memory (CLSTM), a widely recognized deep learning architecture known for its strong performance on sequential data like text.

Our primary objectives were to compare the performance of baseline and advanced models, investigate whether the CLSTM model captures features similar to those highlighted by TF-IDF, and study the effect of sample size on model performance. We also sought to evaluate the generalizability of these models on datasets with varying levels of variation. This work not only highlights the strengths of frequency-based methods in text classification but also identifies potential areas for improving model transparency and generalization to unseen data.

2 Dataset

The dataset used for this study is the `fake_news` dataset from the Hugging Face platform. It consists of a total of 24.4k rows in the training set, while both the test and validation sets contain 8.12k rows each. Each row in the dataset is structured into four columns, where the first column serves as an index, and the remaining three columns provide the following information:

- **Title:** This column contains the headline of the news article. Titles are typically concise and summarize the main idea of the news piece, providing a quick overview of the content.
- **Text:** This column includes the body or content of the news article. It offers additional context and details to complement the title, ranging from brief descriptions to extensive narratives.
- **Label:** This column provides the classification of the news article. A binary label indicates whether the news is fake (**0**) or real (**1**). These labels are crucial for training and evaluating the fake news detection models.

We first did exploratory data analysis (EDA) to see if there is a relationship between the length of the content and label in our training dataset. We found that fake news in training datasets are more likely to have a longer content length.

One of the interesting things we found is that there were some censored words in the

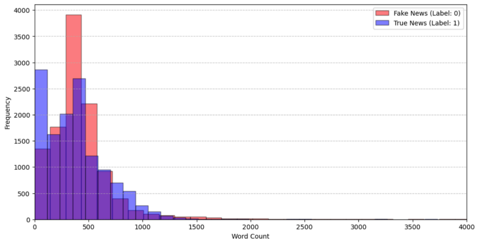
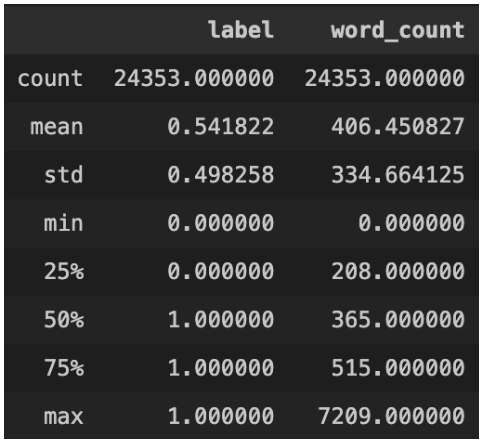


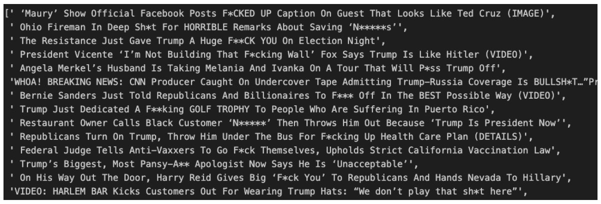
Figure 1: Overlaid Histogram of Word Count by News Label



	label	word_count
count	24353.000000	24353.000000
mean	0.541822	406.450827
std	0.498258	334.664125
min	0.000000	0.000000
25%	0.000000	208.000000
50%	1.000000	365.000000
75%	1.000000	515.000000
max	1.000000	7209.000000

Figure 2: Words Count vs. Label Description

title, the total number of news that contain censored words in their title is 343. We then aimed to investigate whether fake news articles are more likely to contain censored words in their corresponding titles. To test this hypothesis, we analyzed the dataset and found that all news articles with censored words in their titles were classified as fake.



```
[' ‘Maury’ Show Official Facebook Posts F*CKED UP Caption On Guest That Looks Like Ted Cruz (IMAGE)',
 ' Ohio Fireman In Deep Sh*t For HORRIBLE Remarks About Saving ‘N*****s’',
 ' The Resistance Just Gave Trump A Huge F**CK YOU On Election Night',
 ' President Vicente ‘I’m Not Building That F*cking Wall’ Fox Says Trump Is Like Hitler (VIDEO)',
 ' Angela Merkel’s Husband Is Taking Melania And Ivanka On A Tour That Will P*ss Trump Off',
 'WHOA! BREAKING NEWS: CNN Producer Caught On Undercover Tape Admitting Trump–Russia Coverage Is BULLSH*T…”Pr
 ' Bernie Sanders Just Told Republicans And Billionaires To F*** Off In The BEST Possible Way (VIDEO)',
 ' Trump Just Dedicated A F**king GOLF TROPHY To People Who Are Suffering In Puerto Rico',
 ' Restaurant Owner Calls Black Customer ‘N*****’ Then Throws Him Out Because ‘Trump Is President Now’',
 ' Republicans Turn On Trump, Throw Him Under The Bus For F*cking Up Health Care Plan (DETAILS)',
 ' Federal Judge Tells Anti–Vaxxers To Go F*ck Themselves, Upholds Strict California Vaccination Law',
 ' Trump’s Biggest, Most Pansy–A** Apologist Now Says He Is ‘Unacceptable’',
 ' On His Way Out The Door, Harry Reid Gives Big ‘F*ck You’ To Republicans And Hands Nevada To Hillary',
 'VIDEO: HARLEM BAR Kicks Customers Out For Wearing Trump Hats: “We don’t play that sh*t here”',
```

Figure 3: Examples of Titles with Censored Words

3 Predictive Task

The predictive task is to classify whether a given news article is true (1) or false (0), a binary classification problem. This task has the real-world significance of helping people to identify false news and stop the spreading of misinformation. To evaluate the models for this predictive task, we will use accuracy, precision, recall, F1-score, and balanced accuracy. Accuracy measures the proportion of correctly predicted labels, precision assesses the percentage of predicted true labels that are correct, recall evaluates how well the model identifies true labels among all actual true cases, F1-score balances precision and recall to give an overall performance measure, and balanced accuracy averages recall for both classes.

To assess the validity of the models' predictions, the evaluation is conducted on an independent validation set for parameter tuning and subsequently on a test set to see how well the model performs. By isolating these datasets during training, the risk of overfitting is minimized, and performance metrics reflect the models' ability to classify unseen data.

The dataset's text was preprocessed by converting all characters to lowercase and removing punctuation. Rows with missing or invalid entries in the text or label columns were dropped. For feature extraction, the Bag-of-Words and TF-IDF transform the text into numerical representations, with each limited to 1,000 features. The TF-IDF matrix contains the TF-IDF score for each term in each document. With each row representing a document and each column representing a term. For example, the entry at the (i,j) position in the TF-IDF matrix will be the jth term's TF-IDF score in ith document. The terms are sorted in alphabetical order and only the terms that have the highest TF-IDF score over all documents are chosen. The equation of calculating TF-IDF score are listed below:

$$TF(t,d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

$$IDF(t) = \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing term } t{+}1}\right)$$

$$TF\text{-}IDF(t,d) = TF(t,d) \cdot IDF(t)$$

Baseline Model 1: Bag-of-Words with Logistic Regression

The Bag-of-Words approach vectorizes the text by counting the occurrences of each word, limited to 1,000 words by frequency. This method captures the basic textual content without regard to word importance or context. A Logistic Regression classifier with class weighting to address imbalances was trained on these features. On the validation set, this baseline achieved strong performance, demonstrating its ability to classify fake news accurately by leveraging frequent word patterns. Its strength lies in simplicity, though it lacks contextual understanding, which may affect nuanced classification.

Baseline Model 2: TF-IDF with Logistic Regression

TF-IDF refines the Bag-of-Words representation by weighing words based on their importance, penalizing frequently occurring but less informative terms. The same Logistic Regression setup was used to classify these features. Although slightly outperformed by the Bag-of-Words approach, this method proved effective in identifying key distinguishing words for true and false articles. Its recall was slightly lower than Bag-of-Words, indicating potential difficulty in capturing some nuanced patterns. This approach is particularly useful when distinguishing articles relies on specific indicative terms.

Baseline Model 3: Hyperparameter Tuning with TF-IDF and Logistic Regression

The third baseline model extended the TF-IDF approach by introducing hyperparameter tuning for the Logistic Regression regularization parameter C. This process iteratively adjusted C to optimize the trade-off between underfitting and overfitting. The best performance was achieved with C=10. significantly improving the model's balanced accuracy and generalization. The hyperparameter-tuned model surpassed both Bag-of-Words and plain TF-IDF baselines, demonstrating the importance of optimization in model performance. It validated predictions effectively through rigorous evaluation on unseen test data, ensuring the robustness of its results.

4 Model

To improve the performance of the baseline model, we use the CLSTM model. The CLSTM (Convolutional LSTM) model leverages the combined strengths of Convolutional Neural Networks (CNNs) and Long Short-Term Memory Networks (LSTMs) for robust text classification.

The process starts with a frozen embedding layer initialized using pretrained 300-dimensional GloVe embeddings. This ensures stable and semantically rich vector representations of the input text. The model employs two 1D convolutional layers to extract local features such as n-grams. The first convolutional layer has 64 filters with a kernel size of 5, followed by another layer with 128 filters and the same kernel size. These layers capture indicative patterns in the text. Max-pooling layers with a kernel size of 4 follow each convolutional layer, reducing the dimensionality while preserving significant features. The output of the convolutional layers is transposed and passed to an LSTM with 256 hidden units.

This LSTM captures the sequential

dependencies in the text, processing it to retain meaningful contextual information. Dropout regularization (with a rate of 0.5) is applied to the final hidden state of the LSTM to prevent overfitting. The final hidden state is passed through a fully connected layer with a sigmoid activation function to output a probability score indicating whether an article is true or fake. The model is optimized using the Binary Cross-Entropy Loss (BCELoss) function and the Adam optimizer, with training conducted over multiple epochs.

$$\begin{aligned} \text{BCE Loss} = -\frac{1}{N}\sum_{i=1}^{N}\Big[& y_i \log(p_i) \\ &+ (1-y_i)\log(1-p_i)\Big] \end{aligned} \tag{1}$$

This model was selected because it addresses the limitations of the baseline models by leveraging both local feature extraction (via CNNs) and sequential dependency modeling (via LSTMs). The convolutional layers enhance the model's ability to detect indicative word combinations, while the LSTM ensures that contextual nuances are not lost, making it particularly effective for the fake news detection task. Compared to the simpler Bag-of-Words and TF-IDF baselines, the CLSTM provides a richer representation of the text, resulting in superior classification accuracy.

The optimization of the CLSTM involved multiple strategies. First, pretrained embeddings were used to initialize the embedding layer, reducing the need for extensive training data. The architecture includes dropout regularization and max-pooling to prevent overfitting, ensuring the model generalizes well to unseen data. Additionally, the learning rate was set to 0.001 with the Adam optimizer to balance convergence speed and stability.

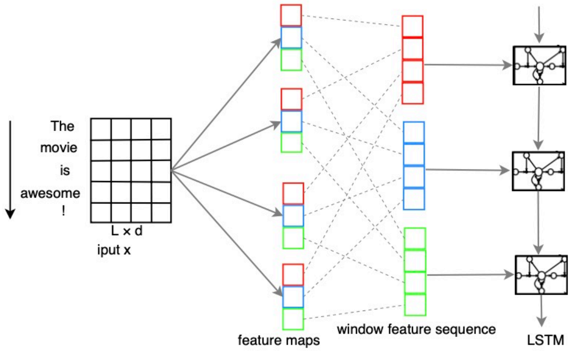


Figure 4: The architecture of C-LSTM for sentence modeling. Blocks of the same color in the feature map layer and window feature sequence layer correspond to features for the same window. The dashed lines connect the feature of a window with its source feature map. The final output of the entire model is the last hidden unit of the LSTM.

Potential Overfitting Issues

Examining the loss and accuracy outputs from the training logs revealed a steady decline in both training and validation losses, coupled with high validation accuracy. However, the narrowing gap between the training and validation losses in later epochs suggests the potential for overfitting if training continues without additional regularization. This is particularly evident in later epochs, where the validation accuracy plateaus while the training accuracy improves slightly, indicating that the model is beginning to memorize the training data. Specifically, as shown in the table below, the training loss consistently decreases throughout the epochs, reaching 0.0397 in the 10th epoch, while the test loss hovers around 0.061, showing minimal improvement after the 6th epoch. Similarly, the test accuracy remains stable at approximately 98.4% after this point, reflecting the model's limited ability to generalize further despite increased training efforts.

The table highlights that while the model demonstrates high accuracy, the negligible improvement in test loss and accuracy across

Epoch	Train Loss	Test Loss	Test Accuracy
1	0.5972	0.36	0.857
2	0.4042	0.3344	0.834
3	0.1175	0.0729	0.982
4	0.0776	0.0711	0.981
5	0.0709	0.09	0.972
6	0.0693	0.063	0.984
7	0.0616	0.0601	0.985
8	0.0542	0.0583	0.985
9	0.0461	0.0641	0.984
10	0.0397	0.061	0.984

Table 1: **Training and test losses along with test accuracy across 10 epochs**

later epochs, combined with the continued reduction in training loss, strongly indicates overfitting. To address this, additional regularization techniques, such as increasing dropout rates, early stopping, or weight regularization, could be applied to ensure the model generalizes more effectively to unseen data.

Scalability Analysis

Scalability was assessed by varying the size of the training data and monitoring the model's performance across multiple metrics: training loss, test loss, and test accuracy. Increasing the training set size generally improved performance metrics, as demonstrated in the three graphs below.

The Training Loss vs. Epochs graph reveals that with smaller training sizes (e.g., 0.2), the training loss starts high but decreases rapidly, reflecting fast convergence due to limited data complexity. As the training size increases, the initial loss decreases, and convergence becomes steadier. For larger datasets (e.g., 0.8), the training loss stabilizes at a lower value, demonstrating the model's ability to fit more complex data without overfitting.

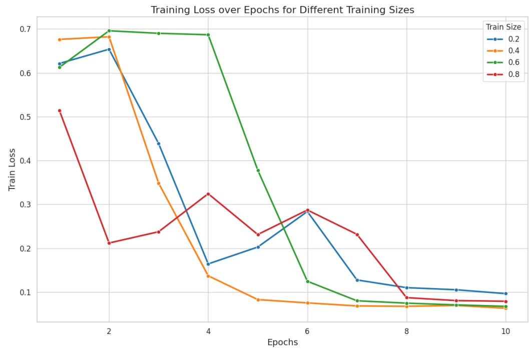


Figure 5: Training Loss over Epochs for Different Training Sizes

The Test Loss vs. Epochs graph shows that smaller training sizes initially result in higher test losses due to underfitting, as the model lacks sufficient data to generalize effectively. Larger training sizes, however, lead to significantly lower and more stable test losses across epochs, particularly after epoch 4. This improvement highlights the benefit of increased training data for reducing generalization errors.

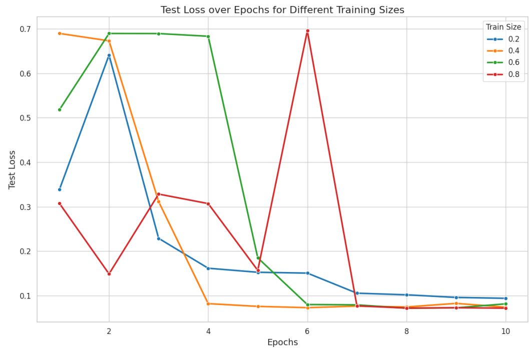


Figure 6: Test Loss over Epochs for Different Training Sizes

The Test Accuracy vs. Epochs graph illustrates a similar trend. For smaller training sizes, test accuracy starts lower and improves at a slower rate, plateauing early due to limited learning capacity. With larger training sizes, test accuracy consistently improves across epochs, eventually reaching nearly 98% for the largest training size (0.8). However, the diminishing returns after a certain point suggest that further increases in training size may require hyperparameter tuning or adjustments to the model architecture to continue improving performance.

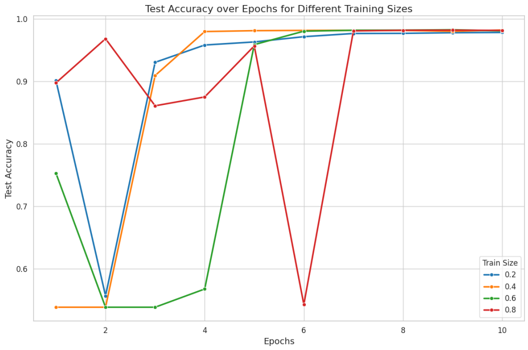


Figure 7: Test Accuracy over Epochs for Different Training Sizes

These graphs collectively indicate that the model scales well with increasing training data. However, careful tuning is required to maintain efficiency and prevent saturation in performance metrics as the dataset grows. While the model benefits significantly from larger datasets, computational costs and convergence time also increase, emphasizing the importance of balancing scalability and practicality.

Generalization of the CLSTM Model

To evaluate the generalization capability of the CLSTM model, it was tested on a new dataset containing a different set of news articles. This dataset, which differs in style, vocabulary, and potentially thematic focus from the original training data, provides a robust benchmark for assessing how well the model adapts to unseen distributions.

The model achieved a test loss of 1.9174 and a test accuracy of 50.3% on the new dataset. This performance is significantly lower than the results on the original dataset, where test accuracy exceeded 98%, indicating that the model struggles to generalize to new data sources. This gap in performance highlights the limitations of the CLSTM model in handling domain shifts or variations in data distribution.

The primary reason for this poor generalization lies in the vocabulary and contextual dependencies of the new dataset. The CLSTM relies on a vocabulary built from the original training data, which may not fully overlap with the vocabulary of the new dataset. Words or phrases unique to the new dataset are mapped to an ”unknown” token during preprocessing, causing a loss of semantic richness and reducing the model’s ability to make accurate predictions. Additionally, the model’s reliance on pretrained embeddings may amplify this effect if the embeddings are not fine-tuned to reflect the nuances of the new dataset.

Another contributing factor is the potential difference in writing style or linguistic patterns between the datasets. The CLSTM model’s convolutional layers are tuned to detect specific local patterns, while the CLSTM layers model sequential dependencies. If these patterns or dependencies differ significantly in the new dataset, the model’s performance is adversely affected.

5 Literature

5.1 Related Literature

CLSTM Neural Network for Text Classification

Our model builds upon the foundation laid by Zhou et al. (2015), who introduced the CLSTM neural network for text classification. This model integrates convolutional neural networks (CNNs) with long short-term memory (LSTM) networks, enabling it to effectively capture both local patterns (e.g., n-grams) and sequential dependencies in text data. Zhou et al. demonstrated that CNNs excel at extracting local features, while LSTMs model temporal relationships across sequences. By merging these two architectures, the CLSTM offers a comprehensive representation of textual features [1]. This work directly inspired the design of our CLSTM model, which applies a similar hybrid approach to the domain of fake news detection. Our adaptation aims to address the limitations of traditional content-only models by capturing nuanced word patterns and contextual relationships in textual data.

Incorporating Social Context into Fake News Detection

Beyond text-based approaches, researchers have highlighted the importance of incorporating social context into fake news detection frameworks. Tacchini et al. (2017) and Shu et al. (2017) emphasized that user interactions, such as likes and shares, provide valuable contextual signals that improve detection accuracy. These studies revealed the limitations of content-only methods and demonstrated the potential of hybrid models that integrate external social context [2, 3]. Building on this idea, Ruchansky et al. (2017) proposed the CSI model, which combines textual features with user behavior and social context, achieving superior performance over traditional approaches. Similarly, Monti et al. (2019) utilized geometric deep learning to model social media interactions as graphs, emphasizing the role of propagation patterns in detecting misinformation [4, 6].

Adversarial and User-Specific Biases

More recently, research has focused on addressing adversarially generated fake news and user-specific biases. Zellers et al. (2019) introduced Grover, a transformer-based model designed to both generate and detect adversarial fake news. Grover leverages large-scale pretraining and fine-tuning to identify stylistic and structural cues that distinguish machine-generated misinformation [5]. Dou et al. (2021) further extended this work by exploring user preference-aware detection, showcasing the importance of personalization in understanding biases in fake news consumption [7]. Collectively, these studies highlight the evolution of fake news detection from simple content-based approaches to sophisticated frameworks that incorporate adversarial and contextual considerations.

5.2 State-of-the-Art Methods

CSI: A Hybrid Deep Model for Fake News Detection

Ruchansky et al. (2017) introduced the CSI model, which integrates content representation through recurrent neural networks (RNNs), user behavior analysis, and social context to address the multifaceted nature of fake news. By combining these modalities, CSI achieves superior accuracy compared to content-only models while maintaining computational efficiency. This model sets a benchmark for hybrid approaches in fake news detection [4].

Defending Against Neural Fake News

Zellers et al. (2019) developed Grover, a transformer-based model aimed at detecting adversarially generated fake news. Grover's dual-purpose design allows it to both generate and detect machine-generated misinformation. By leveraging large-scale pretraining and fine-tuning, Grover adapts to emerging threats, identifying stylistic and structural cues indicative of neural-generated content. This makes it a highly relevant tool in today's misinformation landscape [5].

5.3 Comparison to Our Findings

Both CSI and Grover underscore the limitations of content-only approaches like our CLSTM. CSI's inclusion of user behavior and social context highlights a critical gap in our CLSTM model, which solely relies on textual features and lacks contextual integration. Similarly, Grover demonstrates the necessity of advanced architectures capable of handling adversarially generated content. While our CLSTM effectively captures local patterns and sequential dependencies in human-written fake news, it is less equipped to address the sophisticated challenges posed by adversarial and contextual scenarios. These comparisons emphasize the importance of integrating multi-modal features and adapting to adversarial conditions to build robust fake news detection frameworks.

6 Results and Conclusion

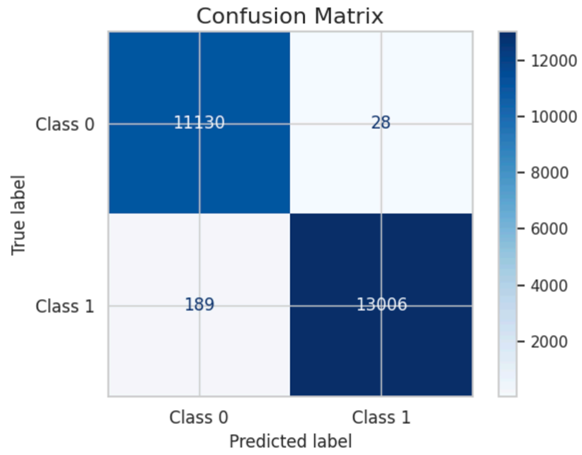


Figure 8: Confusion Matrix for CLSTM

Through this project, we compared the performance of baseline models and an advanced CLSTM-based model for fake news detection. The baseline models, including Bag-of-Words Logistic Regression, TF-IDF Logistic Regression, and a hyperparameter tuned Logistic Regression, achieved accuracies of 97.00%, 96.62%, and 97.32%, respectively, with F1-scores ranging from 96.91% to 97.52%. In contrast, the LSTM model outperformed all baselines with an accuracy of 99.1% and an F1-score of 99.2%, demonstrating its ability to capture contextual and sequential information beyond the capabilities of frequency-based methods.

The significance of our result is that it demonstrates the ability of our model to accurately predict whether a news article is fake or real based solely on its text. This has profound implications for combating the spread of misinformation, which can influence public opinion, create panic, and erode trust in the media. By integrating such a model into news platforms and social media, we can help flag or filter fake news in real-time, promoting transparency.

The feature representation of the CLSTM model was its use of GloVe embeddings, which provided rich semantic representations of words, enabling the model to effectively capture the relationships and meanings within the text. This feature representation significantly outperformed Bag-of-Words and TF-IDF, which rely on frequency-based metrics and fail to account for word semantics or context. The convolutional layers in the CLSTM model extracted local patterns in the sequences, while the LSTM layer modeled long-range dependencies, essential for understanding the sequential nature of text. The model's parameters, such as the pre-trained embedding matrix and LSTM hidden states, contributed to its success by enabling it to balance generalization and specificity. In contrast, the logistic regression models lacked the capacity to capture these deeper, context-aware relationships, limiting their ability to achieve comparable performance, and that's why they failed in terms of the accuracy when compared to the CLSTM model.

References

[1] Zhou, C., Sun, C., Liu, Z., & Lau, F. C. M. (2015). A C-LSTM neural network for text classification. *arXiv preprint arXiv:1511.08630*. Retrieved from `https://arxiv.org/abs/1511.08630`.

[2] Tacchini, E., Ballarin, G., Della Vedova, M. L., Moret, S., & de Alfaro, L. (2017). Some like it hoax: Automated fake news detection in social networks. *Proceedings of the Second Workshop on Data Science for Social Good (SoGood 2017)*, 38–51.

[3] Shu, K., Wang, S., & Liu, H. (2017). Beyond news contents: The role of social context for fake news detection. *Proceedings of the 2017 ACM on Web Science Conference (WebSci '17)*, 317–326. `https://doi.org/10.1145/3091478.3091487`.

[4] Ruchansky, N., Seo, S., & Liu, Y. (2017). CSI: A hybrid deep model for fake news detection. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*, 797–806. `https://doi.org/10.1145/3132847.3132877`.

[5] Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., & Choi, Y. (2019). Defending against neural fake news. *Advances in Neural Information Processing Systems (NeurIPS '19)*, 32, 9054–9065.

[6] Monti, F., Bronstein, M. M., & Castellani, U. (2019). Fake news detection on social media using geometric deep learning. *Proceedings of the 2019 ACM on Conference on Information and Knowledge Management (CIKM '19)*, 959–966. `https://doi.org/10.1145/3357384.3358016`.

[7] Dou, Y., Shu, K., Yu, P. S., Sun, L., & Yu, L. (2021). User preference-aware fake news detection. *Proceedings of the 2021 ACM International Conference on Web Search and Data Mining (WSDM '21)*, 777–785. `https://doi.org/10.1145/3437963.3441782`.