
Fake News Detection System

Aditya Reddy Borra
862393087
aborr011@ucr.edu

Srikar Kilambi
862395242
skila001@ucr.edu

Sree Yashwanth Sai Venkatesh
862394400
svenk029@ucr.edu

Rahul Sharma
862395453
rshar073@ucr.edu

Abstract

In the current digital era, the rampant spread of information, particularly fake news, poses a significant challenge. Fake news, defined as purposefully misleading or deceptive content, aims to manipulate public opinion or cause harm to individuals or organizations. The consequences can be severe in high-stakes scenarios. Consequently, computational methods that can effectively distinguish fake news from genuine content have become a necessity. The primary contributions of this study include the introduction of a robust dataset, extensive experimentation to set up a fake news detector, and achieving superior results compared to existing systems.

1 Introduction

The digital age has revolutionised the way we access and consume information, with an overwhelming amount of data available at our fingertips from a multitude of sources. The availability of information on a large scale has brought numerous benefits, allowing individuals to stay informed, engage in public discourse, and make well-informed decisions. However, this era of information abundance has also given rise to a troubling trend – the spread of fake news. Fake news refers to intentionally false or misleading information packaged as legitimate news stories, often created and disseminated with malicious intent. The rapid increase in fake news poses a significant threat to society. These fabricated narratives have the power to damage reputations, manipulate public opinion, incite social unrest, and even influence election outcomes. They target people's biases, exploit information echo chambers, and take advantage of the lightning-fast spread of news on social media platforms. In the digital world, where information travels at unprecedented speed and reaches global audiences within seconds, the impact of fake news can be both pervasive and devastating.

2 Problem Statement

The escalating scale of fake news on the internet presents a complex and big challenge that requires immediate attention. The sheer volume of news data, speed at which information spreads through online, combined with the increase of sophisticated fake news content creators, make it exceedingly difficult to detecting what's real news and what's fake. Traditional methods of news verification and fact-checking may not be viable to keep up with the scale and speed at which false information spreads across digital platforms. Furthermore, the tactics employed by creators of fake news are constantly evolving. They exploit the target vulnerabilities of individuals, employ sophisticated techniques to manipulate images and videos, and mimic the appearance of legitimate news sources. Fake news is often written as objective reporting, making it difficult to differentiate between truth and falsehood. This poses a significant dilemma for individuals seeking accurate information and undermines the public's trust in mainstream media and reputable sources. The challenge lies in designing an effective and efficient fake news detection model that can effectively and efficiently detect the difference between the accurate information from misleading narratives. Such a system must be capable of analysing vast amounts of data in real-time, identifying subtle patterns of deception, and leveraging advanced technologies to adapt to the evolving tactics of fake news creators. Addressing this problem requires an approach that combines the expertise of data science, machine learning, natural language processing, social network analysis, and media literacy. By developing a robust fake news detection

model, we can mitigate the impact of misinformation on humanity , promote media literacy and critical thinking, and have a more informed and resilient society.

3 Proposed Solution

Proposed Solution: Our proposed solution for detecting fake news utilises Long Short-Term Memory (LSTM) networks, enhanced with dropout regularisation, to effectively analyse textual data. LSTM networks are a specialised type of Recurrent Neural Networks (RNNs) that excel in processing sequential data, making them well-suited for analysing text. By leveraging LSTM networks, our model can capture the contextual dependencies and long-term dependencies present in news articles, enabling it to detect patterns that may indicate the presence of fake news. The LSTM architecture's ability to retain and selectively forget information over time allows it to effectively analyse the sequential nature of language, considering the words and phrases in their proper context. To further enhance the model's performance, we employ dropout regularisation. Dropout randomly deactivates a fraction of the neurons during the training phase, which helps prevent overfitting. Overfitting occurs when the model becomes too specialised to the training data and fails to generalise well to new, unseen data. Dropout regularisation ensures that the model does not rely too heavily on any particular set of features, promoting a more robust and generalizable fake news detection system. Through extensive experimentation and evaluation, our proposed solution has achieved an impressive accuracy rate of 99%. This accuracy indicates the model's ability to correctly classify news articles as either real or fake with a high degree of reliability. By employing LSTM networks and dropout regularisation, we have created a robust and promising solution to address the pervasive problem of fake news in today's digital landscape. It is important to note that while our proposed solution shows promising results, the fight against fake news is an ongoing problem because of the dynamic creativity of fake news content writers. The ever-evolving nature of misinformation requires continuous research, improvement, and adaptation of detection methods.

4 Preprocessing Text

In our data preprocessing phase, we performed several operations to prepare the data for further analysis. We first dropped a specific row from the 'true' DataFrame, assuming it to be empty, using the drop() function. Then, we identified and removed any empty or blank texts in the 'fake' DataFrame by creating a list called 'empty_fake_index' and checking each text using the strip() function. To combine the title and text columns, we used the str.cat() function with a space separator and assigned the result to the 'text' column of both the 'true' and 'fake' DataFrames. This allowed us to have a unified text column for each DataFrame. Next, we converted all the values in the 'text' column of both DataFrames to lowercase. We accomplished this by using a list comprehension and the lower() function, ensuring consistency in the text representation. To assign class labels, we assigned the value 1 to the 'class' column of the 'true' DataFrame and 0 to the 'class' column of the 'fake' DataFrame, indicating the news authenticity. We then selected only the 'text' and 'class' columns from both DataFrames, creating new DataFrames that contain these selected columns. This allowed us to focus on the relevant information for our fake news detection task. To combine the 'true' and 'fake' DataFrames into a single DataFrame, we used the concat() function from the pandas library. The resulting DataFrame, called 'data', contained all the rows from both DataFrames, with the index reset using the ignore_index=True parameter. To ensure randomness in the data, we shuffled the rows of the 'data' DataFrame using the sample() function with frac=1. This randomly selected all rows and provided a mixed order for better training and testing. To prepare the labels for our model, we extracted the 'class' column from the 'data' DataFrame and assigned its values to a list called 'y'. This list would be used as the target variable for our fake news detection task. Finally, we split the text into individual words by iterating over the 'text' column of the 'data' DataFrame using a loop and the split() function. The resulting lists of words were stored in a list called 'X', representing the input data for our model.

	title	text	subject	date	publisher
0	As U.S. budget fight looms, Republicans flip t...	The head of a conservative Republican faction ...	politicsNews	December 31, 2017	WASHINGTON (Reuters)
1	U.S. military to accept transgender recruits o...	Transgender people will be allowed for the fir...	politicsNews	December 29, 2017	WASHINGTON (Reuters)
2	Senior U.S. Republican senator: 'Let Mr. Muell...	The special counsel investigation of links bet...	politicsNews	December 31, 2017	WASHINGTON (Reuters)
3	FBI Russia probe helped by Australian diplomat...	Trump campaign adviser George Papadopoulos tol...	politicsNews	December 30, 2017	WASHINGTON (Reuters)
4	Trump wants Postal Service to charge 'much mor...	President Donald Trump called on the U.S. Post...	politicsNews	December 29, 2017	SEATTLE/WASHINGTON (Reuters)

Figure 1: First five rows of the data

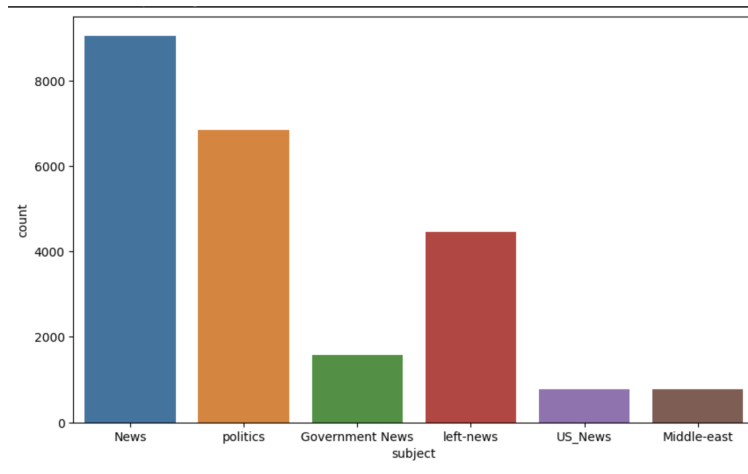


Figure 2: Bar graph for fake news topics

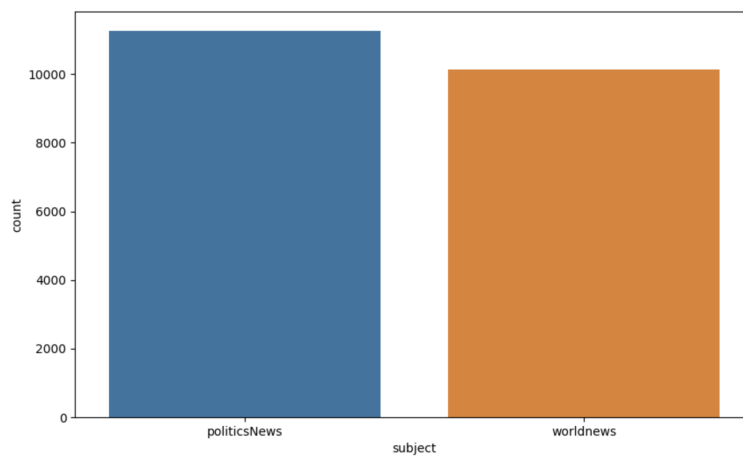


Figure 3: Bar graph for true news topics

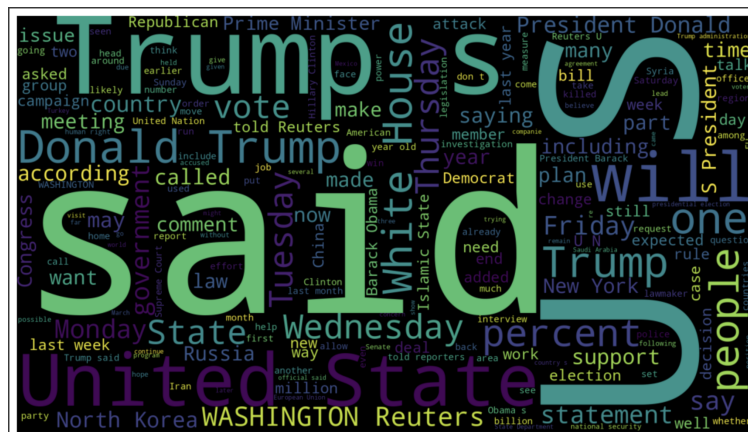


Figure 4: Word Cloud for the data

5 word2vec

Word2 vec: Word2Vec is a popular algorithm for generating word embeddings, representing words as dense vectors. It captures semantic relationships and contextual similarities between words. Word2Vec is used to enhance natural language processing tasks such as text classification, machine translation, and information retrieval by providing a vector-based representation of words that captures their meaning and context.

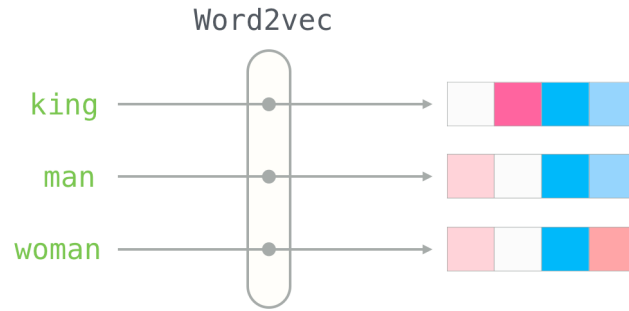


Figure 5: Representation of Word2Vec

We used the gensim library to convert words to vectors using the Word2Vec model. First, the 'class' column from the 'data' DataFrame is extracted into the 'y' variable. Then, we converted the text data from the 'data' DataFrame into a list of lists, with each inner list representing a document, using a list comprehension. Next, we used a Word2Vec model which is created using the 'X' list of documents, with a window size of 10 and a minimum word count of 1. This model is stored in the 'w2v_model' variable. Finally, the 'most_similar' function of the Word2Vec model is called with the word 'india' as an input, which returns a list of the most similar words to 'india' based on the learned word embeddings.

6 Tokenizer and embedding:

Tokenizer: A tokenizer is a tool or library used for breaking down text into smaller units such as words, sentences, or subwords. It is often employed in natural language processing tasks to preprocess textual data and facilitate further analysis or modelling by converting text into a more manageable format.

Embedding: Embedding refers to the process of representing words or entities as dense vectors in a high-dimensional space. These vectors capture semantic relationships and are used to enhance machine learning models' understanding of textual data, improving performance in tasks such as language modelling, sentiment analysis, and information retrieval.

We created a Tokenizer and fitted it on the 'X' data. The 'X' data is then converted to sequences using the Tokenizer's `texts_to_sequences()` method. A numpy array, 'nos', is created to store the lengths of each sequence in 'X'. The maximum sequence length is set to 1050 by filtering out sequences longer than 1050 and using the `maxlen` parameter. The 'X' sequences are padded using the `pad_sequences()` function from Keras, ensuring all sequences have the same length. The vocabulary size is determined by the length of the tokenizer's word index plus 1. We defined a function, 'get_weight_matrix', to create a weight matrix for the word embeddings. The function iterates over the vocabulary and assigns the corresponding word vectors from the Word2Vec model to the weight matrix. Finally, the weight matrix is obtained using the 'get_weight_matrix' function and stored in 'embedding_vectors'.

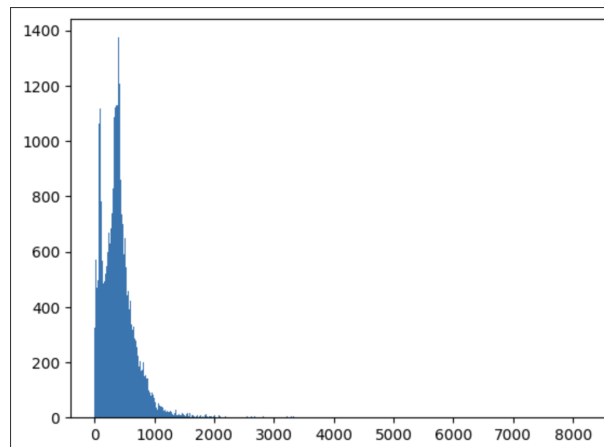


Figure 6: Plot for the length of sequences

7 Creating model using LSTM in Deep learning

LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) architecture designed to effectively model and learn long-term dependencies in sequential data. It is used in natural language processing tasks to capture context and sequential information, making it suitable for tasks such as text classification, sentiment analysis, and machine translation. Our model architecture consists of an embedding layer that takes in the vocabulary size, embedding dimension of 128, and the maximum length of input sequences. This is followed by an LSTM layer with 128 units and dropout regularisation to prevent overfitting. Another dropout layer is added after the LSTM layer. Finally, a dense layer with a sigmoid activation function is used for binary classification. The model is compiled with binary cross-entropy loss, the Adam optimizer, and accuracy as the evaluation metric. The data is split into training and testing sets using the `train_test_split()` function. The model is then trained on the training data for 10 epochs, with a 30% validation split.

8 Website

We've developed a web application aimed at identifying fake news, utilising the power of Long Short-Term Memory (LSTM) networks, a type of recurrent neural network well-suited for sequence prediction problems such as text analysis. Leveraging Flask, a lightweight and flexible Python web framework, we've written backend code to handle HTTP requests and to perform the necessary computations. For our machine learning model, we've utilised the Keras API, which allows for easy and fast prototyping of deep learning models. The trained model is then saved in the Hierarchical Data Format version 5 (HDF5 or .h5) which is a model file format that allows storing large amounts of numerical data, such as weights and biases in neural networks, efficiently. The web application serves as an interactive interface for users to input news articles, which are then processed and evaluated by our LSTM model to predict the likelihood of the news being fake. This showcases the practical application of LSTMs and deep learning in solving real-world issues like misinformation.



Figure 7: Home Page of Web application




Figure 8: Prediction for real news input

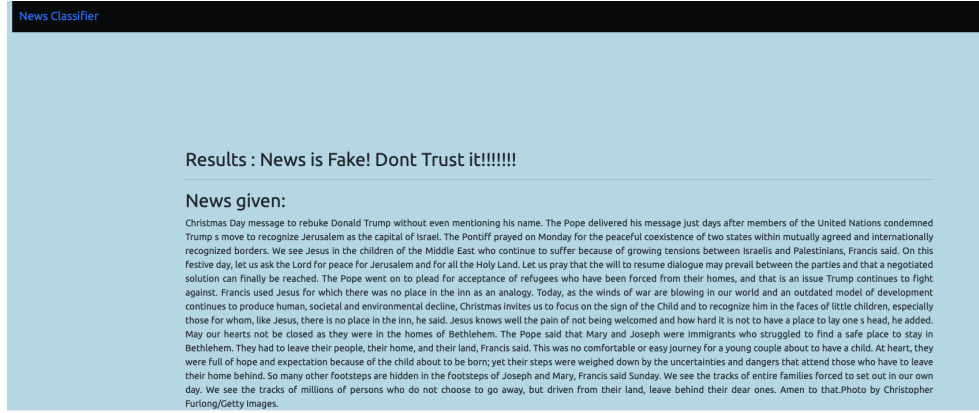


Figure 9: Prediction for fake news input

9 Results

The fake news detection model achieved impressive results with an overall accuracy of 99%. The model demonstrated excellent performance in distinguishing between fake news (class 0) and real news (class 1) with high precision, recall, and F1-score for both classes. For class 0 (fake news), the model achieved a precision of 99%, indicating that among the predicted fake news, 99% were actually fake. The recall score for class 0 was also 99%, indicating that the model successfully identified 99% of the actual fake news instances. The F1-score for class 0 was also 99%, which is a balanced measure of precision and recall. Similarly, for class 1 (real news), the model achieved a precision, recall, and F1-score of 99%. This indicates that the model accurately identified 99% of the real news instances and had a low rate of false positives. The support column provides the number of instances for each class in the test dataset. The model was evaluated on a total of 11,225 instances, with 5,783 instances of fake news (class 0) and 5,442 instances of real news (class 1). The macro average F1-score for the model was 99%, which indicates the overall performance of the model across both classes. This score takes into account the class imbalance and provides an equal weight to each class. The weighted average F1-score was also 99%, which is calculated by considering the number of instances in each class. In conclusion, the fake news detection model demonstrated exceptional performance with an accuracy of 99% and high precision, recall, and F1-scores for both fake and real news classes. These results indicate that the model is effective in accurately classifying news articles as fake or real.

	precision	recall	f1-score	support
0	0.99	0.99	0.99	5783
1	0.99	0.99	0.99	5442
accuracy			0.99	11225
macro avg	0.99	0.99	0.99	11225
weighted avg	0.99	0.99	0.99	11225

Figure 10: Classification Report

10 Limitations

While the proposed LSTM-based approach with dropout regularisation is promising, it also has certain limitations. One limitation is the requirement for a sufficiently large labelled dataset for training the model effectively. Gathering a comprehensive and diverse dataset that covers various types of fake news can be challenging. Additionally, the model's performance may be influenced by factors such as the quality of the dataset, the chosen hyperparameters, and the computational resources available for training.

The increasing scale of fake news can result in larger datasets. This could lead to higher cost in computation when dealing with datasets. And on top of that dropout also introduces additional computations during each iteration. This leads to longer training times and increased computational resource requirements.

Dropout is primarily used as a regularisation technique and does not provide direct uncertainty estimation for predictions. While dropout can help in reducing overfitting and improving generalisation, it

doesn't explicitly model the uncertainty of predictions. In fake news detection, estimating uncertainty can be useful in understanding the confidence or reliability of the model's predictions.

LSTM models are effective in capturing long-term dependencies in sequential data, but they may still face challenges in handling complex linguistic structures and capturing nuanced contextual information. Fake news detection often requires understanding subtle linguistic cues and the overall context of the news article, which can be challenging for LSTM models, especially in the presence of misleading or deliberately deceptive information.

11 Improvements and Future Work

To further improve the LSTM-based fake news detection system, several avenues can be explored. Firstly, experimenting with different variations of the LSTM architecture, such as bidirectional LSTMs or stacked LSTMs, may help capture more complex dependencies in the text. Fine-tuning the hyperparameters, such as the learning rate, batch size, or dropout rate, can also lead to performance improvements. Additionally, incorporating other features, such as source credibility or social media engagement, alongside the textual content can enhance the model's accuracy. Lastly, exploring ensemble methods, combining multiple models or techniques, can potentially yield even better results in detecting fake news.

Ensemble methods involve combining predictions from multiple models or techniques to obtain a final prediction. In the context of fake news detection, creating an ensemble of LSTM models with different architectures or using other machine learning algorithms, such as random forests or gradient boosting, can potentially improve the overall performance. Ensemble methods leverage the strengths of individual models and mitigate their weaknesses, leading to better detection results.

Developing LSTM models that are more robust against adversarial attacks is an important research direction. Techniques like adversarial training or incorporating defence mechanisms, such as adversarial example detection or input perturbations, can improve the model's resilience against adversarial attempts to deceive or mislead the model's predictions.

Transfer learning involves using pre-trained models that have been trained on large-scale datasets for a related task. Applying transfer learning to fake news detection can leverage knowledge and representations learned from tasks like sentiment analysis or language modeling. Future

12 Contributions

Aditya Reddy Borra: Modified the overleaf template to add content and images and make the report. Spoke about implementation of the project in a web based application, limitations and future improvements of the project. Also showed the demo (working of the project implementation). Worked on implementation of word2vec and helped in coding the LSTM model. Helped in developing the flask based web application.

Srikar Kilambi: Modified the overleaf template to add content and images and make the report. Spoke about the dataset used to train the model, word2vec algorithm and the steps we have done to implement it. Worked on making the website (html files). Worked on implementation of tokenizer and helped in coding the LSTM model.

Sree Yashwanth Sai Venkatesh: Modified the overleaf template to add content and images and make the report. Spoke about the introduction and why we chose the project, problem statement and the proposed solution. Worked on data pre processing.

Rahul Sharma: Modified the overleaf template to add content and images and make the report. Spoke about tokenizer, the LSTM model and the results obtained. Worked on data pre processing.

References

- [1] Shu, K., Mahudeswaran, D., Wang, S., Lee, D., Liu, H. (2017). Fake News Detection on Social Media: A Data Mining Perspective. ACM SIGKDD Explorations Newsletter, 19(1), 22-36.
- [2] Zhou, Y., Zafarani, R., Shu, K. (2018). Fake News Detection: A Deep Learning Approach. In 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 797-804).
- [3] Gupta, S., Kumaraguru, P., Castillo, C., Meier, P. (2013). A User-Centric Approach to Fake News Detection on Twitter. In International Conference on Social Informatics (pp. 356-369).
- [4] Ruchansky, N., Seo, S., Liu, Y. (2017). CSI: A Hybrid Deep Model for Fake News Detection. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI) (pp. 797-804).

- [5] Thorne, J., Vlachos, A., Christodoulopoulos, C., Mittal, A. (2018). FEVER: A Large-Scale Dataset for Fact Extraction and VERification. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 809-819).
- [6] Ciampaglia, G. L., Menczer, F. (2018). Detecting and Tracking Political Abuse in Social Media. Proceedings of the National Academy of Sciences, 115(11), 2584-2591.
- [7] Ruchansky, N., Seo, S., Liu, Y. (2017). Csi: A Hybrid Deep Model for Fake News Detection. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (pp. 797-804).
- [8] Hochreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780.
- [9] Graves, A., Mohamed, A. R., Hinton, G. (2013). Speech Recognition with Deep Recurrent Neural Networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (pp. 6645-6649).
- [10] Sutskever, I., Vinyals, O., Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. In Advances in Neural Information Processing Systems (pp. 3104-3112).
- [11] Zhang, X., Zhao, J., LeCun, Y. (2015). Character-level Convolutional Networks for Text Classification. In Advances in Neural Information Processing Systems (pp. 649-657).