Leveraging Natural Language Processing for Fake News Detection

Haoran Chang
hchang27@ucla.edu
University of California, Los Angeles
Los Angeles, CA, USA

Zhi Li
mikelili@ucla.edu
University of California, Los Angeles
Los Angeles, CA, USA

Jie Shao
jieshao@g.ucla.edu
University of California, Los Angeles
Los Angeles, CA, USA

Yuchen Wang
ycw509@g.ucla.edu
University of California, Los Angeles
Los Angeles, CA, USA

ACM Reference Format:
Haoran Chang, Zhi Li, Jie Shao, and Yuchen Wang. 2022. Leveraging Natural Language Processing for Fake News Detection. In *CS 263 Spring 2023: Natural Language Processing, March–June, 2023, Los Angeles, CA.* ACM, New York, NY, USA, 4 pages. https://doi.org/0.0/0.0

ABSTRACT

This study proposes an effective approach for fake news detection leveraging a fine-tuned BERT model. By adapting BERT's language understanding capabilities, we achieved improved performance in identifying fake news articles. Our approach achieved better result comparing to traditional machine learning methods like CNN and base BERT. This project underscores the potential of BERT models in addressing the fake news issue and suggests new opportunities for further research in this field.[1]

1 INTRODUCTION

The digital age brings with it the challenge of fake news, deceptive information masquerading as genuine news. Traditionally, detection strategies have relied on machine learning models like Convolutional Neural Networks (CNNs), but their effectiveness in complex language tasks, such as text classification, is limited. A key drawback of CNNs is their local understanding of text. CNNs, while excellent for spatial hierarchies in image data, often fail to capture the long-range dependencies in textual data. They lack a global perspective of the text, focusing on local, immediate features and making them less effective in understanding the overall context, which is vital for tasks like fake news detection.

This leads us to explore the Bidirectional Encoder Representations from Transformers (BERT), a sophisticated model in the field of natural language processing. BERT addresses the limitations of traditional models by understanding the contextual relationships between words using self-attention mechanisms, reading text in a truly bidirectional way. Additionally, BERT's ability to be pre-trained on a large corpus and fine-tuned for specific tasks allows it to better adapt to various language tasks, including fake news detection. [2]

In this study, we compare the fine-tuned BERT model with traditional methods using a large dataset of fake and genuine news articles. Our findings highlight the potential of BERT in addressing the challenge of fake news detection, offering valuable insights for future research in this field.

2 LEMMATIZATION

To address the unstructured and noisy nature of this data complicates the extraction of valuable knowledge. We employed text lemmatization. Lemmatization transforms words to their root form, enhancing data quality, reducing dimensionality, and facilitating more precise and efficient data analysis. [4]

We offers an in-depth exploration of an advanced lemmatization technique aimed at augmenting preprocessing efficiency for a variety of Natural Language Processing (NLP) applications. By harnessing the power of WordNet's lexical database for word stemming and Python's Natural Language Toolkit (NLTK), our method embodies a versatile, context-aware approach to data cleaning and word normalization, leading to notable improvements in data quality and the effectiveness of NLP tasks.

CS 263 Spring 2023, March–June, 2023, Los Angeles, CA

ACM ISBN XXXXX…$0.00
https://doi.org/0.0/0.0

2.1 Methods and Implementation

Our lemmatization technique makes extensive use of Python's NLTK and WordNet's lexical database. The method begins by assigning each word in the text a Part-of-Speech (POS) tag using the pos_tag function from NLTK. These tags are then translated into WordNet POS tags to create a richer semantic context for the words. Guided by these accurate POS tags, the WordNetLemmatizer from NLTK lemmatizes each word, ensuring a more precise and context-aware lemmatization.

In terms of implementation, our method is designed to seamlessly integrate with pandas DataFrames, a popular choice for handling structured data in data science. The function can be applied directly to the DataFrame, leading to a streamlined and efficient preprocessing workflow. User experience is further enriched with real-time progress updates, courtesy of the tqdm library.

Beyond lemmatization, our method also includes thorough text cleaning operations. Regular expressions and Python's string manipulation functions are utilized to scrub off URLs, HTML tags, non-ASCII characters, redundant spaces, and other noise elements, ensuring a cleaner and more homogeneous dataset.

2.2 Significance

Our lemmatization method stands apart from conventional techniques by virtue of its context-awareness, enabled by the use of POS tags. While traditional lemmatization techniques treat words in isolation, our method appreciates the broader context of word usage, which allows for a more accurate lemmatization. By transforming words to their root form, the technique reduces data dimensionality, simplifying the computational task and potentially enhancing the effectiveness of downstream NLP models. Furthermore, the thorough cleaning operations embedded in our method refine the quality of the input data, filtering out irrelevant noise and allowing models to focus on the crux of the text data, leading to a more focused analysis and accurate model.

3 BASELINE MODEL

As a comparison of the performance to the state-of-the-art BERT model, a convolutional neural network[3] was built as the baseline performance. A convolutional neural network can capture the spatial information of the input data by the kernels and text as sequential data are expected to have spatial correlations. In the report, the original fake dataset and true dataset are well balanced, hence the data augmentation were not performed in this task. Both the data set have five columns, and the column used as the input data is the "text" column because it contains all the useful information for this task. The average length of the text is 400 words, that is, 400 dimensions. We decided to use 500 dimensions for the input data and pad those are less than 500 words and simply truncate those are over 500 words like the keras.pad_sequences function. The word embedding method is GloVe50d.

3.1 Baseline Model Architecture

The baseline model is constructed with two convolutional layers, each followed by a ReLU activation function and a max-pooling layer. After these, the network flattens the output and feeds it into a fully connected layer, followed by another ReLU activation. Finally, there's another fully connected layer that maps the network's output to two classes, designed for binary classification tasks such as fake news detection, where the aim is to classify news as either "true" or "fake".

3.2 Performance Analysis

The accuracy of the base model is 89.88%, the precision is 88.76% and the F1 score is 89.31%. It performed worse than the state-of-the-art BERT model and the following reasons can be discussed. First, the baseline model is simpler and may lack the depth and complexity to capture high-level semantic meanings in the text, which is a crucial aspect in tasks like fake news detection. Second, unlike BERT, which utilizes transformer architecture to understand the context of words in a sentence, the baseline model doesn't have a built-in mechanism for understanding the order or context of words. Moreover, BERT is a pre-trained model that leverages a large corpus of text data and captures generalized language understanding before fine-tuning for specific tasks. In contrast, the baseline model learns from scratch and may not generalize well.

4 MODEL ARCHITECTURE

We are using the BERT, which is a pre-trained transformer-based model developed by Google for natural language processing tasks. Compared with CNN and RNN, BERT could better extract the context information since its bidirectional feature makes it understand the context more effectively. The transformer-based architecture allows the BERT to weigh the importance of words in the text relative to each other since it is using the self-attention mechanism. Since the BERT model is a pre-trained model that does not work perfectly on every task, we need to fine-tune this model for our specific target in order to achieve the best accuracy, which allows the model to adapt to the particularities of the task. Therefore, we used nn.Module in PyTorch to reference the BERT model and added some layers to improve its performance.

4.1 torch.nn.Module

The torch.nn.Module is a popular open-source machine learning library designed in Python and serves as the foundation

for all neural network modules in PyTorch. Any model we create by ourselves will become a subclass of this class. These modules are designed to be hierarchical, implying that one module can contain several other modules as attributes. All these attribute modules' parameters are incorporated when the parent module's parameters() function is called. The library operates in two modes, which are the training mode and the evaluation mode. This feature is useful when specific behaviors in our modules should vary between the training and inference stages. Additionally, when creating a subclass, it is important to define a forward() function. This function determines the actions the module should take with the received inputs, serving as the point where we can call our layers as per the designed model architecture. The forward() method is called when we pass input data into an instance of the module.

4.2 Choice of Layers and Parameters

Besides the pre-trained Bert model, we added some layers in order to improve the overall performance of our model. Our first layer after Bert is the fully connected layer, which converts the dimension from 768 to 512. The second layer is an activation function ELU() that could be used to speed up learning and produces more accurate results. We also tried ReLU, which is also an activaion function. Nevertheless, ELU (Exponential Linear Units) demonstrates superior performance because it can handle negative values. This ability enables them to shift mean unit activations closer to zero, similar to batch normalization, but with less computational complexity. This shift accelerates the learning process, as it draws the standard gradient nearer to the unit natural gradient, minimizing the effect of bias shift. Then, we applied the maximum pooling operation to the data, since we wanted to extract the most important features from the previous layer and reduce the computational cost by reducing the number of parameters in the network and the spatial size of the representation. The next layer is the dropout layer with the parameter 0.3. This means that we will randomly zero 30% of the input tensor. This is an effective technique for preventing the over-fitting problem. At this step, we want to convert the dimension of our tensor into 2 since we are dealing with the binary classification task. So, we added another fully connected layer which helped us to reduce the dimension from 512 to 2. The last layer is softmax which is used as the activation function in the output layer that predicts a multinomial probability distribution. Since we are dealing with the binary classification task, the softmax helped us to classify the current given context based on the probabilities calculated.

4.3 Effectiveness of our model

After building the model architecture, we used Adam optimizer with learning rate = $le-5$ which helped in quicker convergence and better generalization. For Adam, the learning rate adjusts during training, based on the historical gradient. It is sturdy against noisy gradients and memory-efficient which helped us a lot during training. After training for 50 epochs, we expect our model will have much higher accuracy than the original BERT model and the baseline.

5 EVALUATION

5.1 Performance Metrics

The evaluation metrics for our fine-tuned BERT model on the test dataset reveal promising results. The model achieved a precision of 0.99 for real news class and 0.98 for fake news class. This demonstrates that our model was highly precise, rarely misclassifying news articles.

The recall was also excellent with values of 0.99 for both classes. This suggests that the model was highly adept at detecting both real and fake news items in the dataset.

The F1 score, was also high for both classes, standing at 0.99. This suggests that the model maintained a robust balance between precision and recall during testing.

Our model achieved an overall accuracy of 0.99. Finally, the overall accuracy of the model was recorded at 0.99, indicating that in 99 percent of the cases, our model was able to correctly classify the news items as real or fake.

5.2 Training and Validation

The fine-tuning of the BERT model was performed over 50 epochs. In the last three epochs (48, 49, 50), the model training losses were 0.0305, 0.0307, and 0.0301, respectively, while the validation losses were 0.0303, 0.0281, and 0.0272.

This trend in loss metrics over the epochs shows that the model was learning effectively, with the validation loss decreasing over time. This indicates the model wasn't overfitting on the training data and was generalizing well to unseen data.

5.3 Comparison with Baseline model

The performance of our fine-tuned BERT model vastly exceeded that of the baseline model in detecting fake news. While the baseline model demonstrated commendable performance with an accuracy of 89.88%, precision of 88.76%, and an F1 score of 89.31%, our BERT model significantly outperformed it, achieving an impressive accuracy, precision, and F1 score all at 0.99.

Upon comparison, the reasons for BERT's superior performance become apparent. Firstly, the convolutional baseline model, despite its four-layer architecture, likely lacks the

depth and complexity required to understand the high-level semantic meaning in the text, an aspect crucial for tasks such as fake news detection. Secondly, the baseline model doesn't incorporate a built-in mechanism to comprehend the context or order of words, a feature integral to BERT through its transformer architecture.

Moreover, BERT's pre-training on a large text corpus allows it to capture generalized language understanding before being fine-tuned for specific tasks such as fake news detection. In contrast, the baseline model, learning from scratch, may not generalize as effectively.

	precision	recall	f1-score	support
0	0.99	0.99	0.99	3212
1	0.98	0.99	0.99	3414
accuracy			0.99	6626
macro avg	0.99	0.99	0.99	6626
weighted avg	0.99	0.99	0.99	6626

Figure 1: Evaluation

6 CONCLUSION

This study demonstrated the effectiveness of a fine-tuned BERT model in the challenging task of fake news detection. BERT's inherent capabilities of capturing high-level semantic meaning and understanding word context led to a significant improvement in detection accuracy, precision, and F1 score, reaching 0.99 for all metrics, a clear enhancement over traditional machine learning methods and the base CNN model. We also showcased a lemmatization technique that augments text preprocessing efficiency and boosts model performance. Our approach revealed BERT's potential in combating the fake news problem and opened new avenues for leveraging sophisticated models for similar text classification tasks. Future work will explore enhancing our model and applying it to other complex NLP tasks, providing further insights into how advanced NLP models like BERT can be utilized for real-world applications.

7 REFERENCES

[1] Mateusz Szczepański, Marek Pawlicki, Rafał Kozik & Michał Choraś, 2021, "New explainability method for BERT-based model in fake news detection", https://www.nature.com/articles/s41598-021-03100-6

[2]Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, Yoav Artzi, 2021 "REVISITING FEW-SAMPLE BERT FINE-TUNING", https://arxiv.org/pdf/2006.05987.pdf

[3] Keiron O'Shea, Ryan Nash, "An Introduction to Convolutional Neural Networks", 2015, https://arxiv.org/abs/1511.08458

[4]Chaitanya Malaviya, Shijie Wu, Ryan Cotterell, A Simple Joint Model for Improved Contextual Neural Lemmatization, NAACL. 2019, https://arxiv.org/abs/1904.02306