Technical Report: Final Project DS 5220: Supervised Machine Learning and Learning Theory

Team Members:
Cameron Jester & Rimsha Kayastha
Khoury College of Computer Sciences
Data Science Program
`jester.c@northeastern.edu, kayastha.r@northeastern.edu`

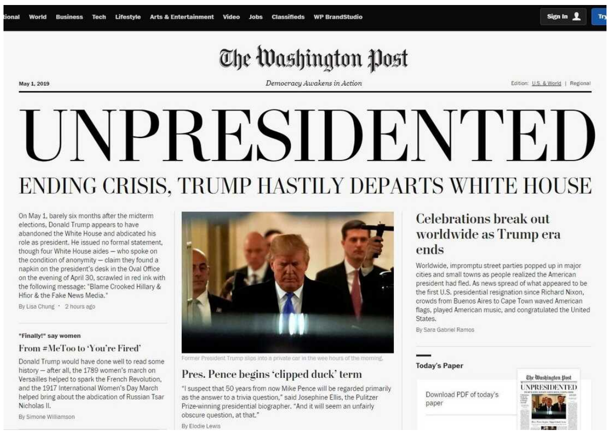
December 10, 2024

Contents

1 Abstract

In the ever-evoliving world of media consumption, news outlets, and news itself have gone through immense changes. With tools such as ChatGPT and the new marketing technique of "click-bait", the rise in fake news has been dramatic. With fake news lurking around every media outlet, consumers can become lost in what is real or not, and can lead to issues such as news distrust, the inability to make informed decisions, and even drive people against one another. The motiviation for this final project is to create a classification model to detect fake news to alleviate some of the issues prevously mentioned.

With 2024 being an election year, it is of utmost importance that citizens of the U.S. are not being fed misinformation. Candidates may use platforms to intentionally spread lies about others in order to gain more popularity for voters. This type of behavior can be combated with a news classificiation tool. Classifying news as real or fake can have lasting implications for the political world and beyond.


tional World Business Tech Lifestyle Arts & Entertainment Video Jobs Classifieds WP BrandStudio
Sign In
Try
Download PDF of today's paper

The Washington Post

May 1, 2019 — *Democracy Awakens in Action* — Edition: U.S. & World | Regional

UNPRESIDENTED

ENDING CRISIS, TRUMP HASTILY DEPARTS WHITE HOUSE

On May 1, barely six months after the midterm elections, Donald Trump appears to have abandoned the White House and abdicated his role as president. He issued no formal statement, though four White House aides — who spoke on the condition of anonymity — claim they found a napkin on the president's desk in the Oval Office on the evening of April 30, scrawled in red ink with the following message: "Blame Crooked Hillary & Hfior & the Fake News Media."

By Lisa Chung · 2 hours ago

"Finally!" say women

From #MeToo to 'You're Fired'

Donald Trump would have done well to read some history — after all, the 1789 women's march on Versailles helped to spark the French Revolution, and the 1917 International Women's Day March helped bring about the abdication of Russian Tsar Nicholas II.

By Simone Williamson

Former President Trump slips into a private car in the wee hours of the morning.

Pres. Pence begins 'clipped duck' term

"I suspect that 50 years from now Mike Pence will be regarded primarily as the answer to a trivia question," said Josephine Ellis, the Pulitzer Prize-winning presidential biographer. "And it will seem an unfairly obscure question, at that."

By Elodie Lewis

Celebrations break out worldwide as Trump era ends

Worldwide, impromptu street parties popped up in major cities and small towns as people realized the American president had fled. As news spread of what appeared to be the first U.S. presidential resignation since Richard Nixon, crowds from Buenos Aires to Cape Town waved American flags, played American music, and congratulated the United States.

By Sara Gabriel Ramos

Today's Paper

Figure 1: Counterfit Article Distributed by Activists, npr.org

As this supervised machine learning problem involves text-based data, it will become a robust Natural Language Processing project. Due to the nature of news articles, many have meanings embedded within long phrases or sentences. Given this, it is important to chose a model with self-attention mechanisms to learn the behavior of words in context. Therefore, BERT (Bidirectional Encoder Representations from Transformers) was chosen to complete this task as it a flexible, well-documented model to suit the needs of this project. After implementing BERT for encoding, two methods of classifying will be used: logistic Regression and K-Nearest-Neighbors. These two models will provide a straightford classification that will work well with the binary classes of the data.

As this dataset was downloaded from Kaggle.com, there was a few users who had used BERT to tokenize the data, but the projects listed build the model from the ground-up, whereas for this project the HuggingFace model architecture was used and then fine-tuned. Additionally, this projet deviated from others when it came to classifying the articles as real or fake as logistic regression and KNN were used.

Limitations in the approach of this project mainly revolved around time, given just 42 days between presentations, not excluding time off for holiday breaks, it was difficult to go as in-depth as desired for this project. A second major limitiation was the computational time and energy needed for this project. Unfortunatley, one of the group members' laptops could not locally handle the many parameters (340 million) of BERT, and their Google Collab ran out of RAM as well. These limitations made it difficult to work synchronously on the project, which was crucial given the limited time to complete everything.

To approach this as a supervised machine learning problem, it is critical to implement a model on labeled data.

2 Experiment Setup

For our project, the Title column was used to train models as opposed to the Text columns due to a 512 token limitation on BERT. Given the nature of news articles, it felt unnessisary to truncate the Text column when the Title can provide accurate classification. Therefore, the Title column was used to classify whether an article was real or fake.

2.1 Dataset Statistics

The dataset was obtained from Kaggle.com, a common data sharing platform. It was given a 10.0 usability score, meaning the data is complete, creditble, and compatible for anyone to use. The data contains 6256 news artciles and includes 4 columns: ’number’, ’title’, ’text’, and ’label’. The ’number’ column is simply a unique identifier and will be dropped for the purpose of analysis. The ’title’ column is for the title of articles, and conversely, the ’text’ column is for the body of each article. Finally, the ’label’ column is where each article is given a binary class of either ’real’ or ’fake’, assinging it as a real article or a fake article repsectivly. The Title and Text columns both include punctuation such, but not limited to; periods, commas, and exclamation points.

2.2 Implementation Techniques

To begin the technical aspect of this project, Visual Studio code was used as the IDE. Several packages needed to be installed which include: torch, transformers, scikit-learn, pandas, datasets, AutoModel, tensorflow, and tf-keras. Similarly, many packages were imported and can be seen below.

```python
import pandas as pd
from bs4 import BeautifulSoup
import re
import torch
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
# from datasets import Dataset
from transformers import AutoTokenizer # Hugging Face Transformers
from transformers import AutoModel
from transformers import TFAutoModelForSequenceClassification
from tensorflow.keras.optimizers import Adam
from sklearn.metrics import accuracy_score
```

Listing 1: Neccessary Packages to Import

After reading in the data, an encoder was initialized to transform the Label column as a One Hot Encoded column. The dataset was then split into a testing and training set, using an 80/20 split. A validation set was not created for this project. The tokenizer and models were initialized through the pretrained HuggingFace architechture previously installed.

Then, the preprocessed Title column was fed into the model as an input and the resulted into a tokenized sequence for the model. There are a few important parameters in this line of code that should be further explained. For example, returntensors = 'pt' means that the input should be compatible for pytorch. Following this is 'padding = True', is a parameter that is important for Transformers to conduct what is called batched processing. As padding was set to True, then all titles with less than 512 tokens (which is intialized later in this code line) are padded to get up to 512 tokens. Conversely, 'truncation = True' takes any text with over 512 tokens and reduces then until they meet the max length.

After loading the data, the "label" column was encoded using one-hot encoding for compatibility with the models. The dataset was split into training and testing sets using an 80/20 split. A validation set was not created for this project due to time constraints.

Preprocessing involved tokenizing the "Title" column using the HuggingFace tokenizer. Key parameters included: - **return_tensors='pt'**: Ensures compatibility with PyTorch. - **padding=True**: Enables batched processing by padding sequences shorter than 512 tokens. - **truncation=True**: Truncates sequences longer than 512 tokens to ensure uniform input length.

These preprocessing steps ensured that the data was correctly formatted for BERT's input requirements.

Following this, an optimizer called AdamW, which is popular from pytorch, was assigned with an learning rate of 5e $^{-}$5. The learning rate is set to how much each iteration should update the weights for each parameter.

Finally, a for loop was setup to pass over the data 3 times in which AdamW optimized through the data. It is important to note that in this step, it was set to zerograd() which means that backpropigation gradients did not acrue for each pass. As this part is only extracting the features for the model gradients do not need to be calculated, from there outputs (features) are extracted from the inputs we set and then cls embeddings from the models hidden states are set to a numpy array.

```python

tokenizer = AutoTokenizer.from_pretrained("distilbert-base-uncased-
    finetuned-sst-2-english")
model = AutoModel.from_pretrained("distilbert-base-uncased-finetuned-
    sst-2-english")
inputs = tokenizer(news_df['title'].to_list(), return_tensors='pt',
    padding=True, truncation=True, max_length=512)
optimizer = AdamW(model.parameters(), lr=5e-5)
model.train()
for epoch in range(10):
    optimizer.zero_grad()
    with torch.no_grad():
        outputs = model(**inputs)
        cls_embeddings = outputs.last_hidden_state[:, 0, :].numpy()
```

Listing 2: Implementing and Fine-Tuning BERT

2.3 Model Architecture

When deciding on the parameters for the model building, it was important to consider computational time. For epochs, 1, 3, and 5 were tested and timed. Respectively each took, 3.5 minutes, 9.5 minutes, and 16 minutes. Additionally, when selecting learning rate, 3 were tested for changes in model accuracy. The learning rates for the AdamW optimizer were $5e^-9$, $5e^-5$, $5e^-1$. with respective accuracies of .765, .770, and .760. Indicating $5e^-5$ has the highest accuracy by a slight amount.

3 Experiment Results

3.1 Main Results

After implementing BERT for tokenization, three classification models were selected to compare: Logistic Regression, KNN, and RandomForest. The test error across the different epochs for logistic regression were generally higher than that for KNN and RF, as can be compared in figure 2, figure 3, and figure 4

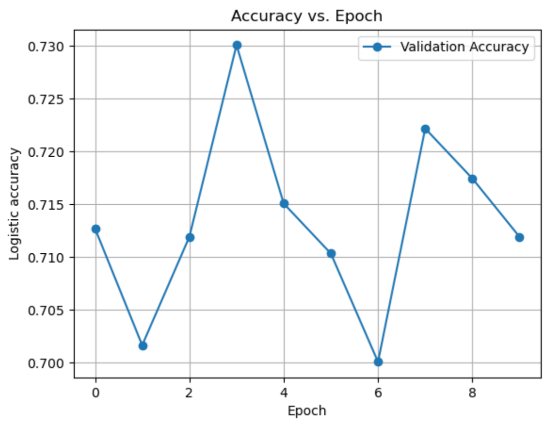


Figure 2: Test accuracies across epochs for logistic regression

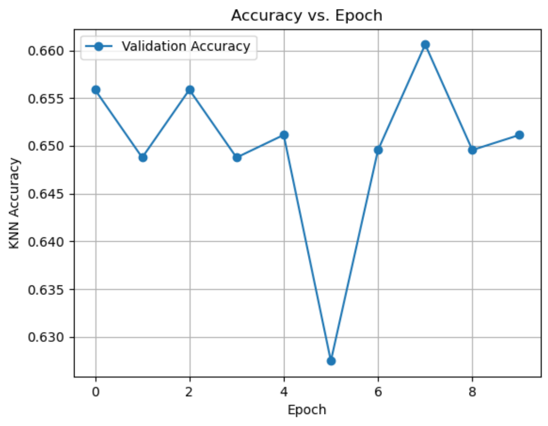


Figure 3: Test accuracies across epochs for KNN

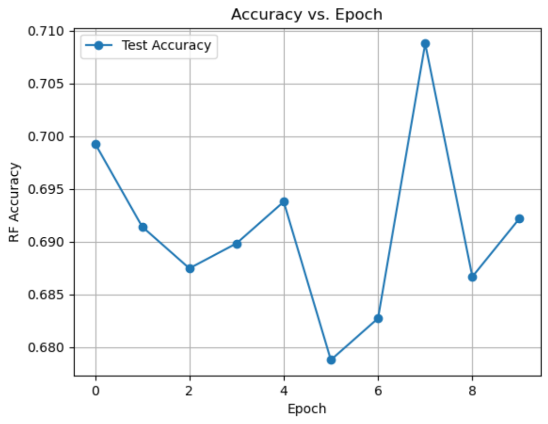


Figure 4: Test accuracies across epochs for RandomForest

For logistic regression, the most optimal epoch with the highest accuracy was 3, while for KNN and RF, it was 7.

For Logistic Regression, KNN, RandomForest at their respective optimal epoch size, testing accuracies were: 0.723,0.654 , and 0.709.

4 Discussion

This project used a distilled BERT uncased model combined with logistic regression and KNN to classify news titles as real or fake based on sentiment. Logistic regression outperformed KNN, likely due to its ability to handle the high-dimensional embeddings produced by BERT effectively, whereas KNN struggled with the cons of dimensionality. The optimal configuration of 3 training epochs and a learning rate of 5e-5 ensured the model reached peak accuracy possible. Distilled BERT, as a compact alternative to full-scale BERT, balanced computational efficiency and performance.

Despite promising results, the model's generalizability might be limited by the dataset size and fixed training epochs. Further exploration with larger datasets and additional training time could enhance robustness. KNN performance might benefit from dimensionality reduction techniques like PCA. Future work could also explore further fine-tuning of the distilled BERT model or the use of ensemble methods to boost accuracy. Expanding the model's application to full article text and integrating multimodal features such as metadata or images could further improve its utility for fake news detection. Additionally, testing accuracies, and cross-validation error, and AUC could be used to evaluate classification models.

5 Conclusion

This project successfully applied a distilled BERT uncased model combined with logistic regression and KNN classifiers to analyze sentiment in news titles for detecting fake news. The AdamW optimizer, with a carefully chosen learning rate of 5e-5, enabled efficient weight updates during training over three epochs, leveraging zero gradient accumulation for computational efficiency during feature extraction.

Future research could expand on this work by incorporating additional data, using more advanced fine-tuning techniques, and exploring ensemble approaches for improved accuracy and generalization. This project highlights the significance of combining advanced natural language processing models with practical implementation strategies for addressing contemporary challenges in information reliability.

At the conclusion of this project, team members gained an in-depth understanding of the real-life possibilities of machine learning models on different fields that affect the public directly. Subsequently, members have insights into self-attention mechanisms and what they mean.

6 References

References

TensorFlow. (n.d.). Classify text with Bert: *text: tensorflow.* Retrieved from `https://www.tensorflow.org/text/tutorials/classify_text_with_bert`

Logit Function. (n.d.). Retrieved from `1*058lacRrfNZBBdyHvVlg_w.png`

Mottesi, C. (2024a, October 21). *Bert vs GPT: Comparing the two most popular language models.* InvGate. Retrieved from `https://blog.invgate.com/gpt-3-vs-bert#:~:text=While%20both%20models%20are%20very,one%20used%20to%20train%20BERT`

National Public Radio. (2019, January 16). *Real fake news: Activists circulate counterfeit editions of The Washington Post.* NPR. Retrieved from `https://www.npr.org/2019/01/16/685857177/real-fake-news-activists-circulate-counterfeit-editions-of-the-washington-post`

Tech, J. S. (2022, March 31). *Fake or real news.* Kaggle. Retrieved from `https://www.kaggle.com/datasets/jillanisofttech/fake-or-real-news/data`

Dremio. (2024, August 28). *Transformers in NLP.* Retrieved from `https://www.dremio.com/wiki/transformers-in-nlp/#:~:text=Transformers%20have%20been%20used%20to,Parallelization%20leads%20to%20faster%20training`