Fake News Detection

CS3008: Deep Learning

Thribhuvan S (EC21B1093)
Chetan Vasista (EC21B1099)
Amar Rohith (EC21B1106)

Indian Institute of Information Technology, Design and Manufacturing, Kancheepuram

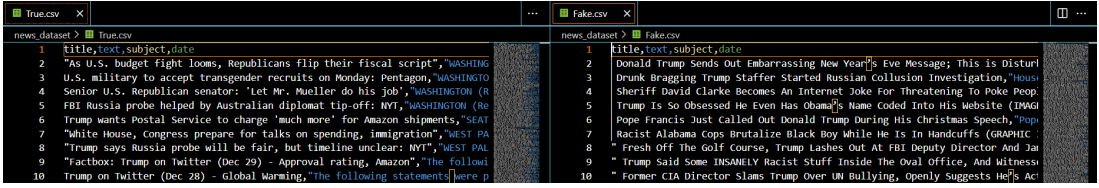
April 20, 2025

Overview

1. Data Preparation
2. Vocabulary and Dataset
3. Model Architecture
4. Positional Encoding
5. Training Process
6. Model Saving and Deployment
7. Evaluation Metrics
8. Results Visualization
9. Custom Input Prediction
10. Attention Visualization

Data Preparation

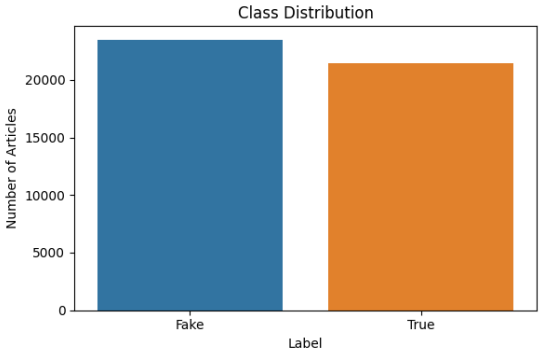
- True.csv, Fake.csv
- Combined ‘title‘ and ‘text‘ fields into a single ‘content‘ column.
- Text cleaning: lowercase conversion, trimming, removal of non-alphanumeric characters.



Sample of the dataset

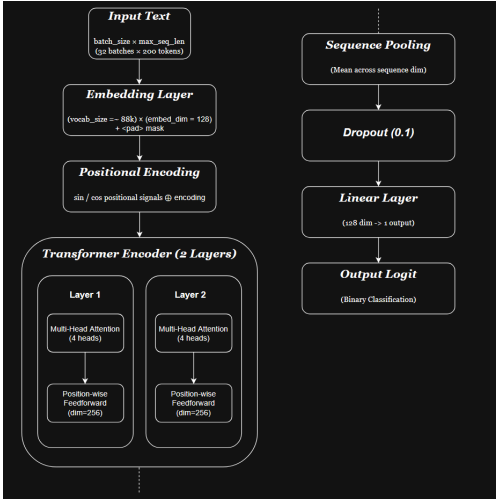
Vocabulary and Dataset

- Fake.csv slightly larger than True.csv
- Built vocabulary from training texts with minimum token frequency threshold of 2.
- Defined custom ‘NewsDataset‘ class to tokenize, numericalize, and pad sequences up to a maximum sequence length of 200.
- Created PyTorch ‘DataLoader‘s for train, validation, and test splits (60-20-20 split).



Model Architecture

- Embedding layer: Maps token indices to 128-dimensional vectors.
- Positional encoding added to embeddings for sequential information.
- Transformer encoder with 2 layers, 4 attention heads, and 256-dimensional feedforward network.
- Classification head: global average pooling + dropout + linear layer to output logit.

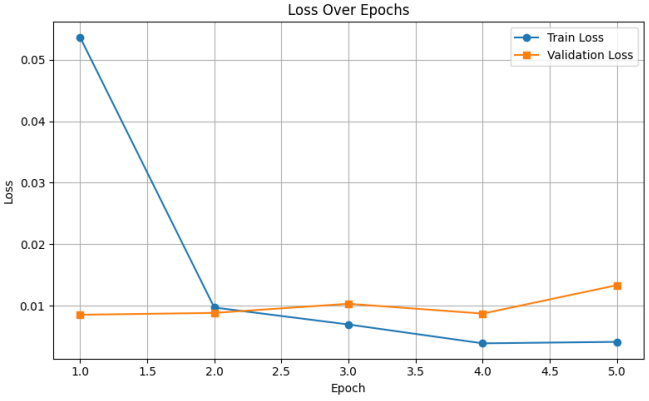


Positional Encoding

- Implemented sinusoidal positional encoding.
- Adds unique position-based vectors to embeddings to retain token order.
- No additional parameters (fixed during training).

Training Process

- Loss: Binary Cross-Entropy with Logits *(BCEWithLogitsLoss)*.
- Optimizer: Adam with learning rate 1×10^{-3}.
- Trained for 5 epochs, tracking training and validation loss.



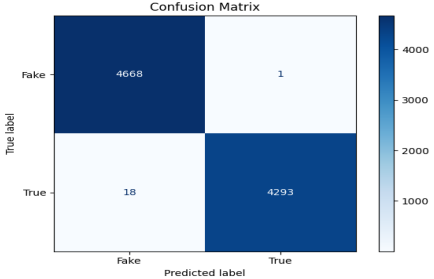
Model Saving and Deployment

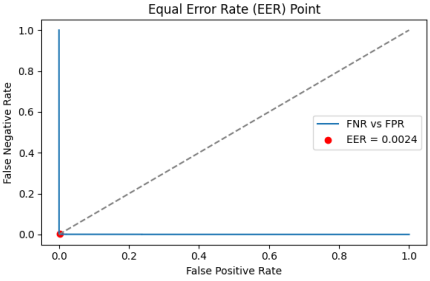
- Saved trained model state to ‘transformer_fake_news_model.pth‘.
- Vocabulary serialized to ‘vocab.pkl‘ using Python ‘pickle‘.
- Ready for inference: Load model and vocab, run prediction on custom text input.

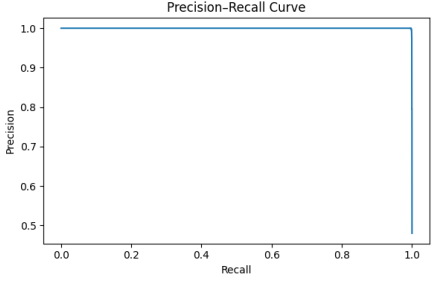
Evaluation Metrics

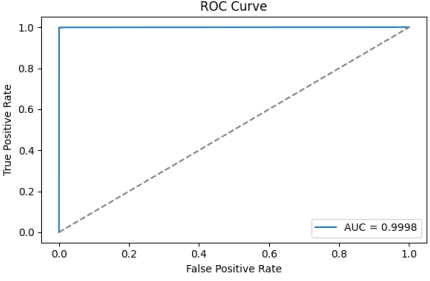
- Accuracy: Correct classifications over total samples = 0.9979
- AUC: Area under ROC curve for model discrimination = 0.9998
- Precision: True positives divided by predicted positives = 0.9998
- Equal Error Rate (EER): Point where False Positive Rate equals False Negative Rate = 0.0024 at threshold 0.0131

Results Visualization









Custom Input Prediction

- Cleans and tokenizes input text.
- Pads/truncates to max sequence length.
- Outputs predicted class (‘True‘ or ‘Fake‘) and confidence score.

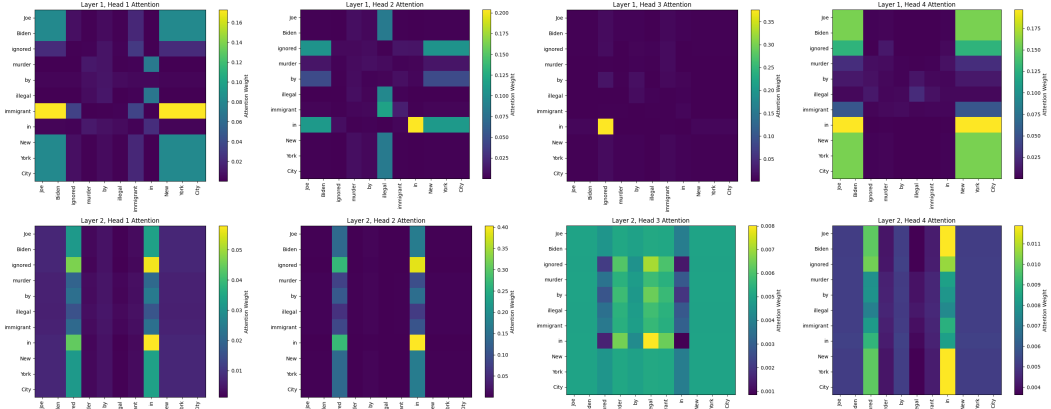
```python
model = TransformerClassifier(
    vocab_size=vocab_size,
    embed_dim=128,
    nhead=4,
    num_encoder_layers=2,
    dim_feedforward=256,
    dropout=0.1,
    max_seq_len=max_seq_len,
    num_classes=1
)

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.load_state_dict(torch.load(model_save_path, map_location=device))
model = model.to(device)
model.eval()

if __name__ == "__main__":
    custom_text = "Joe Biden ignored murder by illegal immigrant in New York City" # Change this prompt
    predicted_label, confidence = predict_custom_input(custom_text, model, vocab, max_seq_len, device)
    print("Custom Input Prediction: ")
    print("(Only works on US geopolitics and internal politics between 2016 and 2017 due to the nature of the dataset)")
    print("-----------------------")
    print(f"Input Text       : {custom_text}")
    print(f"Predicted Label  : {predicted_label}")
    print(f"Confidence Score (0 for Fake, 1 for True): {confidence:.4f}")
```

✓ 4.1s

```
Custom Input Prediction:
(Only works on US geopolitics and internal politics between 2016 and 2017 due to the nature of the dataset)
-----------------------
Input Text       : Joe Biden ignored murder by illegal immigrant in New York City
Predicted Label  : Fake
Confidence Score (0 for Fake, 1 for True): 0.0001
```

Attention Visualization

- Patched multi-head attention to extract weight matrices for selected layer and head to interpret model focus on token relationships.



Thank You