National College of Ireland

Configuration Manual

MSc Research Project
MSc in Artificial Intelligence

Mohammad Shehnaj
Student ID: 23305762

School of Computing
National College of Ireland

Supervisor: Abdul Razzaq

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Mohammad Shehnaj
Student ID:	23305762
Programme:	MSc in Artificial Intelligence **Year:** 2025
Module:	MSc Research Project – Configuration Manual
Lecturer:	Abdul Razzaq
Submission Due Date:	15/09/2025
Project Title:	A Comparative Study on the Impact of Input Length on Transformer Model Performance in Misinformation Classification
Word Count:	2107 words **Page Count:** 19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Mohammad Shehnaj

Date: 15/09/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Mohammad Shehnaj
23305762

1 Introduction

1.1 Project summary

This configuration manual explains how to set up, run and fully reproduce the experiments for the project "A Comparative Study on the Impact of Input Length on Transformer Model Performance in Misinformation Classification". It is a step-by-step guide focused on environment setup, datasets, code execution, model training, evaluation and troubleshooting to replicate results on Kaggle with the same configuration.

The system evaluates how input text length - Short, Medium, Long and Hybrid impacts both classification effectiveness (Accuracy, Precision, Recall, F1, ROC-AUC) and computational efficiency (training time, inference latency, memory, throughput) across multiple transformer models such as BERT, RoBERTa, Longformer, BigBird and LLaMA 2. (Devlin et al., 2019; Liu et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020; Touvron et al., 2023)

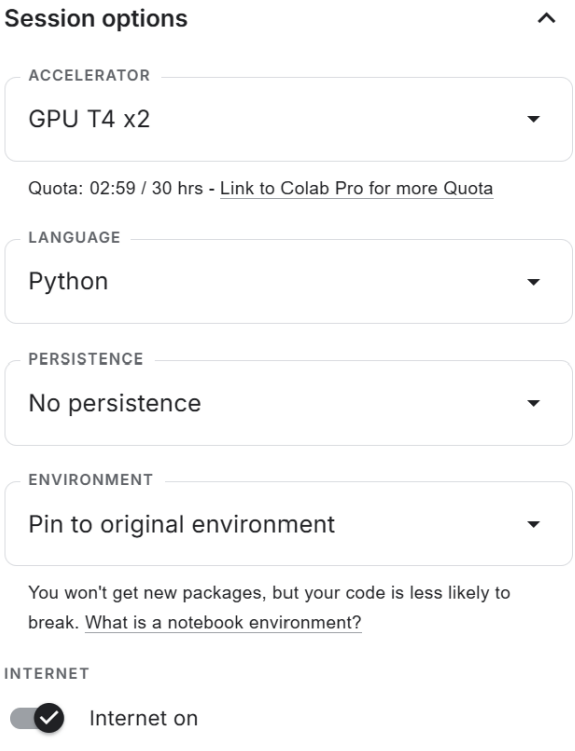
Core datasets include US Elections 2024, WELFake, LIAR and FNC-1 (US Elections 2024 Fake News Dataset, n.d.; WELFake, n.d.; Wang, 2017) and the full pipeline covers preprocessing, tokenization, length binning, model-aware training per bin, evaluation with visualizations and paired t-tests / Wilcoxon for statistical validation.

2 System Requirements

2.1 Recommended Runtime Environment

This project is designed and tested primarily on Kaggle Notebooks (Kaggle, n.d.).

- Runtime type: GPU
- GPU type: Two NVIDIA T4 GPUs (Kaggle's "GPU T4 × 2" setting) (NVIDIA, n.d.)
- CPU: 4 vCPUs
- RAM: 16 GB
- Disk space: Minimum 20 GB free (datasets + model checkpoints + intermediate cache files).



2.2 Software & Dependencies

- Python environment: Python 3.10 (Kaggle default)
- Core dependencies are specified in requirements.txt and must be installed before running the notebooks.
- Here's the exact list used in the project:

transformers==4.38.2
datasets==2.18.0
evaluate==0.4.1
scikit-learn==1.4.0
numpy==1.26.3
pandas==2.2.0
matplotlib==3.8.2
seaborn==0.13.2
torch==2.2.1
sentencepiece==0.1.99
bitsandbytes==0.42.0
accelerate==0.27.2
peft==0.10.0

2.3 Dataset Storage Requirements

The datasets together occupy approximately 5–6 GB when uncompressed.

- Stored in "/kaggle/input/" automatically when using Kaggle datasets.
- Local replication: store datasets under a data/ folder, maintaining the exact subfolder structure given in Section 3.

2.4 Internet Access Requirement

Kaggle's "Internet" option must be enabled in Notebook Settings for:

- Downloading models from Hugging Face (transformers auto-download) (Hugging Face, n.d.).
- Fetching missing tokenizer files (Hugging Face, n.d.).
- Installing any additional packages not preinstalled in Kaggle (Kaggle, n.d.).

2.5 Model Hardware Requirements

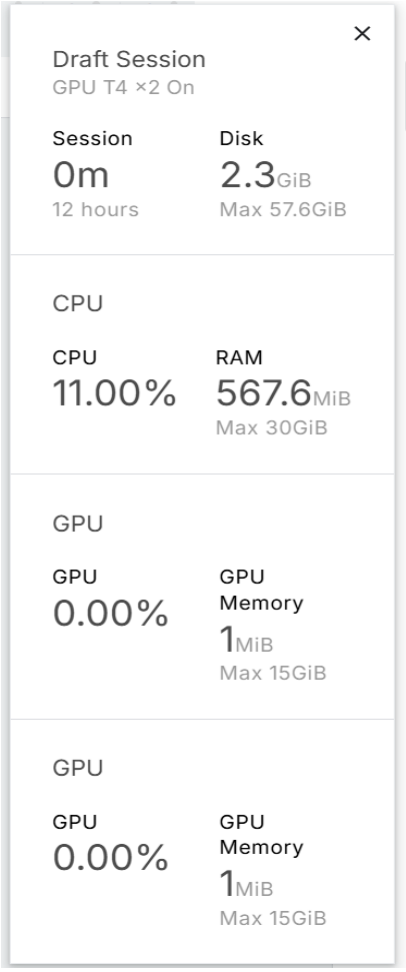
Model	Minimum VRAM (per GPU)	Observations
BERT/ RoBERTa	~8 GB	Runs on single T4 easily.
Longformer	~12 GB	Can run on single T4, faster with two.
BigBird	~14 GB	Requires gradient accumulation for long bins.
LLAMA 2 (4-bit)	~12 GB	Needs bitsandbytes + peft for LoRA fine-tuning.

3 Installation & Environment Setup

3.1 Environment Setup on Kaggle

The project was performed on Kaggle Notebook. Ensure the following Notebook Settings before starting:

- Accelerator: GPU
- GPU type: T4 × 2 (NVIDIA, n.d.)
- Internet: ON (Kaggle, n.d.)



3.2 Cloning Project & Setting Working Directory

To run on Kaggle with uploaded code (Kaggle, n.d.):

- Open a new notebook and click on "File", "Import Notebook" and then select the notebook file.
- To import the datasets, on right panel, under "Input" section, click on "Upload", upload the datasets folder in it.
- Once it is uploaded, start the session and execute it.

3.3 Installing Dependencies

On Kaggle, install all required packages using:

!pip install -r requirements.txt

This ensures all libraries like transformers, datasets and accelerate (Hugging Face, n.d.), scikit-learn (Scikit-learn Developers, n.d.), numpy (NumPy Developers, n.d.), pandas (Pandas Development Team, n.d.), seaborn (Seaborn Developers, n.d.), torch (PyTorch Developers, n.d.) are available in the environment.

3.4 Dataset Download & Placement

3.4.1 From Kaggle Datasets

The datasets are added via Kaggle Dataset Import in Notebook Settings (Kaggle, n.d.).

1. Click Add Data → Search dataset name → Add to Notebook.
2. Kaggle will mount it to /kaggle/input/<dataset-name>/.

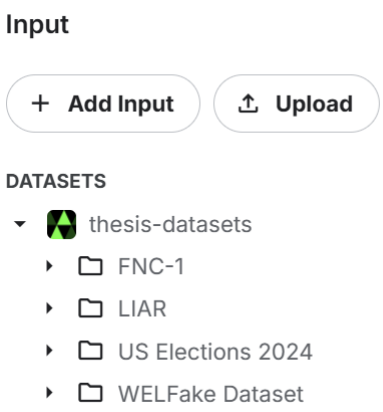


Figure 1: Screenshot of "Add Data" panel in Kaggle

3.5 Folder Structure

When setup is complete, your Kaggle working directory should look like this:

```
/kaggle/working/
├── Thesis_Code.ipynb
├── requirements.txt
├── data/
│   ├── FNC-1/
│   ├── LIAR /
│   ├── US Elections 2024/
│   └── WELFake Dataset/
├── models/              # Saved fine-tuned models
└── outputs/             # Evaluation results & plots
```

4 Dataset Preparation

4.1 Overview

This section standardizes all datasets to a common schema, cleans text, computes token lengths, bins samples into Short / Medium / Long / Hybrid and creates train/val/test (70/15/15) splits with label balance. Length thresholds used here match the thesis: Short ≤ 145, Medium 146–387, Long > 387 and Hybrid is a uniform mix across bins.

Core datasets used are:

- FNC-1 (Riedel et al., 2017)
- LIAR (Wang, 2017)
- US Elections 2024 (US Elections 2024 Fake News Dataset, n.d.)
- WELFake (WELFake, n.d.)

4.2 Load datasets

Paste this in a new cell and run. Adjust paths if you've added datasets via Kaggle's Add Data panel.

```
# Load Dataset 1: WELFake Dataset
welfake_path = "/kaggle/input/datasets-thesis/WELFake Dataset/WELFake Dataset/WELFake_Dataset.csv"
df1 = pd.read_csv(welfake_path)
print("Dataset 1:")
display(df1.head())
```

```
Dataset 1:
```

	Unnamed: 0	title	text	label
0	0	LAW ENFORCEMENT ON HIGH ALERT Following Threat...	No comment is expected from Barack Obama Membe...	1
1	1	NaN	Did they post their votes for Hillary already?	1
2	2	UNBELIEVABLE! OBAMA'S ATTORNEY GENERAL SAYS MO...	Now, most of the demonstrators gathered last ...	1
3	3	Bobby Jindal, raised Hindu, uses story of Chri...	A dozen politically active pastors came here f...	0
4	4	SATAN 2: Russia unvelis an image of its terrif...	The RS-28 Sarmat missile, dubbed Satan 2, will...	1

```
# Load Dataset 2: LIAR

df_train = pd.read_csv("/kaggle/input/datasets-thesis/LIAR/LIAR/train.tsv", sep='\t', header=None)
df_test = pd.read_csv("/kaggle/input/datasets-thesis/LIAR/LIAR/test.tsv", sep='\t', header=None)
df_valid = pd.read_csv("/kaggle/input/datasets-thesis/LIAR/LIAR/valid.tsv", sep='\t', header=None)

# Combine them
df2 = pd.concat([df_train, df_test, df_valid], axis=0).reset_index(drop=True)
print("LIAR Combined Shape:", df2.shape)
display(df2.head())
```

```
LIAR Combined Shape: (12791, 14)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	2635.json	false	Says the Annies List political group supports ...	abortion	dwayne-bohac	State representative	Texas	republican	0.0	1.0	0.0	0.0	0.0	a mailer
1	10540.json	half-true	When did the decline of coal start? It started...	energy,history,job-accomplishments	scott-surovell	State delegate	Virginia	democrat	0.0	0.0	1.0	1.0	0.0	a floor speech.
2	324.json	mostly-true	Hillary Clinton agrees with John McCain "by vo...	foreign-policy	barack-obama	President	Illinois	democrat	70.0	71.0	160.0	163.0	9.0	Denver
3	1123.json	false	Health care reform legislation is likely to ma...	health-care	blog-posting	NaN	NaN	none	7.0	19.0	3.0	5.0	44.0	a news release
4	9028.json	half-true	The economic turnaround started at the end of ...	economy,jobs	charlie-crist	NaN	Florida	democrat	15.0	9.0	20.0	19.0	2.0	an interview on CNN

```
# Load Dataset 3: US Elections
df3 = pd.read_csv("/kaggle/input/datasets-thesis/US Elections 2024/US Elections 2024/fake_news_elections_2024.csv")
print("Dataset 3:")
display(df3.head())
```

```
Dataset 3:
```

	text	source	date_published	keyword_category	outlet
0	Key Points: 53 percent of people live below t...	https://news.google.com/rss/articles/CBMiRmh0d...	10/20/2023	Race/Ethnicity-Based Terms	AZ Animals
1	Our experts answer readers' investing question...	https://news.google.com/rss/articles/CBMicWh0d...	10/20/2023	Race/Ethnicity-Based Terms	Business Insider
2	Review: 'Cyberpunk 2077: Phantom Liberty' give...	https://news.google.com/rss/articles/CBMiK2h0d...	10/21/2023	Race/Ethnicity-Based Terms	Arab News
3	From Del. Rip Sullivan's BLUE DOMINION PAC: W...	https://news.google.com/rss/articles/CBMihwFod...	10/20/2023	Race/Ethnicity-Based Terms	Blue Virginia
4	Brief Overview of "Bastard Out Of Carolina" "...	https://news.google.com/rss/articles/CBMiS2h0d...	10/20/2023	Race/Ethnicity-Based Terms	CitizenSide

```python
# Load Dataset 4: FNC-1

stance_path = "/kaggle/input/datasets-thesis/FNC-1/FNC-1/fnc-1-master/train_stances.csv"
body_path = "/kaggle/input/datasets-thesis/FNC-1/FNC-1/fnc-1-master/train_bodies.csv"

df_stance = pd.read_csv(stance_path)
df_body = pd.read_csv(body_path)
df4 = pd.merge(df_stance, df_body, how='left', left_on='Body ID', right_on='Body ID')
print("Merged FNC-1 shape:", df4.shape)
display(df4.head())
```

```
Merged FNC-1 shape: (49972, 4)
```

	Headline	Body ID	Stance	articleBody
0	Police find mass graves with at least '15 bodi...	712	unrelated	Danny Boyle is directing the untitled film\n\n...
1	Hundreds of Palestinians flee floods in Gaza a...	158	agree	Hundreds of Palestinians were evacuated from t...
2	Christian Bale passes on role of Steve Jobs, a...	137	unrelated	30-year-old Moscow resident was hospitalized w...
3	HBO and Apple in Talks for $15/Month Apple TV ...	1034	unrelated	(Reuters) - A Canadian soldier was shot at the...
4	Spider burrowed through tourist's stomach and ...	1923	disagree	Fear not arachnophobes, the story of Bunbury's...

Figure 2: Code cell that loads all datasets.

4.3 Standardize + concatenate datasets

```python
# Load all 4 cleaned datasets
df_wel  = pd.read_csv(os.path.join(cleaned_path, "cleaned_welfake.csv"))
df_liar = pd.read_csv(os.path.join(cleaned_path, "cleaned_liar.csv"))
df_us   = pd.read_csv(os.path.join(cleaned_path, "cleaned_us_elections_2024.csv"))
df_fnc1 = pd.read_csv(os.path.join(cleaned_path, "cleaned_fnc1.csv"))

# Combine all datasets
combined_df = pd.concat([df_us, df_wel, df_liar, df_fnc1], axis=0).reset_index(drop=True)
print("Combined dataset shape:", combined_df.shape)
display(combined_df.head())
print(combined_df['label'].value_counts())
```

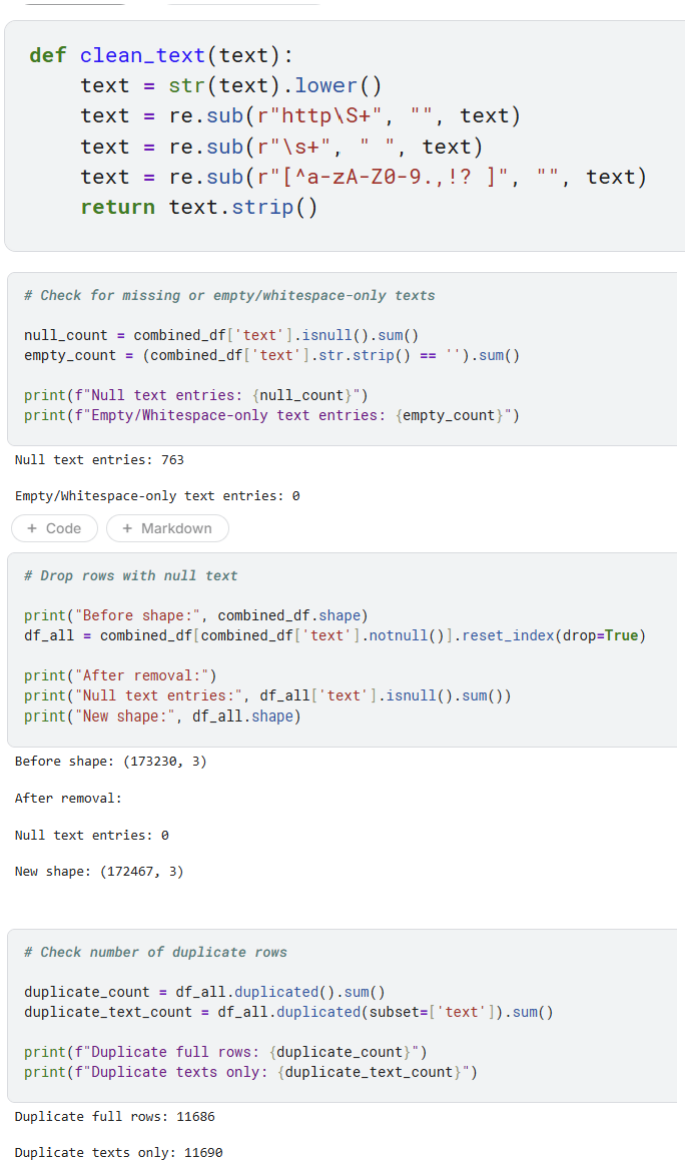
```
Combined dataset shape: (173230, 3)
```

	text	label	source_dataset
0	key points 53 percent of people live below the...	0	US_Elections_2024
1	our experts answer readers investing questions...	0	US_Elections_2024
2	review cyberpunk 2077 phantom liberty gives a ...	1	US_Elections_2024
3	from del. rip sullivans blue dominion pac welc...	0	US_Elections_2024
4	brief overview of bastard out of carolina bast...	1	US_Elections_2024

Figure 3: Head of the unified dataframe (text, label, source)

4.4 Text cleaning (lightweight, model-friendly)

Match your notebook's cleaning rules (keep URLs/emojis removal consistent with your runs).



```
def clean_text(text):
    text = str(text).lower()
    text = re.sub(r"http\S+", "", text)
    text = re.sub(r"\s+", " ", text)
    text = re.sub(r"[^a-zA-Z0-9.,!? ]", "", text)
    return text.strip()
```

```
# Check for missing or empty/whitespace-only texts

null_count = combined_df['text'].isnull().sum()
empty_count = (combined_df['text'].str.strip() == '').sum()

print(f"Null text entries: {null_count}")
print(f"Empty/Whitespace-only text entries: {empty_count}")
```

```
Null text entries: 763

Empty/Whitespace-only text entries: 0
```

+ Code + Markdown

```
# Drop rows with null text

print("Before shape:", combined_df.shape)
df_all = combined_df[combined_df['text'].notnull()].reset_index(drop=True)

print("After removal:")
print("Null text entries:", df_all['text'].isnull().sum())
print("New shape:", df_all.shape)
```

```
Before shape: (173230, 3)

After removal:

Null text entries: 0

New shape: (172467, 3)
```

```
# Check number of duplicate rows

duplicate_count = df_all.duplicated().sum()
duplicate_text_count = df_all.duplicated(subset=['text']).sum()

print(f"Duplicate full rows: {duplicate_count}")
print(f"Duplicate texts only: {duplicate_text_count}")
```

```
Duplicate full rows: 11686

Duplicate texts only: 11690
```

Figure 4: Sample of cleaned text

4.5 Assign length bins

Use your thresholds: Short ≤145, Medium 146–387, Long >387. Hybrid will be created later as a uniform mix.

```
# Categorize into short / medium / long categories
def categorize_length(wc):
    if wc <= 145:
        return 'short'
    elif wc > 145 and wc <= 387:
        return 'medium'
    else:
        return 'long'

df_all['length_category'] = df_all['word_count'].apply(categorize_length)
print(df_all['length_category'].value_counts())
display(df_all[['text', 'word_count', 'length_category']].head())
```

```
length_category

medium    53673

long      53585

short     53523

Name: count, dtype: int64
```

	text	word_count	length_category
0	key points 53 percent of people live below the...	25	short
1	our experts answer readers investing questions...	68	short
2	review cyberpunk 2077 phantom liberty gives a ...	88	short
3	from del. rip sullivans blue dominion pac welc...	57	short
4	brief overview of bastard out of carolina bast...	129	short

Figure 5: Counts per bin and per source

4.6 Create a Hybrid set

Hybrid is a 1:1:1 mixture of short, medium and long

```
# Split dataset into short, medium, long datasets
df_short = df_all[df_all['length_category'] == 'short']
df_medium = df_all[df_all['length_category'] == 'medium']
df_long = df_all[df_all['length_category'] == 'long']

# Determine minimum count across the three
min_len = min(len(df_short), len(df_medium), len(df_long))
per_class = min_len // 3

# Sample equally from each
df_hybrid = pd.concat([
    df_short.sample(per_class, random_state=42),
    df_medium.sample(per_class, random_state=42),
    df_long.sample(per_class, random_state=42),],
    ignore_index=True).sample(frac=1, random_state=42).reset_index(drop=True)

# Label this new category as 'hybrid'
df_hybrid['length_category'] = 'hybrid'
df_all_combined = pd.concat([df_all, df_hybrid], ignore_index=True)
print(df_all_combined['length_category'].value_counts())
display(df_all_combined.head())
```

```
length_category

medium    53673

long      53585

short     53523

hybrid    53523

Name: count, dtype: int64
```

Figure 6: Hybrid bin count

4.7 Balance labels within each bin

This mirrors your thesis's emphasis on balanced evaluation. If needed, downsample the majority class per bin (Scikit-learn Developers, n.d.).

```
Final Distribution across length categories:

length_category

hybrid    39567

short     39519

long      39378

medium    39226

Name: count, dtype: int64


Label distribution per category:

label                0      1

length_category

hybrid           19981  19586

long             19885  19493

medium           19809  19417

short            19957  19562


Overall label distribution:

label

0    79632

1    78058

Name: count, dtype: int64
```

Figure 7: Value counts after balancing per bin

4.8 Train/Val/Test split (70/15/15) - stratified by label within each bin

Splitting per bin preserves distribution for each subsequent model training run (Scikit-learn Developers, n.d.).

```python
def stratified_group_split(df, test_size=0.15, random_state=42, group_cols=["source_dataset", "length_category"]):
    """
    Stratified split that ensures both classes exist in each (source_dataset, length_category) group in both train and test.
    """
    df["group"] = df[group_cols].astype(str).agg("_".join, axis=1)
    grouped = df.groupby("group")

    train_df_list = []
    test_df_list = []

    for group_name, group_data in grouped:
        if group_data["label"].nunique() < 2:
            train_part, test_part = train_test_split(group_data, test_size=test_size, random_state=random_state)
        else:
            train_part, test_part = train_test_split(group_data, test_size=test_size, stratify=group_data["label"], random_state=random_state)
        train_df_list.append(train_part)
        test_df_list.append(test_part)

    train_df = pd.concat(train_df_list).sample(frac=1, random_state=random_state).reset_index(drop=True)
    test_df = pd.concat(test_df_list).sample(frac=1, random_state=random_state).reset_index(drop=True)

    df.drop(columns=["group"], inplace=True, errors="ignore")
    return train_df, test_df
```

Figure 8: Printout of split sizes per bin & label

4.9 Save preprocessed files

- All the csv files are saved in output folder.
- These files are to be downloaded one by one before session ends.
- Once the session ends, the files will be discarded and you will have to execute the code again.

5 Code Configuration

5.1 Project layout

Your Kaggle notebook is the single-entry point. Keep paths and names consistent so training and evaluation cells find everything automatically.

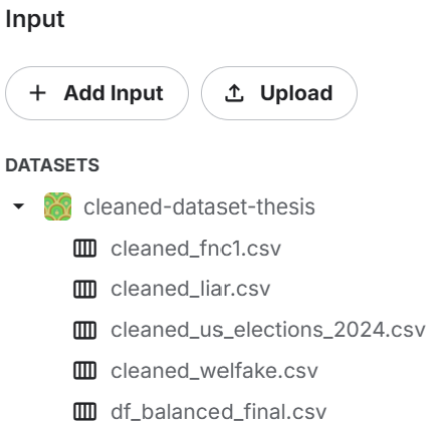


Figure 9: CSV files and balanced dataset for further processing

5.2 Tokenizer & truncation strategy

- Baseline models BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) are limited to 512 tokens → use smart truncation or sliding window for "long" bin.
- Longformer (Beltagy et al., 2020) and BigBird (Zaheer et al., 2020) accept long sequences (4k–8k) → set higher max_length.
- LLaMA 2 context depends on checkpoint (Touvron et al., 2023); for classification, keep max_length modest unless quantized/LoRA-tuned.

6 Execution Workflow

6.1 Data Loading

- Open the Kaggle notebook containing the project code.
- Ensure the datasets are available in the Kaggle environment.

6.2 Preprocessing & Input Length Categorization

- Text preprocessing removes noise (punctuation, HTML tags, extra spaces).
- Tokenization is done using Hugging Face tokenizer (AutoTokenizer from_pretrained()) matching the selected transformer model.
- Input text is categorized into length groups:
    - Short: <= 145
    - Medium: 146 - 387

- Long: > 387
- Hybrid: Combination of all three (short, medium and long)

6.3 Model Selection & Training

- Models implemented: BERT, RoBERTa, Longformer, BigBird and LlaMA 2.
- Select the model by updating the model_name variable in the code.
- Fine-tuning is performed using Hugging Face Trainer API or custom training loops.
- The project supports multi-length category training by passing filtered datasets per length type.

```
Epoch 1
100%|██████████| 11115/11115 [36:34<00:00,  5.07it/s]
Training Loss: 7504.5361

Epoch 2
100%|██████████| 11115/11115 [36:47<00:00,  5.03it/s]
Training Loss: 6781.4571

Epoch 3
100%|██████████| 11115/11115 [36:47<00:00,  5.03it/s]
Training Loss: 6512.0295

Training complete and model saved.
```

, , , [1938/1938 1:12:57, Epoch 3/3] ,

, ,

Epoch	Training Loss	Validation Loss
1	0.228100	0.095316
2	0.078900	0.086433
3	0.037400	0.093341

6.4 Evaluation & Metrics

- Models are evaluated on Accuracy, Precision, Recall, F1-score, ROC-AUC, Inference Time, Training Time, Memory Usage and Throughput.
- Confusion matrix and ROC curves are generated for each model and length category.
- Evaluation metrics for each model, dataset source-wise, length category wise has been generated.

```
Metrics:
Accuracy   : 0.8110
Precision  : 0.7851
Recall     : 0.8511
F1 Score   : 0.8168
AUC Score  : 0.9035
Inference Time (s): 830.39
Throughput (samples/sec): 11.90
Memory Usage (MB): 1019.73

Classification Report:
              precision    recall  f1-score   support

        Real       0.84      0.77      0.80      4990
        Fake       0.79      0.85      0.82      4890

    accuracy                           0.81      9880
   macro avg       0.81      0.81      0.81      9880
weighted avg       0.81      0.81      0.81      9880
```

7 Output & Results

This section documents the expected outputs when running the project successfully. All results are generated directly from the Kaggle notebook after training and evaluation.

7.1 Performance Metrics Table

For each model (BERT, RoBERTa, Longformer, BigBird, LlaMA 2) and each input length category (Short, Medium, Long, Hybrid), the evaluation produces:

- Accuracy
- Precision
- Recall
- F1-score
- ROC-AUC

These values are presented in a comparative table for easy analysis.

Table 4: Performance on Short-Length Inputs

Model	Accuracy	F1-score	ROC-AUC
RoBERTa	**0.8206**	**0.8337**	**0.9180**
BERT	0.8161	0.8320	0.9124
BigBird	0.8110	0.8168	0.9035
LLaMA	0.6842	0.7057	0.7501
Longformer	0.5771	0.4757	0.5943

Table 5: Performance on Medium-Length Inputs

Model	Accuracy	F1-score	ROC-AUC
RoBERTa	**0.9555**	**0.9562**	**0.9941**
BigBird	0.9540	0.9539	0.9907
BERT	0.9476	0.9477	0.9889
LLaMA	0.5971	0.4971	0.6475
Longformer	0.5049	0.0000	0.5045

Table 6: Performance on Long-Length Inputs

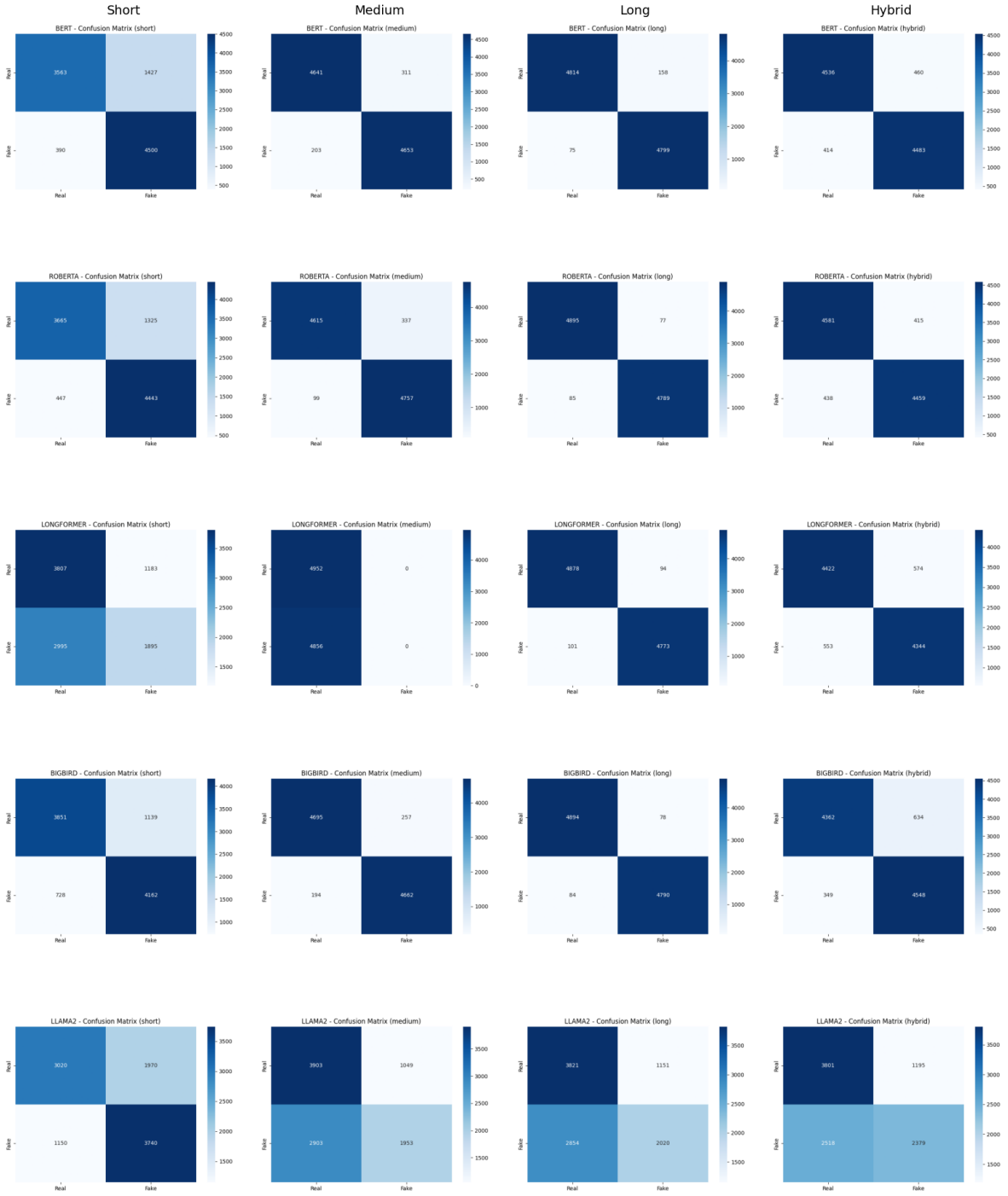
Model	Accuracy	F1-score	ROC-AUC
RoBERTa	**0.9835**	**0.9834**	**0.9990**
BigBird	**0.9835**	**0.9834**	0.9984
Longformer	0.9802	0.9800	0.9981
BERT	0.9763	0.9763	0.9971
LLaMA	0.5932	0.5022	0.6433

Table 7: Performance on Hybrid-Length Inputs

Model	Accuracy	F1-score	ROC-AUC
RoBERTa	**0.9138**	**0.9127**	**0.9796**
BERT	0.9117	0.9112	0.9768
BigBird	0.9006	0.9025	0.9706
Longformer	0.8861	0.8852	0.9553
LLaMA	0.6247	0.5617	0.6803

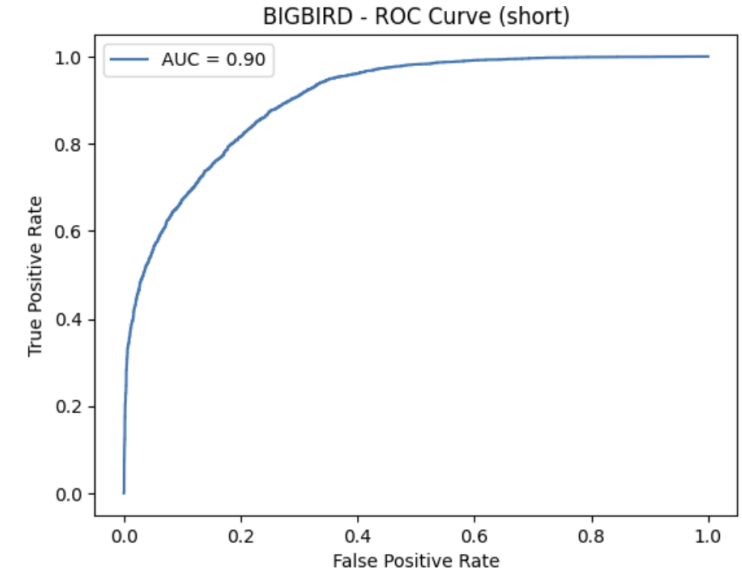
7.2 Confusion Matrix

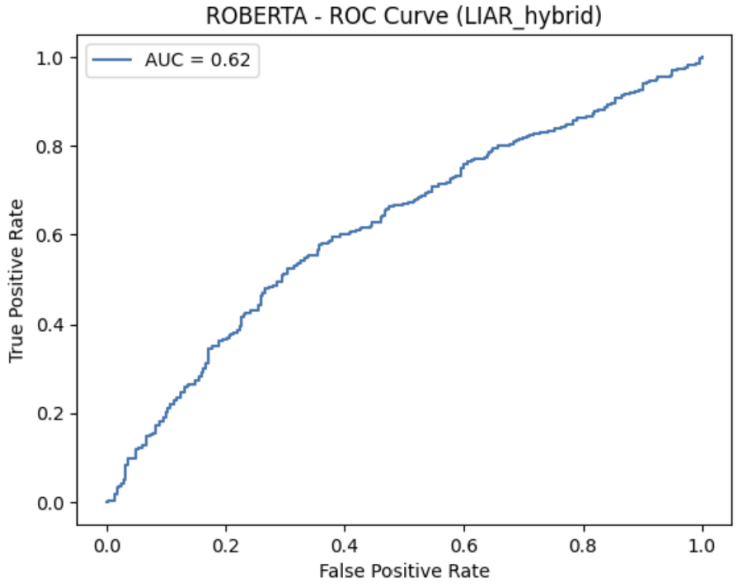
- A confusion matrix is generated for each model and length category.
- This visualizes correct vs. incorrect classifications, helping assess class-level performance.



7.3 ROC Curve

- ROC curves show the trade-off between True Positive Rate and False Positive Rate across thresholds.
- AUC (Area Under Curve) is included for quantitative evaluation.





7.4 Computational Efficiency Metrics

For each model:

- Training Time (seconds)
- Inference Time per 100 samples (seconds)
- Memory Usage (MB)
- Throughput (samples/sec)

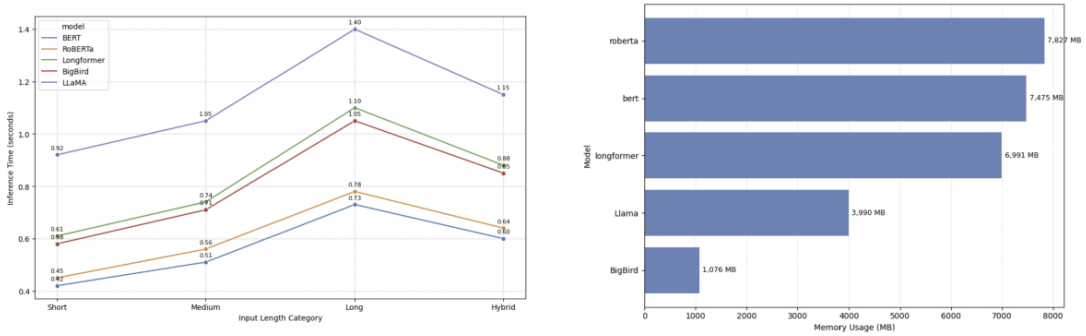


Figure 9: Computational efficiency metrics: (a) Inference time trends and (b) memory usage ranking.

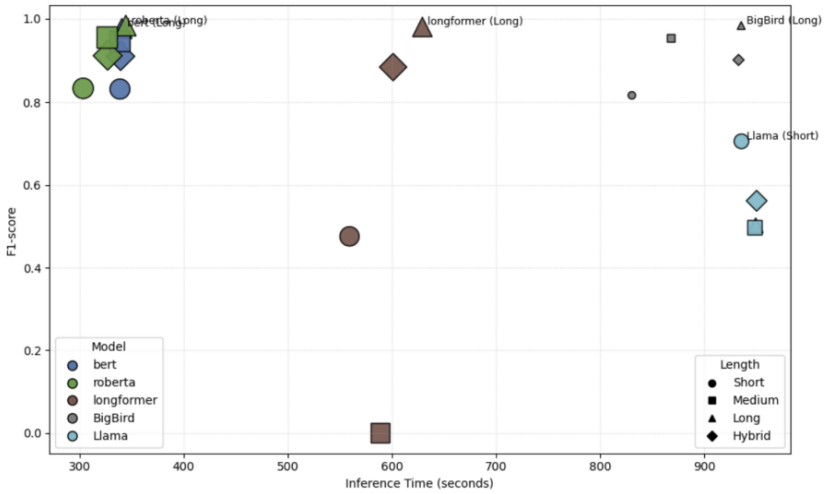


Figure 10: Accuracy–efficiency trade-off: X = inference time, Y = F1-score, bubble size = memory usage, color = model, marker = input length.

7.5 Statistical Tests

- Paired t-test and Wilcoxon signed-rank test performed between models for each length category.
- Determines if differences in accuracy/F1-score are statistically significant.

	Length Category	Model 1	Model 2	Wilcoxon p-value (F1)	Statistically Significant
0	Hybrid	bert	roberta	0.6250	False
1	Hybrid	bert	longformer	0.0625	False
2	Hybrid	bert	Llama	0.0625	False
3	Hybrid	roberta	longformer	0.1875	False
4	Hybrid	roberta	Llama	0.1250	False
5	Hybrid	longformer	Llama	0.1250	False
6	Long	bert	roberta	0.1250	False
7	Long	bert	longformer	0.8750	False
8	Long	bert	Llama	0.2500	False
9	Long	roberta	longformer	0.1250	False
10	Long	roberta	Llama	0.2500	False
11	Long	longformer	Llama	0.2500	False
12	Medium	bert	roberta	0.1250	False
13	Medium	bert	longformer	0.1250	False
14	Medium	bert	BigBird	1.0000	False
15	Medium	bert	Llama	0.1250	False
16	Medium	roberta	longformer	0.1250	False
17	Medium	roberta	BigBird	0.8750	False
18	Medium	roberta	Llama	0.1250	False
19	Medium	longformer	BigBird	0.1250	False
20	Medium	longformer	Llama	0.1250	False
21	Medium	BigBird	Llama	0.1250	False
22	Short	bert	roberta	0.6250	False
23	Short	bert	longformer	0.0625	False
24	Short	bert	BigBird	1.0000	False

	Model	Length 1	Length 2	Wilcoxon p-value (F1)	Statistically Significant
0	bert	Hybrid	Short	0.6250	False
1	bert	Long	Medium	0.8750	False
2	roberta	Hybrid	Short	0.6250	False
3	roberta	Long	Medium	0.8750	False
4	longformer	Hybrid	Short	0.0625	False
5	longformer	Long	Medium	0.1250	False
6	BigBird	Long	Short	0.8125	False
7	BigBird	Medium	Hybrid	0.1250	False
8	Llama	Hybrid	Short	0.3125	False
9	Llama	Long	Medium	1.0000	False

7.6 Visual Comparison

- Bar charts and line graphs comparing F1-score, Accuracy, and ROC-AUC across models and length categories.
- Helps in quick visual identification of trends.

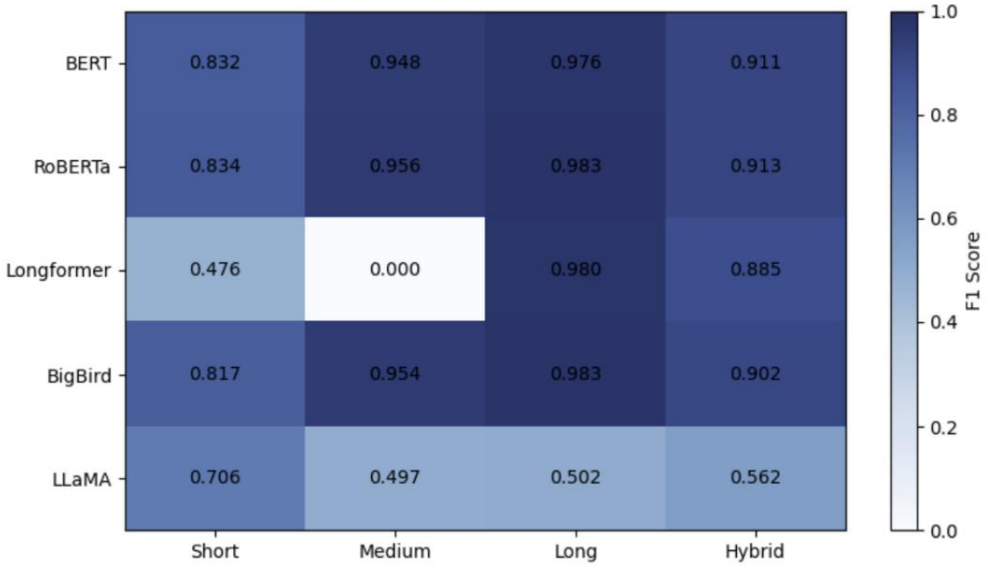


Figure 8: F1-score heatmap for all transformer models across input length categories (Short, Medium, Long, Hybrid)

8 Troubleshooting & Common Issues

This section provides solutions to common issues that might arise during the setup, execution or replication of the project.

8.1 Kaggle Environment Issues

- Problem: GPU not available or runtime error: CUDA out of memory. Solution:
  - Ensure T4 x2 GPU accelerator is selected under Kaggle Notebook Settings → Accelerator → GPU (T4 × 2).

- If memory errors persist, reduce batch size in training code.
- Restart the Kaggle notebook runtime before rerunning.
- Train the model in batches across different sessions and save the models.
- Download the models and reuse them as input during evaluation.

8.2 Dataset Loading Errors

- Problem: File not found or No such file or directory.
  Solution:
  - Ensure datasets are uploaded to the Kaggle notebook "Data" tab.
  - Verify dataset file paths in the code match the uploaded dataset location.
  - If using external datasets from Hugging Face, check internet access settings in the Kaggle environment.

8.3 Model Download Failures

- Problem: Transformer model fails to load due to connection timeout or size limit.
  Solution:
  - Enable Internet in Kaggle Notebook Settings.
  - Download model locally once and store in Kaggle Dataset for reuse.
  - Use cache_dir argument in Hugging Face from_pretrained() to specify a persistent storage location.

8.4 Library Version Conflicts

- Problem: Import errors or unexpected behavior due to library version mismatch.
  Solution:
  - Use the provided requirements.txt file to ensure consistent package versions.
  - Run *!pip install -r requirements.txt* at the start of the notebook.

8.5 Long Training Times

- Problem: Training takes longer than expected.
  Solution:
  - Reduce dataset size temporarily for quick tests.
  - Use fewer epochs for initial debugging runs.
  - Utilize mixed precision training if supported (fp16=True in Trainer).

8.6 Output Mismatch

- Problem: Results differ significantly from expected metrics in Section 7.
  Solution:
  - Verify all preprocessing steps match the configuration manual instructions.
  - Ensure random seeds are set consistently:

Check that dataset splits and input length categorization are identical to the original code.

References

Beltagy, I., Peters, M.E. and Cohan, A. (2020) *Longformer: The Long-Document Transformer*. Available at: https://arxiv.org/abs/2004.05150.

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K. (2019) *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Proceedings of NAACL-HLT 2019. Available at: https://arxiv.org/abs/1810.04805.

FakeNewsChallenge (2017) *FNC-1: Fake News Challenge Stage 1 stance detection dataset*. GitHub. Available at: https://github.com/FakeNewsChallenge/fnc-1/tree/master

ErfanMoosaviMonazzah (n.d.) *Fake News Detection in English Dataset*. Hugging Face. Available at: https://huggingface.co/datasets/ErfanMoosaviMonazzah/fake-news-detection-dataset-English.

Hugging Face (n.d.) *huggingface_hub*. Available at: https://huggingface.co/docs/huggingface_hub/index.

Hugging Face (n.d.) *Transformers: State-of-the-Art Natural Language Processing*. Available at: https://huggingface.co/transformers/.

Kaggle (n.d.) *Kaggle Notebooks*. Available at: https://www.kaggle.com/docs/notebooks.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019) *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. Available at: https://arxiv.org/abs/1907.11692.

NumPy Developers (n.d.) *NumPy*. Available at: https://numpy.org/.

NVIDIA (n.d.) *T4 Tensor Core GPU*. Available at: https://www.nvidia.com/en-us/data-center/tesla-t4/.

Pandas Development Team (n.d.) *Pandas*. Available at: https://pandas.pydata.org/.

PyTorch Developers (n.d.) *PyTorch*. Available at: https://pytorch.org/.

Scikit-learn Developers (n.d.) *scikit-learn: Machine Learning in Python*. Available at: https://scikit-learn.org/stable/.

Seaborn Developers (n.d.) *Seaborn: Statistical Data Visualization*. Available at: https://seaborn.pydata.org/.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E. and Lample, G. (2023) *LLaMA 2: Open Foundation and Fine-Tuned Chat Models*. Available at: https://arxiv.org/abs/2307.09288.

tqdm Developers (n.d.) *tqdm: A Fast, Extensible Progress Bar for Python*. Available at: https://tqdm.github.io/.

US Elections 2024 Fake News Dataset (n.d.) *US Elections Fake News Dataset*. Available at: https://data.mendeley.com/datasets/fakenewsus2024.

Wang, W.Y. (2017) *"Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection*. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 422–426. Dataset available at: https://www.cs.ucsb.edu/~william/data/liar_dataset.zip.

WELFake (n.d.) *WELFake Dataset*. Available at: https://github.com/rajkumardusad/WELFake_Dataset.

Zaheer, M., Guruganesh, G., Dubey, A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L. and Ahmed, A. (2020) *Big Bird: Transformers for Longer Sequences*. Available at: https://arxiv.org/abs/2007.14062.