The University of Manchester

FROM BERT TO FAKEBERT: A COMPARATIVE ANALYSIS OF TRANSFORMER ARCHITECTURES FOR FAKE NEWS DETECTION

AUTHOR

SALMAN ASHRAF

Student No. 10859934

SUPERVISOR

TERENCE MORLEY

Third Year Project

BSc Computer Science

Department of Computer Science

MANCHESTER, APRIL 2025

Contents

Word Count: 13967

List of Figures

List of Tables

Abbreviations and Acronyms

ANN	Artificial Neural Network
BERT	Bidirectional Encoder Representations from Transformers
CNNs	Convolutional Neural Networks
CSF3	Computational Shared Facility
CSI	Capture, Score, and Integrate
DL	Deep Learning
FC	Fully Connected
GLUE	General Language Understanding Evaluation
GRU	Gated Recurrent Unit
LLM	Large Language Model
LSTM	Long Short-Term Memory
ML	Machine Learning
MLM	Masked Language Model
NLP	Natural Language Processing
NN	Neural Network
NSP	Next Sentence Prediction
RNNs	Recurrent Neural Networks
SVM	Support Vector Machines

Notation

e^{z_i}	Exponential of the logit for class i
$\sum_{j=1}^{K} e^{z_j}$	Sums the exponentials of all logits
$O(\Theta)$	Objective Function (or Loss Function)
$\mathcal{L}$	Binary Cross-Entropy (BCE) Loss
y	True label (0 for real news, 1 for fake news)
$\hat{y}$	Predicted probability that the news is fake
$\mathbf{w}$	Model weights
$\mathbf{w}^{(t)}$	Weights at iteration t
$\mathbf{w}^{(t+1)}$	Weights at iteration $t+1$, after the update
η	Learning rate
I	Index set for the current mini-batch
$\nabla O_i(\mathbf{w}^{(t)})$	Gradient of the loss function for the i^{th} sample
$J_{\min}(\mathbf{w})$	Global minimum of the cost function
w_c	Class weight for class c
x	Input sample
$p(c \mid x)$	Predicted probability of class c given input x
$\mathbb{1}(y = c)$	Indicator function equal to 1 if $y = c$, 0 otherwise
θ_t	Model parameters at iteration t
$\hat{m}_t$	Bias-corrected estimate of the first moment of the gradients at iteration t
$\hat{v}_t$	Bias-corrected estimate of the second moment of the gradients at iteration t
ϵ	Small constant for numerical stability in AdamW update rule
λ	Weight decay coefficient in AdamW update rule

Abstract

The widespread circulation of fake news via online platforms poses a significant threat to public trust, health, and political stability. With social media, traditional media and governments accelerating the spread of misinformation, scalable and effective detection systems are urgently needed. This dissertation investigates the use of transformer-based language models for automated fake news detection, focusing on BERT Base, BERT Large, and RoBERTa Base. Each encoder is paired with one of three classifier architectures: a standard fully connected (FC) layer, a hybrid BERT+CNN model, and an enhanced FakeBERT architecture that applies parallel convolutional filters to the transformer outputs. In total, nine model configurations were developed and evaluated across 17 controlled experiments. These were tested in both in-domain and out-of-domain scenarios to assess accuracy and generalisation. My results show that RoBERTa consistently outperformed BERT-based models, and that CNN-based classifiers, especially FakeBERT, enhanced robustness against domain shifts. In my experiments, the best-performing configuration, RoBERTa combined with FakeBERT, achieved 98% in-domain accuracy and 89% out-of-domain accuracy, demonstrating strong resilience and cross-distribution performance. This study highlights the benefits of combining powerful pre-trained language encoders with CNN-based feature extractors to capture nuanced linguistic cues indicative of fake news. The comparative analysis not only confirms the effectiveness of hybrid architectures but also introduces a novel pairing (RoBERTa with FakeBERT) not previously explored in depth. These findings offer valuable guidance for building more accurate, generalisable, and trustworthy fake news detection systems.

Declaration of Authorship

I hereby confirm that this report is my own original work unless referenced clearly to the contrary, and that no portion of the work referred to in the report has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Manchester, April 2025

Salman Ashraf

Copyright

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Dr. Terence Morley , for his unwavering guidance, patience, and insight throughout the duration of this project. His consistent support during weekly meetings, constructive feedback, and ability to steer the project in the right direction at critical moments played a crucial role in shaping both the technical depth and academic quality of this dissertation.

I am particularly thankful for his accessibility and encouraging mentorship, which provided clarity whenever I encountered conceptual or practical challenges. This project would not have been possible without his ongoing encouragement and thoughtful supervision, and I am sincerely grateful for the opportunity to have worked under his guidance.

1

Introduction

1.1 Context and Motivation

The spread of fake news has emerged as a critical global challenge. As of 2025, over 5.24 billion people (about 64% of the world's population) are active on social media platforms [7] making it one of the biggest sources of misinformation alongside traditional media. These platforms enable rapid dispersion of content, often without verification. Unverified or intentionally false information can swiftly influence public opinion, with serious real-world consequences. High-profile incidents, such as misinformation during election campaigns or public health crises, highlight the societal impact of fake news [5]. Vosoughi et al. [57] found that false news stories are 70% more likely to be retweeted than true ones. During the COVID-19 pandemic, 78% of U.S. adults reported believing or being uncertain about at least one false claim related to the virus or vaccines [12]. Figure 1.1 shows widely shared fake news headlines from 2020 to 2023, highlighting the diversity and impact of such content on digital platforms.

Traditional methods (e.g. manual fact-checking or simple keyword filters) cannot keep pace with the content volume. This motivates using artificial intelligence (AI), particularly natural language processing (NLP) techniques, to detect fake news and curb its spread automatically. Recent advances in transformer-based language models like BERT have revolutionised NLP by providing deep contextual understanding. Leveraging these models for fake news detection is promising, as they can capture subtle linguistic cues and semantic context that simpler models might miss. However, evaluating how well such models generalise across different news domains remains crucial as to whether their decisions can be interpreted and trusted. This project is driven by the need for sturdy, generalisable fake news detectors and aims to compare BERT with specialised variants to assess their effectiveness. As illustrated in Figure 1.2, detection methods are generally categorised into three types: content-based, context-based, and hybrid approaches [68].

Content-based methods focus on the article's text and images, using stylistic or semantic features such as sentiment, complexity, keyword patterns, or factual inconsistency. Context-based methods rely on metadata like user interactions, source credibility, and the propagation behaviour of news on social media. Hybrid approaches combine

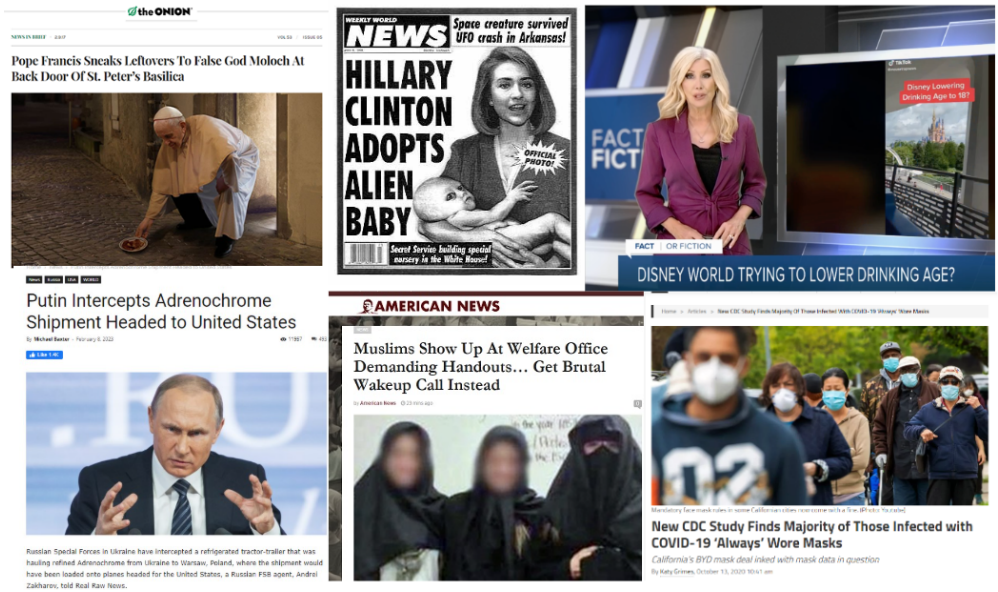


Figure 1.1: ***Widely Shared Fake News Stories from 2020–2023.*** *Examples of popular fake news stories that circulated widely on social media platforms. (Source:* `https://libguides.lib.cwu.edu/c.php?g=625394&p=4391900`*)*

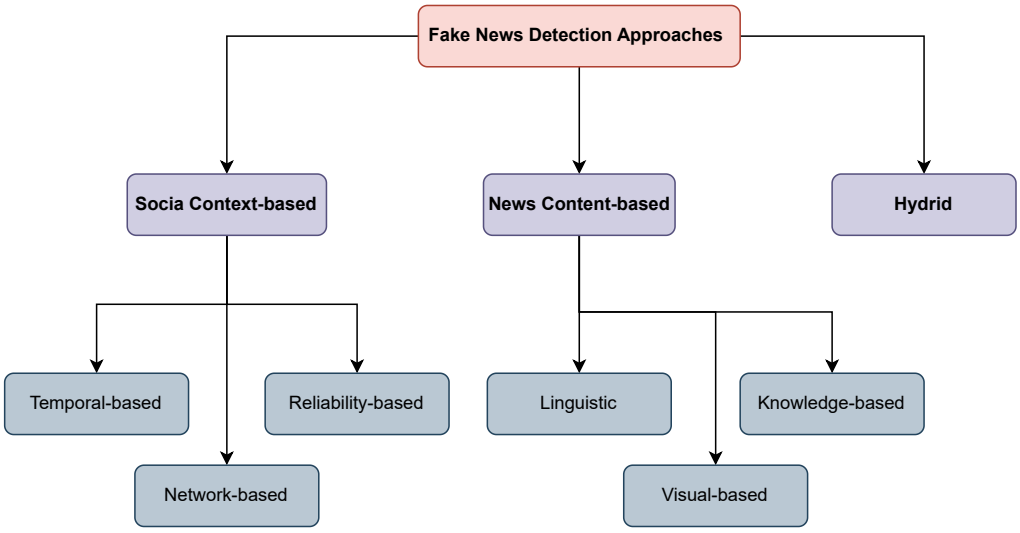


Figure 1.2: ***Taxonomy of Fake News Detection Approaches.*** *This diagram categorises detection strategies into social context-based, news content-based, and hybrid approaches, with subtypes including linguistic, visual, knowledge-based, reliability-based, and temporal methods.*

these signals to improve detection accuracy and strength [50].

Natural Language Processing (NLP) has played a central role in feature detection in recent years. Early models used handcrafted linguistic features or applied CNNs and RNNs to textual data [25]. The emergence of transformer architectures, particularly BERT (Bidirectional Encoder Representations from Transformers) [8], marked a significant advance. BERT's bidirectional self-attention mechanism allows it to model long-range dependencies in text, resulting in deeper contextual understanding.

This context motivates the exploration of advanced transformer models for fake news detection, particularly in content-based scenarios where linguistic and semantic cues play a central role.

1.2 Aims and Objectives

This dissertation is guided by the overarching aim of evaluating and enhancing transformer-based models for automated fake news detection. The project specifically focuses on the comparative analysis of transformer language models and their downstream classification architectures in both in-domain and out-of-domain settings. The objectives are as follows:

- **Evaluate Transformer Models for Fake News Classification:** Assess the performance of three pre-trained transformer-based language models, BERT Base, BERT Large, and RoBERTa Base, on the fake news detection task. Each model is fine-tuned on labelled news data to analyse how model size and pre-training strategy differences affect detection performance.
- **Compare Classification Architectures:** Implement and evaluate three classifier architectures built on top of transformer embeddings:
  1. A baseline Fully Connected (FC) model.
  2. A hybrid BERT+CNN model.
  3. A FakeBERT-style model with parallel CNN blocks inspired by Kaliyar et al. [25]

  This objective investigates whether including convolutional layers enhances local feature extraction and improves classification performance relative to the standard FC approach.
- **Assess In-Domain and Out-of-Domain Generalisation:** Evaluate each model architecture combination by training on one dataset and testing both held-out and unseen datasets. This aims to assess the generalisation capability of each configuration and determine which combinations maintain strong performance when exposed to news content from different distributions or domains.
- **Investigate Regularisation and Overfitting:** Explore the impact of regularisation strategies on model performance, such as dropout layers applied to transformer outputs and CNN filters. We aim to reduce overfitting, particularly in larger

models and evaluate the trade-off between in-domain accuracy and cross-domain generalisation.

By meeting these objectives, the project seeks to advance the development of reliable fake news detection systems and contribute insights into how transformer-based models can best be used and improved for this task.

1.3 Ethical Considerations

This project was conducted with careful attention to ethics. Data ethics was a foremost consideration: the datasets used (see Section 3.3) consist of publicly available news articles and fabricated stories previously published or shared online. No private or personal data were used, and no human participants were directly involved in generating new data. We verified that all datasets come from open sources with clear usage licenses. We systematically evaluated the project for any ethical risks using the University of Manchester's Ethics Decision Tool. Figure 1.3 shows the screenshot of the Ethics Decision Tool's outcome page for our project. The tool confirmed that no formal ethical approval was required for our study since it involves analysis of public domain content and does not collect personal data or interact with human subjects.

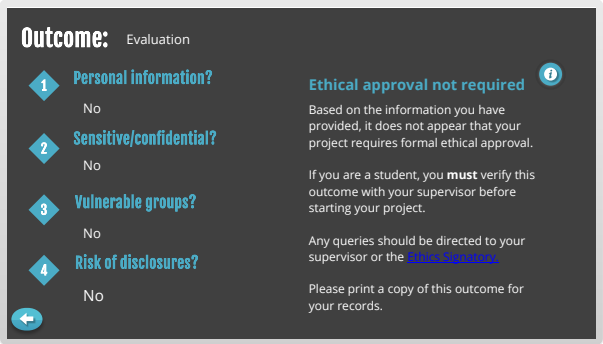


Figure 1.3: ***Ethics Decision Tool Evaluation****. The screenshot confirms that no formal ethical approval was required, as the project involves only publicly available data and does not include personal information or human subject interaction.*

1.4 Overview of Key Theories and Techniques

This project builds on several foundational Natural Language Processing (NLP) and deep learning concepts. Fake news detection is framed as a binary text classification task, where early methods used handcrafted features with traditional classifiers. Recent approaches leverage contextualised embeddings, such as those from BERT and RoBERTa, which generate dynamic word representations based on surrounding context, critical for detecting nuanced or deceptive language [31].

Transformer models like BERT use self-attention mechanisms to model long-range dependencies in text. BERT Base and Large differ in parameter size and capacity, while

RoBERTa enhances BERT's pre-training by removing the next-sentence prediction task and increasing training data and iterations. These models are fine-tuned on downstream tasks using transfer learning, allowing for high performance even with limited labelled data [8].

We also explore convolutional neural networks (CNNs) applied to BERT outputs to improve local pattern recognition. CNNs can capture short-term dependencies and phrases indicative of fake news. The FakeBERT architecture builds on this by applying parallel CNN filters of varying kernel sizes to BERT embeddings, extracting multi-scale features. While this increases model complexity, it has been shown to improve detection accuracy when paired with appropriate regularisation [25].

1.5 Project Planning and Execution

This project was conducted over the academic year following a structured and iterative workflow, as outlined in the Gantt chart in Figure 1.4. The initial phase involved a comprehensive literature review, which shaped the methodological approach and model selection. Existing influential works provided a taxonomy of fake news detection techniques. These insights positioned the project firmly within the fake news detection paradigm.

After establishing the theoretical foundation, the next phase involved data acquisition and preprocessing. Two benchmark datasets (detailed in Section 3.3) were selected to provide evaluation scenarios. Extensive preprocessing, including cleaning text and tokenisation, was performed to ensure the data was suitable for model training.

Hyperparameter tuning and experimentation were conducted in several iterations. Regularisation strategies such as weight decay and dropout addressed overfitting, particularly with BERT Large. The evaluation was split across in-domain and out-of-domain tasks to test generalisation(explained in Table 2.2.3, and misclassifications were analysed to assess limitations.

A research journal was maintained throughout the year, documenting implementation changes, troubleshooting efforts, and experimental insights. Regular supervisory meetings helped refine objectives and milestones. Report writing was phased throughout the year, with writing activity pausing during code-intensive stages and resuming after key experimental phases.

By the final phase, all model–architecture combinations had been tested, results analysed, and the dissertation drafted with insights grounded in both theoretical understanding and experimental evidence.

1.6 Report Structure

The remainder of this dissertation is organised as follows:

- **Chapter 2 – Literature Review**

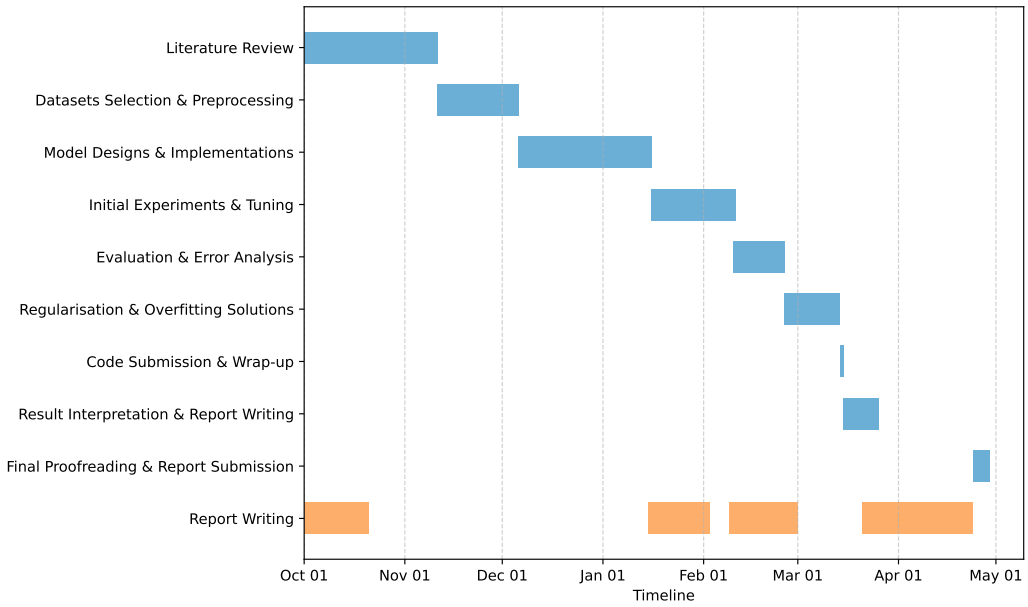


Figure 1.4: ***Chart of Project Timeline****. Report writing was interleaved across multiple phases, with breaks during code-heavy periods such as model development and experimentation.*

  - Defines fake news and related terminology.
  - Explains the fundamental tools used in model implementation.
  - Reviews prior research.
  - Highlights existing models.
  - Introduces BERT's architecture and pipeline.
  - Identifies limitations addressed by this project.
  - Highlights technologies used in the prior research.

- **Chapter 3 – Methodology**
  - Describes datasets and preprocessing steps.
  - Details model architectures: BERT/RoBERTa + FC, BERT + CNN, and FakeBERT.
  - Explains training setup, hyperparameters, and regularisation techniques.
  - Highlights issues encountered and solutions.
  - Outlines in-domain vs out-of-domain evaluation strategies.

- **Chapter 4 – Results and Evaluation**
  - Reviews hyperparameter tuning results.
  - Presents in-domain and generalisation results with performance metrics.
  - Includes tables, confusion matrices, and training behaviour analysis.
  - Compares model performance and discusses strengths/weaknesses.

- **Chapter 5 – Discussion and Conclusion**
  - Interprets results relative to objectives and the literature.
  - Discusses limitations, overfitting, and model comparisons.

- Includes personal reflection and proposals for future work.
- Summarises contributions and key findings.

This structure reflects the logical progression from background and design through to experimentation and final insights.

2

Background and Theory

2.1 Defining Fake News and its Challenges

2.1.1 Definition of Fake News

False or incorrect information has been disseminated throughout history via various means. Often, they deceive or mislead the audience for various reasons, including political gain, financial profit, or to manipulate public opinion. However, the term "fake news" only became widely used in the 21st century. Merriam-Webster defines "fake" as counterfeit and "news" as information disseminated through news media [1]. There have been attempts to characterise this term. According to [23], it is defined as misinformation that is fabricated and spread on social media to mislead the audience for political and/or financial gain.

Another example is from an associate professor at the University of North Carolina, who defines the term as "deliberately and strategically constructed lies that are presented as news articles and are intended to mislead the public" [47]. Therefore, this specifies that fake news is intentionally crafted to be misleading and manipulative. Often designed to shape public opinion by exploiting readers' emotions [62]. The historical development and propagation of fake news across different eras is summarised in Figure 2.1.

In this paper and project, the term "fake news" will denote a piece of **information intentionally or purposefully fabricated and presented as legitimate.** This definition aligns with the majority of modern experts in this field [9, 31, 54].

2.1.2 Challenges in Detecting Fake News

Detecting and filtering fake news presents several key challenges, primarily ambiguity and context dependency. Figure 2.2 shows an example of ambiguity in language, where a single phrase can be interpreted in multiple ways, potentially leading to misunderstanding or misinformation. Some detection methods focus on writing styles, as fake news is often crafted with deliberate falsifications [25]. Nonetheless, as explained in [50], fake news is designed to mislead, making content-based detection insufficient.

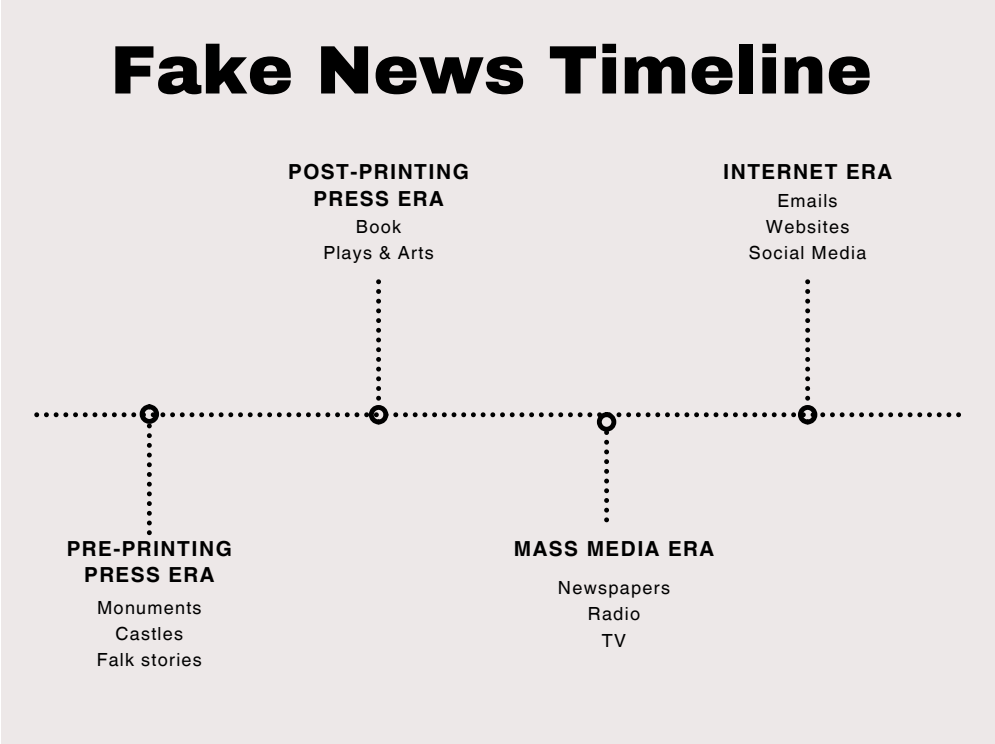


Figure 2.1: ***Fake News Timeline****. The figure illustrates the evolution of fake news dissemination across different historical eras, from the pre-printing press period to the rise of the internet and social media [45].*

This complexity makes it difficult for automated systems and human readers to assess authenticity. The increasing sophistication of misinformation also deepens the problem.

Figure 2.2: ***Example of Ambiguity in Language****. The figure illustrates how ambiguous phrasing ("Call me a cab!") can lead to multiple interpretations, demonstrating one of the challenges in both human and automated understanding of fake news. (source: https://www.thoughtco.com/syntactic-ambiguity-grammar-1692179)*

Trolls Human users who spread misinformation, manipulate feelings, and provoke strong reactions using sensational language or misleading visuals. This emotional manipulation impairs the audience's ability to critically assess content, accelerating the

spread of misinformation on social media [15, 50].

The vast scale of information transmission further complicates detection. Millions of tweets, posts, and articles are shared daily, rendering manual verification impractical and challenging for automated systems to keep pace.

Moreover, the low barrier to creating social media accounts has enabled the proliferation of social bots, algorithmically controlled accounts that automatically generate and disseminate fake news [11, 50]. In parallel, cyborg accounts, operated by humans with algorithmic assistance, blend human input and automation to amplify misinformation campaigns [15].

2.1.3 Psychological and Cognitive Biases

The spread of fake news is closely tied to psychological and cognitive biases, which make humans inherently poor at distinguishing between true and false information [3, 50].

Naive Realism Refers to the cognitive bias whereby individuals view their perspectives objectively accurate while perceiving opposing views as uninformed, irrational, or biased [61]. This bias often leads to social conflict and misunderstandings as individuals struggle to recognise the validity of differing viewpoints.

Confirmation Bias Consumers are more likely to accept information that aligns with their pre-existing beliefs [36]. As discussed in [50], confirmation bias makes fake news particularly potent, as individuals may accept demonstrably false stories that reinforce their worldview. Once such misperceptions are formed, they are difficult to correct.

Emotional Triggers As discussed in Section 2.1.2, emotional manipulation is a powerful tool in fake news transmission. Emotional content fosters attention, engagement, and the likelihood of sharing without verification [18]. Misinformation designed to provoke strong emotional reactions is more likely to spread rapidly across social networks.

2.2 Machine Learning, NLP, and Deep Learning for Detection

Fake news detection relies heavily on Natural Language Processing (NLP), Machine Learning (ML), and Deep Learning (DL) tools. These components allow systems to extract, represent, and classify textual patterns to distinguish between reliable and deceptive news content.

2.2.1 NLP Fundamentals

Tokenisation The process of segmenting text into units, such as words or subwords, enabling structured analysis by ML models. Modern NLP tokenisers can effectively handle language nuances, contractions, and punctuation [24].

Embeddings Word embeddings transform tokens into dense numerical vectors. Classic methods like Word2Vec [34] and GloVe [39] encode semantic relationships through co-occurrence patterns. Contextual models like BERT [8] generate dynamic embeddings that depend on the surrounding words, improving performance on nuanced text tasks.

Semantics Semantic modelling focuses on meaning extraction, capturing relationships like synonymy, antonyms, and entailment, crucial for detecting stylistic and rhetorical cues in deceptive language [24].

Interpretability Interpretability in NLP refers to the ability to understand and explain model predictions. Techniques such as attention visualisation and feature importance analysis help reveal which textual elements influence model outputs [31].

2.2.2 DL Fundamentals

Convolutional Neural Networks (CNNs) CNNs are deep learning models that apply filters over input sequences to detect local patterns, such as key phrases or n-grams. In text classification, CNNs apply filters over word embeddings to capture short-range patterns efficiently [27]. Figure 2.3 illustrates a typical CNN pipeline. While originally designed for vision tasks, this architecture is adapted in NLP to extract local text patterns.

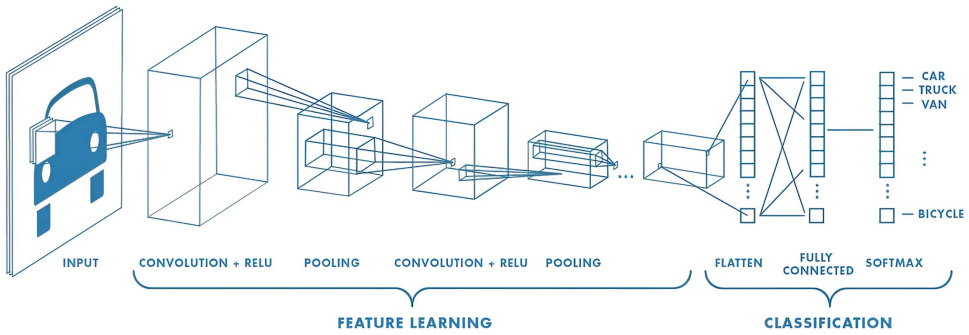


Figure 2.3: ***Convolutional Neural Network (CNN) Architecture****. The figure illustrates a typical CNN pipeline including convolution, pooling, and fully connected layers [41].*

Embedding Layers These layers map input tokens to learned vectors during training. Pre-trained embeddings (e.g., GloVe, Word2Vec) can be integrated into these layers for

more effective initialisation [39].

Fully Connected Layers Also known as dense layers, these connect every neuron in one layer to every neuron in the previous layer. They combine features extracted by convolution and pooling operations to make classification decisions [25]. As shown in Figure 2.4, fully connected layers are positioned after feature extraction stages to aggregate information and perform the final classification.

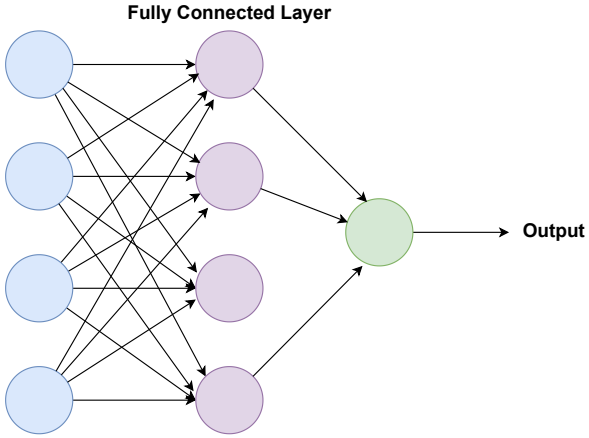


Figure 2.4: ***Fully Connected Layer****. Illustration of a fully connected neural network structure, where each neuron is connected to all neurons in the previous layer, enabling complex pattern learning and decision-making.*

Dropout Regularisation Dropout randomly turns off neurons during training to prevent over-fitting and improve generalisation. It is one of the most widely used regularisation techniques in DL [53].

Weight Decay Regularisation Weight decay penalises large model weights by adding an L_2 term to the loss function. This helps prevent overfitting by promoting simpler models [30].

Pooling Layers Pooling reduces the dimensionality of feature maps while preserving important features. Max-pooling is common in CNNs for selecting the most salient local activations [14].

Convolution Layers These apply learned filters across token sequences to detect informative patterns in the input. In 1D CNNs, filters move across the text horizontally to capture dependencies [41].

Batch Normalisation A technique to normalise layer inputs during training, stabilise the learning process and enable faster convergence [20].

ReLU The Rectified Linear Unit (ReLU) is a widely used activation function in deep learning. Mathematically defined as $\text{ReLU}(x) = \max(0, x)$ [14]. As illustrated in Figure 2.5, it outputs zero for negative inputs and returns the input itself for positive values.

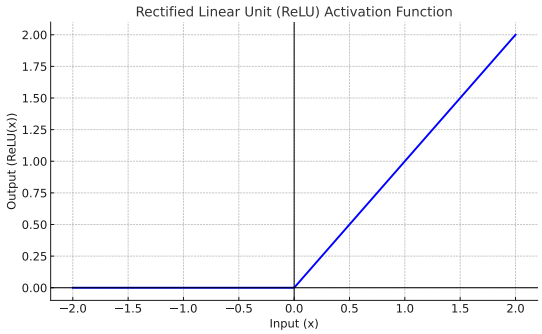


Figure 2.5: ***ReLU Activation Function****. Graph of the Rectified Linear Unit (ReLU) function, which outputs zero for negative inputs and increases linearly for positive inputs.*

2.2.3 ML Concepts

Supervised Learning (SL) SL involves learning models from labelled data to predict outcomes for unseen instances. It is the dominant paradigm in fake news detection, relying on datasets such as LIAR and FakeNewsNet [51, 60].

N-grams N-grams are consecutive n-tuples of n items (typically words or characters) from text. They are local context and sequential patterns well-suited for feature extraction in traditional machine learning models. In fake news detection, n-grams help find common phrases or linguistic patterns that can distinguish fake from real articles [24]. An example of unigrams, bigrams, and trigrams is shown in Figure 2.6.

Learning Rate The hyperparameter that determines how much the model weights are adjusted with each update. Choosing an optimal learning rate is crucial for convergence and stability [14].

Batch Size/Processing Batch size determines the number of samples processed at once during training. Smaller batches introduce noise to the gradient estimation, sometimes acting as a regulariser [26].

Softmax Function The softmax function converts raw model outputs, known as logits, into a probability distribution over output classes. It ensures that all probabilities are positive and sum to one, making it suitable for multi-class classification tasks [14]. Given a vector of logits $z = (z_1, z_2, \ldots, z_K)$, the softmax function computes the probability for each class i as:

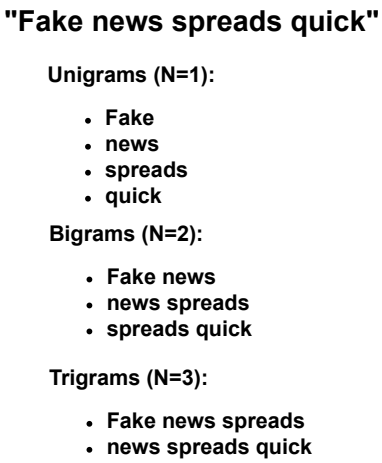


Figure 2.6: ***N-gram Example****. Illustration of unigrams, bigrams, and trigrams extracted from the sentence "Fake news spreads quickly".*

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

where e^{z_i} is the exponential of the logit for class i, and the denominator sums the exponentials of all logits. In fake news detection, softmax is typically used in the final layer to determine the likelihood that a news article belongs to a particular class.

Loss Function The loss function calculates the difference between predicted values and true labels during training. In binary classification tasks, the most commonly used function is the Binary Cross-Entropy (BCE) loss, which penalises incorrect predictions using a log-loss formulation [14]:

$$\mathcal{L} = -\big(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\big)$$

$y \in \{0, 1\}$ is the ground truth (0 for true news, 1 for fake news) and $\hat{y} \in (0, 1)$ is the predicted probability that the input is fake news. When the true label is 1, the second term drops out and the loss is $-\log(\hat{y})$, which penalises the model more if it assigns low probability to the true class. Conversely, if the true label is 0, the loss reduces to $-\log(1 - \hat{y})$.

Optimisation Algorithms In deep learning, optimisation algorithms play a crucial role in updating model parameters to minimise the loss function. Among these, *Stochastic Gradient Descent (SGD)* and *Adam* are widely adopted due to their balance of computational efficiency and convergence performance. These algorithms iteratively update the model's weights to reduce the discrepancy between predicted and actual outputs [14]. A general form of the *weight update rule* for gradient-based optimisation is

given by:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \sum_{i \in I} \nabla O_i \left(\mathbf{w}^{(t)} \right)$$

Where:

- $\mathbf{w}^{(t)}$ are the weights at iteration t,
- η is the **learning rate**,
- I is the index set for the current mini-batch,
- $\nabla O_i(\mathbf{w}^{(t)})$ denotes the gradient of the objective (loss) function for the i^{th} sample in the mini-batch.

Figure 2.7 illustrates how gradient descent guides weight updates in the direction of steepest descent.

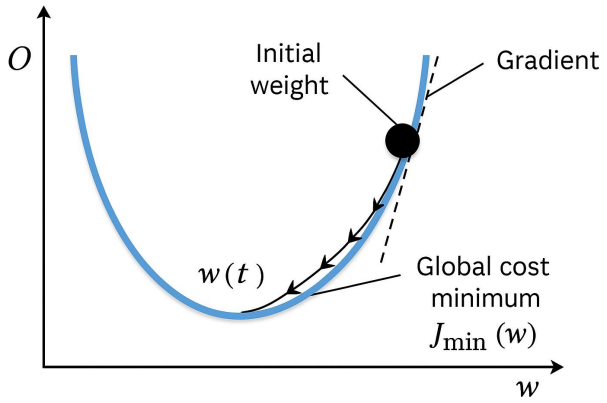


Figure 2.7: ***Gradient Descent Optimisation****. The figure depicts the weight update process during optimisation. The cost function $O(\mathbf{w})$ is plotted on the vertical axis, and the model weights $\mathbf{w}$ are on the horizontal axis. The initial weight $\mathbf{w}^{(t)}$ lies on the cost curve, and the dashed tangent line indicates the gradient at that point. Arrows show the movement in the direction of negative gradient, ultimately converging towards the global cost minimum $J_{\min}(\mathbf{w})$.*

This formulation underpins many variants of gradient-based learning algorithms.

Performance Metrics and Confusion Matrix Evaluating a classifier requires more than just accuracy, especially when dealing with imbalanced datasets [50]. A confusion matrix provides a structured breakdown of prediction outcomes by comparing predicted labels with the actual ground truth. It comprises four key components [50]:

- **True Positive (TP):** Fake news correctly classified as fake.
- **True Negative (TN):** True news correctly classified as true.
- **False Positive (FP):** True news incorrectly classified as fake.
- **False Negative (FN):** Fake news incorrectly classified as true.

These outcomes support the computation of several important metrics:

- **Accuracy:** The proportion of correct predictions over total samples. It ranges from 0 to 1, with 1 indicating perfect classification. However, accuracy alone can be misleading in imbalanced datasets.
- **Precision:** $\frac{TP}{TP+FP}$ — the fraction of predicted positive instances that are actually correct. It ranges from 0 to 1, with higher values indicating fewer false positives.
- **Recall:** $\frac{TP}{TP+FN}$ — the proportion of actual positive instances correctly identified. A high recall (close to 1) means the model misses fewer fake news articles.
- **F1-Score:** The harmonic mean of precision and recall, providing a single measure that balances both:

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

  This score is especially valuable in imbalanced settings where both false positives and false negatives carry significant consequences.
- **Macro-Averaging:** When computing precision, recall, and F1-score across multiple classes, macro-averaging treats all classes equally, calculating the mean of the metric for each class independently. This prevents majority classes from dominating the evaluation, ensuring a balanced perspective even if class distributions are unequal.
- **Confusion Matrix:** A 2×2 table that visualises TP, TN, FP, and FN, offering insight into specific types of classification errors [52]. Table 2.1 will be used to record evaluation outcomes in Chapter 4.

Table 2.1: *Labelled confusion matrix showing the relationships between actual and predicted labels in binary classification.*

True Label	Predicted: True	Predicted: Fake
True	True Positive (TP)	False Positive (FP)
Fake	False Negative (FN)	True Negative (TN)

These metrics, used throughout this report, provide a comprehensive view of model behaviour and are essential for evaluating the effectiveness of fake news detection systems.

In-domain vs Out-of-domain In-domain testing is when models are tested on samples of data abstracted from the same dataset or distribution as the training set. Out-of-domain testing, on the other hand, tests generalisability on entirely different datasets, an essential step towards determining real-world robustness since linguistic and topical forms of online disinformation change constantly [50].

Over-fitting Overfitting occurs when a model performs well on training data but fails to generalise to unseen examples. Regularisation techniques like dropout, early stopping, and data augmentation mitigate this risk [53, 67].

2.3 Datasets for Fake News Detection

Effective fake news detection requires large, diverse, and well-labelled datasets that capture both truthful and deceptive news content. An ideal dataset should include full articles with clear ground-truth labels, representing varied topics and writing styles. Balanced class distributions and minimal noise are essential for smooth model training, and additional metadata such as source or publication date can enhance analysis [51].

Table 2.2 presents an overview of several widely used datasets in fake news detection research. These datasets vary in structure, content scope, and labelling methodology.

Table 2.2: *Summary of datasets commonly used in fake news detection research.*

Dataset	Description	Reference
LIAR	Short political statements manually fact-checked by PolitiFact. Each instance is labelled across six truth levels.	[60]
FakeNewsNet	Aggregates news content and user/social context from PolitiFact and GossipCop. Includes metadata and propagation data.	[51]
BuzzFeed-Webis	Combines fact-checked BuzzFeed stories with crawled content from Webis. Focus on U.S. politics during the 2016 election.	[50]
ISOT	Full-length news articles from legitimate (e.g., Reuters) and fake sources. Used widely in binary fake news classification.	[21]
WELFake	Over 60,000 news articles collected from multiple online sources. Cleaned and balanced for deep learning training.	[19]

Two of these datasets are utilised in this study and described in detail in Section 3.3. Their structure, class distribution, and preprocessing steps are discussed as part of the experimental design. These choices reflect a balance between dataset size, topical diversity, and compatibility with deep learning architectures used in this work.

2.3.1 Examples of Input Data

Four real excerpts from these datasets are presented in Table 2.3 to better understand the nature of the input data used to train and evaluate the fake news detection models. These examples illustrate the stylistic and factual characteristics that differentiate trustworthy news from fake news.

Each example above is a real input taken from the datasets used in this study, with their assigned labels and classifications justified as follows:

- The first example (**True**) reports a real political and economic issue involving the U.S. Export-Import Bank. It references specific institutions, political figures, and

Table 2.3: *Examples of News Articles Used in Fake News Detection, with highlighted misleading content in Fake examples.*

Label	Example Text	Source
True	*"The chief executive of the U.S. Aerospace Industries Association urged President-elect Donald Trump on Tuesday to quickly nominate new board members to the U.S. Export-Import Bank. . . said billions of dollars worth of potential orders were in limbo because the agency's board lacked the necessary number of board members to back important trade deals."*	ISOT
Fake	*"WASHINGTON (Reuters) - The White House said on Tuesday that it had "serious concerns" about a bill the U.S. Senate passed that would allow survivors and relatives of those killed in the Sept. 11 attacks to sue to seek damages from the government of Saudi Arabia. "Given the concerns that we have expressed, it's difficult to imagine the president signing this legislation," White House spokesman Josh Earnest told reporters."*	WELFake
True	*"All we can say on this one is it's about time someone sued the Southern Poverty Law Center! On Tuesday, D. James Kennedy Ministries (DJKM) filed a lawsuit against the Southern Poverty Law Center (SPLC) [...] for defamation, religious discrimination, and trafficking in falsehood."*	WELFake
Fake	*"LONDON (Reuters) - The leader of Britain's main opposition Labour Party, Jeremy Corbyn, and other politicians could lose their seats if proposals to cut the cost of politics go ahead, a report suggested on Tuesday. Under former prime minister David Cameron, the governing Conservative Party pledged to reduce the number of lawmakers from 650 to 600 to make cost savings after criticism of expense claims made by those working in parliament. [...] The Daily Telegraph newspaper said if the changes had been in place before the June election, May's Conservatives would have won a majority."*	WELFake

policy concerns, maintaining a formal, factual tone characteristic of legitimate journalism.

- The second example (**Fake**), although styled like a Reuters article, has been labelled fake due to subtle inconsistencies and selective emphasis. It highlights "serious concerns" and "difficult to imagine the president signing" to exaggerate opposition to the bill, without presenting the broader legislative context, subtly misleading the reader.
- The third example (**True**) covers a verifiable lawsuit filed by D. James Kennedy Ministries against the Southern Poverty Law Center. Although minor emotional phrasing is present ("it's about time someone sued"), the article outlines concrete, verifiable legal actions, supporting its classification as true news.
- The fourth example (**Fake**) presents credible surface-level reporting but distorts content by selectively emphasising speculative outcomes, such as the claim that the Conservatives "would have won a majority", thereby overstating the political impact of boundary changes still under discussion.

Fake news examples in the datasets often exhibit subtle fabrication, selective emphasis, or emotional distortion rather than overtly absurd claims [68]. For instance, exaggerating White House concerns and speculative political consequences introduces emotional bias and misrepresentation. By contrast, true news examples, such as the Export-Import Bank report, maintain neutrality, verifiable sourcing, and a factual tone without speculative exaggeration. Recognising these linguistic and factual patterns is crucial for both human readers and machine learning models trained to detect deceptive content.

2.4 Related Work

Prior research into fake news identification can broadly be categorised into traditional machine learning (ML) approaches and deep learning (DL) approaches, as illustrated in Figure 2.8.

2.4.1 Traditional Machine Learning Models

Early fake news detection systems utilised traditional machine learning (ML) algorithms such as Support Vector Machines (SVM), Naïve Bayes, and Decision Trees [48]. These models primarily relied on hand-engineered features like word frequency, n-grams, and sentiment polarity [2]. For instance, Ramesh [32] applied ML classifiers to news headlines and achieved an accuracy of 84.50% using a Multinomial Naïve Bayes classifier on a balanced dataset. Ahmed et al. [2] reported F1-scores ranging from 63% to 86% for SVMs and Decision Trees on opinion spam datasets, indicating decent baseline performance but also showcasing variability depending on the features used.

Shu et al. [51] took a data mining perspective, emphasising the importance of metadata (e.g., user engagement and propagation) alongside content analysis.

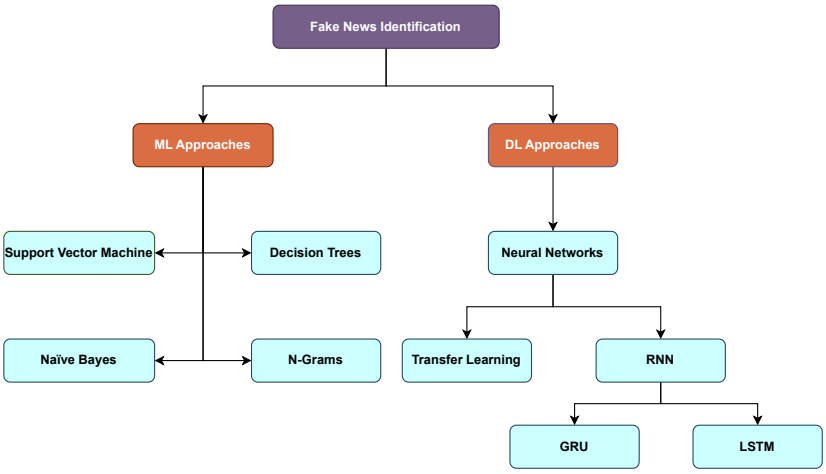


Figure 2.8: ***Overview of ML and DL approaches for fake news identification.*** *A comparison of traditional machine learning and deep learning methods used in prior studies for fake news detection [48].*

2.4.2 Deep Learning Models

Deep learning models offered a leap forward by learning hierarchical representations of text automatically since they do not require any handwritten features [25]. CNNs, initially used for image recognition, were successfully repurposed for sentence classification by Kim [27], who achieved 86.1% accuracy on the Stanford Sentiment Treebank, showing that convolutional filters could extract meaningful patterns from word embeddings.

For fake news, Ruchansky et al. [44] proposed CSI (Capture, Score, and Integrate), a hybrid model combining RNNs with social and temporal features. CSI outperformed conventional methods on two datasets, achieving an accuracy of 89.2% and 95.3%. Rashkin et al. [43] used a LSTM to classify political content, showing improved sensitivity in identifying satire and hoaxes, although they reported a drop in performance (F1 = 65%) when applied to real-world data, reflecting the challenge of domain generalisation.

Ali et al. [4], in their large-scale survey, compiled over 30 studies and concluded that deep learning models generally outperform traditional ML methods in fake news detection. However, they noted that such models typically require longer training times, high computational resources, and large labelled datasets to achieve peak performance.

2.4.3 Limitations of Prior Work

Despite advancements, deep learning models present several limitations. A major shortcoming is the difficulty in modelling long-range dependencies in text, especially relevant in news articles where critical information may be spread across multiple paragraphs. RNNs and some LSTMs can suffer from this phenomenon [64].

CNNs, while effective at detecting local patterns, are limited in their ability to model long-range dependencies [66].

Figueira et al. [11] identified poor scalability and generalisability to adversarial examples as major hurdles for both ML and early DL methods.

Hakak et al. [15] pointed out the complexity of real-world scenarios where multimodal data, including images, hyperlinks, and user comments, are integral to fake news detection. This reinforces the importance of designing tools that can deal with the diversity of data sources, types, and dynamicity. These challenges paved the way for attention-based architectures [56] as discussed in the following section.

2.4.4 Use of Transformer-Based Architecture

Transformers have revolutionised NLP by allowing models to process entire sequences in parallel and focus attention on relevant parts of the input (see Section 2.5.1). Vaswani et al. [56] introduced the Transformer, which formed the backbone for large-scale models. BERT, in particular, achieved a 4–8% improvements over LSTM and CNN baselines across multiple benchmarks (e.g., GLUE, MNLI) [8].

FakeBERT by Kaliyar et al. [25] applied BERT to fake news detection, achieving 98.90% accuracy, a massive improvement over prior ML and DL methods. This study showed that BERT not only outperforms traditional models in accuracy but also excels in recall and precision, especially for nuanced misinformation.

Szczepanski [31] took this further by integrating an explainability layer into BERT. While maintaining 98% accuracy, their model provided attention-based visualisations to justify classifications, addressing one of the biggest concerns in Transformer adoption: model transparency.

Variants such as RoBERTa [29] and DistilBERT [46] enhanced efficiency and accuracy. RoBERTa improved baseline BERT performance by 1.5–2% across various datasets, while DistilBERT reduced model size by 40% with only a 3% drop in accuracy, making it viable for edge deployment.

In general, Transformer-based models have significantly outperformed previous models in terms of accuracy (95–98%) and generalisation, while increasingly addressing concerns about interpretability and computational efficiency.

2.5 Introduction to BERT

BERT (Bidirectional Encoder Representations from Transformers), introduced by Devlin et al. [8] in 2018, set a new benchmark across a range of natural language processing (NLP) tasks. It addresses the limitations of earlier unidirectional models by capturing bidirectional context within text sequences.

Unlike traditional models that process text left-to-right or right-to-left, BERT reads in both directions simultaneously, allowing a richer understanding of context. Its development involves two stages: *pre-training* and *fine-tuning*, discussed in the following sections.

2.5.1 Transformer Architecture

BERT is built upon the Transformer architecture proposed by Vaswani et al. [56], designed to process sequential data using self-attention mechanisms instead of recurrence. This allows the model to capture **global dependencies**, making it highly effective for complex text comprehension [17].

The Transformer comprises two components: an *encoder* and a *decoder*. The encoder maps an input sequence to a continuous vector representation:

$$(x_1, x_2, \ldots, x_n) \mapsto z = (z_1, z_2, \ldots, z_n)$$

The decoder generates output sequences from these vectors:

$$z = (z_1, z_2, \ldots, z_n) \mapsto (y_1, y_2, \ldots, y_m)$$

BERT uses only the encoder, as its focus is on producing contextualised embeddings for downstream tasks [8, 17].

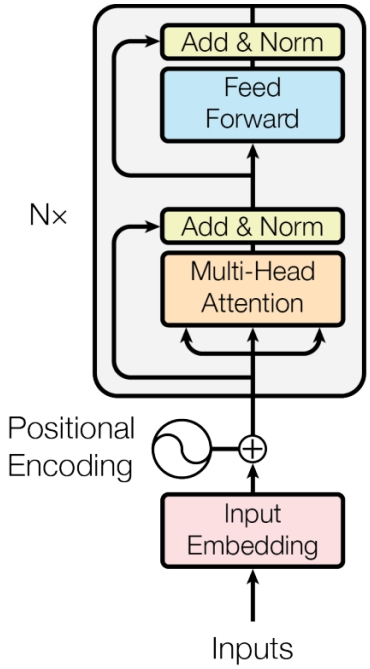


Figure 2.9: ***The Transformer Encoder****. This diagram shows the architecture of the Transformer encoder, including multi-head self-attention and feed-forward layers. These components generate contextual embeddings essential for NLP tasks such as fake news detection [56].*

As shown in Figure 2.9, each encoder layer consists of two sub-modules [17, 56]:

1. **Multi-Head Self-Attention:** Calculates attention scores for all token pairs, enabling each token to attend to relevant information across the sequence.
2. **Feed-Forward Neural Network:** A fully connected layer applied independently to each position, allowing non-linear transformations.

The following section explores the attention mechanism, the core driver of the Transformer's success.

2.5.2 Attention Mechanism

The attention mechanism enables models to dynamically focus on the most relevant parts of the input sequence. Unlike recurrent networks, it operates in parallel and considers all token relationships simultaneously [56].

Each token is projected into three vectors: query (Q), key (K), and value (V). Scaled dot-product attention is calculated as:

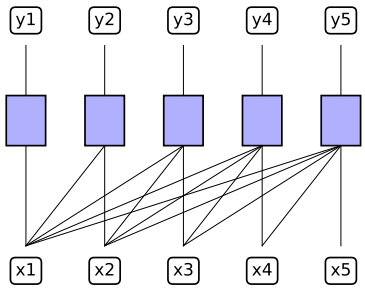
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

This generates weighted sums of the value vectors, emphasising tokens most relevant to each query. This ability to model long-range dependencies is critical for the success of BERT and related models [8].

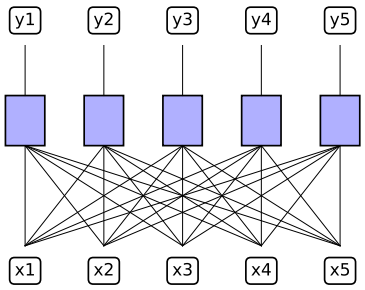
2.5.3 Bidirectional Encoding

A significant innovation of BERT is its deep bidirectional encoding, where context is drawn simultaneously from both preceding and succeeding tokens [8]. This contrasts with earlier models like ELMo [40], which process text sequentially, limiting contextual richness.

BERT jointly conditions on both left and right contexts at every layer of the Transformer encoder. As shown in Figure 2.10, unidirectional architectures (Figure 2.10a) lack access to future tokens, restricting complete contextual understanding. In contrast, BERT's bidirectional design (Figure 2.10b) enables comprehensive dependency modelling across the sequence, producing more context-aware representations critical for tasks such as fake news detection [8].



(a) *Unidirectional transformer [56].*



(b) *Bidirectional transformer (BERT) [8].*

Figure 2.10: ***Comparison of unidirectional and bidirectional transformers.*** *Figure 2.10a shows a transformer that processes input in one direction, limiting contextual depth. Figure 2.10b illustrates BERT's approach, which attends to both preceding and succeeding tokens to build richer contextual representations.*

2.5.4 Training

BERT employs Large Language Model (LLM) training strategies to achieve state-of-the-art performance across natural language understanding tasks. Its training involves two stages [8, 35]:

1. **Unsupervised pre-training** on unlabelled text.
2. **Supervised fine-tuning** on task-specific labelled datasets.

Pre-training

BERT is pre-trained on two extensive corpora to capture linguistic diversity [35]:

- **BooksCorpus:** 800 million words from over 7,000 unpublished books.
- **English Wikipedia:** 2.5 billion words of curated, factual text.

This stage enables BERT to learn general language representations by optimising the objective function $O(\Theta)$ through two tasks:

1. **Masked Language Modelling (MLM):** BERT masks 15% of tokens at random:
   - 80% are replaced with `[MASK]`,
   - 10% are swapped for a random token,
   - 10% are left unchanged.

   The model predicts the masked words using bidirectional context.
   Example: "London is the `[MASK]` of England" → "capital" [35].
2. **Next Sentence Prediction (NSP):** BERT models inter-sentence relationships by predicting whether one sentence logically follows another [8].
   Examples:
   - *Positive:* "She went to the library `[SEP]` She borrowed a book." (Yes)
   - *Negative:* "She went to the library `[SEP]` It is raining today." (No)

Fine-Tuning

After pre-training, BERT is fine-tuned on a variety of downstream tasks using labelled data. One widely used benchmark is GLUE (General Language Understanding Evaluation), which includes nine tasks designed to assess comprehension, inference, and reasoning [35, 58]. Table 2.4 summarises each task.

Table 2.5 presents BERT's performance on the GLUE benchmark, where it outperforms previous models in all tasks. These results highlight the effectiveness of its bidirectional training and large-scale pre-training strategy [55].

Table 2.4: *GLUE Benchmark Tasks with Descriptions.*

Task	Description
CoLA	Determines whether a sentence is grammatically acceptable.
SST-2	Binary classification of sentence sentiment (positive or negative).
MRPC	Classifies whether two sentences are semantically equivalent.
STS-B	Rates sentence similarity on a continuous scale.
QQP	Identifies if two questions are duplicates.
MNLI	Classifies premise-hypothesis relationships: entailment, contradiction, or neutral.
QNLI	Determines if a sentence answers a given question.
RTE	Binary entailment task based on inference from a given premise.
WNLI	Resolves pronoun coreference within a sentence.

Table 2.5: *BERT's performance on GLUE compared to prior models.*

Model	GLUE Average Accuracy (%)
BERT Large	82.1
BERT Base	79.6
OpenAI GPT	75.1
Pre-GPT SOTA	74.0

2.6 BERT Preprocessing Pipeline

2.6.1 Tokenisation and Preprocessing

Preparing raw text for BERT models requires careful tokenisation and preprocessing. BERT employs the WordPiece algorithm, which segments text into subword units, enabling it to handle rare or unseen words. The vocabulary comprises approximately 30,000 subword tokens [8, 58, 64].
Special tokens play key roles in BERT's input formatting [8]:

- **[CLS]**: Prepended to every input sequence; its output embedding is used for classification tasks.
- **[SEP]**: Separates two segments or sentences within an input.
- **[PAD]**: Used to pad sequences to a uniform length, ensuring consistent batch processing.

Preprocessing includes the following steps [8, 56]:

1. **Truncation and Padding**: Ensures all input sequences are of uniform length.
2. **Attention Masking**: Applies binary masks to distinguish between real tokens (1) and padding (0).
3. **Token IDs**: Maps each token to its corresponding integer ID in the vocabulary.

Listing 2.1 demonstrates how BERT tokenises text. IDs 101 and 102 correspond to `[CLS]` and `[SEP]`, while 0 represents `[PAD]`.

```
text = ["She went to the library.", "It is raining today."]
example = tokenizer.batch_encode_plus(
            text,
            add_special_tokens=True,
            max_length=15,
            truncation=True,
            padding='max_length',
            return_attention_mask=True,
          )
print(example["input_ids"])
[[101, 2016, 2253, 2000, 1996, 3075, 1012, 102, 0, 0, 0, 0, 0, 0, 0],
 [101, 2009, 2003, 24057, 2651, 1012, 102, 0, 0, 0, 0, 0, 0, 0, 0]]
```

Listing 2.1: ***Tokenisation in BERT.*** *Example of converting input text into numerical form using subword tokens, special tokens, padding, and attention masks.*

Final Hidden State of the [CLS] Token

As input tokens pass through BERT's Transformer layers, their representations are progressively refined. The final hidden state of the `[CLS]` token serves as a condensed summary of the entire input sequence, enabling efficient classification. However, compressing rich contextual information into a single vector can lead to the loss of fine-grained details, particularly in longer documents [31]. A more detailed discussion of this limitation is provided in Section 2.8.

2.6.2 Embedding Representation

BERT generates contextual embeddings to capture the meaning of words based on their surrounding context. As outlined in Section 2.6.1, input tokens are first mapped to IDs and padded if necessary. Each token ID is then transformed into a dense vector through the embedding layer [10].

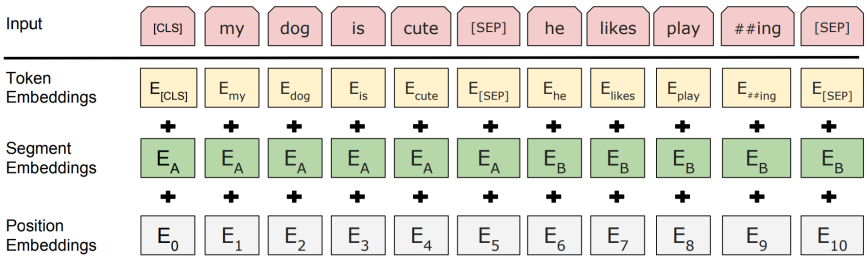


Figure 2.11: ***The input embeddings components of BERT.*** *This figure illustrates how BERT represents input text using three embedding layers: token embeddings (word representation), segment embeddings (sentence distinction), and position embeddings (word order tracking). These embeddings are summed to generate input representations that preserve both syntactic and semantic information [8].*

As shown in Figure 2.11, BERT's embedding representation comprises three distinct components [8, 10]:

- **Token Embeddings**: Represent the content of individual tokens.
- **Segment Embeddings**: Distinguish between segments or sentences (e.g., sentences A versus B).
- **Position Embeddings**: Encode the position of each token within the sequence.

This layered approach enables BERT to integrate syntactic structure and semantic meaning into its token representations, enhancing performance across downstream tasks such as fake news detection.

2.7 Transfer Learning and BERT Variants

2.7.1 Transfer Learning in BERT

Transfer learning refers to reusing knowledge gained from one task to accelerate learning in related tasks. A real-world analogy is shown in Figure 2.12, where skills such as balance and coordination developed through cycling assist in learning to ride a scooter.

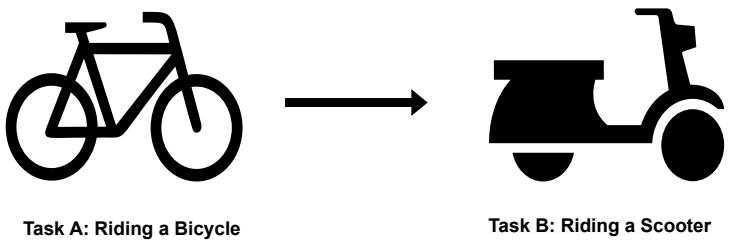


Figure 2.12: ***Transfer Learning Analogy.*** *Knowledge from riding a bicycle helps accelerate learning to ride a scooter, illustrating how transfer learning reuses foundational skills across tasks.*

Training separate models from scratch for each task demands extensive data and computational resources. Transfer learning mitigates this by pre-training a model on large corpora to acquire general linguistic knowledge, which can then be fine-tuned for specific tasks, significantly reducing data requirements and computational cost [31, 63].

Transfer learning has proven highly effective in Natural Language Processing (NLP). BERT exemplifies this strategy by first learning deep, contextual representations from massive datasets, followed by fine-tuning for tasks such as sentiment analysis, question answering, or fake news detection [8]. The rationale for selecting BERT in this project is outlined in Section 3.1.

This approach addresses three key challenges:

- **Limited Training Data:** Fine-tuning reuses broad pre-trained knowledge, reducing dependency on large labelled datasets [8, 63].
- **Training Overhead:** It significantly decreases training time and resource demands [63].
- **Generalisation:** Pre-training on diverse corpora enhances the model's ability to generalise to unseen inputs [31].

In this project, BERT is fine-tuned for the specific task of fake news detection, with the detailed rationale discussed in Section 3.1.

2.7.2 BERT Base

BERT Base is the original configuration introduced by Devlin et al. [8], balancing performance and computational efficiency. Its architecture is summarised in Table 2.6.

Table 2.6: *BERT Base Configuration*

Parameter	Value
Transformer Layers	12
Hidden Embedding Size	768
Attention Heads	12
Vocabulary Size	30,000
Total Parameters	110 million
Pre-training Objectives	MLM + NSP

For fake news detection, BERT Base provides a strong baseline and capacity to learn nuanced contextual and syntactic features. Its bidirectional attention mechanism identifies subtle linguistic patterns often present in deceptive content [4, 31].

Figure 2.13 shows examples from datasets used in this project that illustrate such patterns [19, 22]:

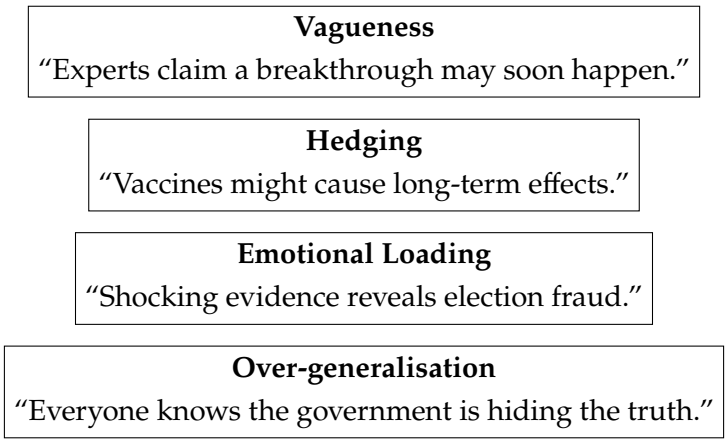


Figure 2.13: ***Subtle linguistic patterns common in fake news.*** *Vagueness, hedging, emotional loading, and over-generalisation are key stylistic markers often detected in deceptive content.*

Such patterns, often subtle and embedded within complex narratives, are effectively captured by BERT's bidirectional processing and contextual embedding mechanisms, making it highly suitable for detecting fake news across diverse sources.

2.7.3 BERT Large

BERT Large builds upon the Base model by significantly increasing representational capacity. With more layers and parameters, it better captures complex semantic

relationships and long-range dependencies. Configuration details are shown in Table 2.7.

Table 2.7: *BERT Large Configuration*

Parameter	Value
Transformer Layers	24
Hidden Embedding Size	1024
Attention Heads	16
Vocabulary Size	30,000
Total Parameters	340 million
Pre-training Objectives	MLM + NSP

Although more computationally intensive, BERT Large has demonstrated improved performance in identifying complex rhetorical strategies and stylistic markers associated with misinformation [4, 31].

Figure 2.14 shows examples from datasets that highlight such techniques [19, 22]:

Appeals to Authority
"Leading scientists confirm secret cure for cancer."

False Dilemmas
"You either support freedom or you are against us."

Conspiracy Framing
"Elite groups are controlling the world's food supply."

Loaded Language
"Corrupt officials rig the entire system."

Figure 2.14: ***Complex rhetorical strategies common in misinformation.*** *Examples include appeals to false authority, binary framing, conspiracy narratives, and emotionally loaded language.*

Such strategies require models to interpret surface-level text and detect deeper manipulative structures, a capability enhanced by BERT Large's greater depth and attention mechanisms.

2.7.4 RoBERTa

RoBERTa (Robustly Optimised BERT Pre-training Approach), developed by Liu et al. [29], retains the core architecture of BERT Base but introduces key modifications to its training procedure to enhance performance and generalisation. The model configuration is summarised in Table 2.8.

Unlike BERT, RoBERTa removes the Next Sentence Prediction (NSP) objective and applies dynamic masking during pre-training. Rather than masking tokens once statically, dynamic masking introduces random masking patterns at each epoch,

Table 2.8: *RoBERTa Base Configuration*

Parameter	Value
Transformer Layers	12
Hidden Embedding Size	768
Attention Heads	12
Vocabulary Size	50,265
Total Parameters	125 million
Pre-training Objectives	MLM (Dynamic Masking)

exposing the model to a broader range of masked examples. These adjustments result in improved performance on downstream tasks, particularly in scenarios demanding strong generalisation and nuanced contextual understanding.

2.8 Limitations of BERT and Motivation for Custom Architecture

While BERT provides a strong foundation for fake news detection, several limitations motivate architectural enhancements.

Reliance on a `[CLS]` Token BERT compresses the meaning of an entire input sequence into the final hidden state of the `[CLS]` token, as detailed in Section 2.6.1. Although effective for short texts, this design can lose critical contextual signals across longer news articles. Szczepanski et al. [31] argue that such compression restricts interpretability, making it challenging to attribute model decisions to specific words or passages. Interpretability is explained in Section 2.2.1.

Lack of Local Pattern Modelling While BERT captures global context through self-attention mechanisms, it lacks an explicit focus on local syntactic structures, such as distinctive phrases, punctuation patterns, or stylistic cues, often exploited in deceptive writing. Convolutional Neural Networks (CNNs), by contrast, are effective at detecting such localised patterns. Kaliyar et al. [25] address this gap with FakeBERT, a hybrid model combining BERT with CNN layers to enhance fine-grained feature detection.

Model Complexity BERT's scale renders it computationally intensive. In high-stakes domains such as misinformation detection, model transparency and Interpretability are crucial [31]. Furthermore, the resource demands of BERT constrain its deployment in real-time or low-resource environments.

Motivation for Hybrid Models Hybrid architectures combining BERT's embeddings with additional fully connected layers and CNN-based local feature extractors have

been proposed to address these challenges. Such models improve fine-grained feature recognition and reduce reliance solely on the [CLS] token.

These insights guided the design of the custom architectures described in Section 2.9 and later in Section 3.5.

2.9 Custom Architectures

In response to the limitations identified in BERT-based classification, three custom architectures were designed to enhance local feature extraction and improve classification performance:

- **Fully Connected (FC) Model:** A baseline model that processes the pooled output of BERT (the [CLS] token embedding) through a series of dense layers. These layers apply non-linear transformations to the aggregated representation of the news article, enabling higher-level pattern learning for fake news detection.
- **BERT+CNN Hybrid Model (BERTCNN Model):** A convolutional model that applies one-dimensional convolutional layers over BERT's sequence output (contextualised token embeddings). The convolutional filters slide across adjacent tokens, capturing local textual patterns [27]. Extracted feature maps are pooled to form a compact representation, which is then passed through dense layers for classification.
- **Multi-branch FakeBERT-inspired Model:** An advanced architecture inspired by Kaliyar et al. [25], incorporating parallel convolutional branches with filters of varying kernel sizes (e.g., 2, 3, 4, and 5 tokens). Each branch captures linguistic features at different n-gram scales, from short expressions to broader contextual patterns. The outputs from all branches are concatenated, pooled, and processed through dense layers, facilitating richer feature integration for improved classification.

All architectures are designed to reduce dependency on the [CLS] token by leveraging token-level information across the input sequence, thus enhancing model strength and interpretability.

2.10 Tools, Languages and Frameworks

A diverse set of tools, programming languages, and frameworks is employed in developing and deploying fake news detection systems. The selection of these resources is influenced by algorithmic complexity, dataset scale, and deployment requirements.

Python remains the dominant programming language in both machine learning and NLP research. Its extensive ecosystem of open-source libraries, seamless integration capabilities, and active community support make it particularly suited for building and evaluating deep learning models. Studies by Ramesh [32] and Ali et al. [4] highlight Python's pervasive role across all pipeline stages, from data preprocessing to model evaluation.

Table 2.9 summarises commonly adopted tools, languages, and frameworks, categorised by function.

Table 2.9: *Summary of common tools, frameworks, and languages used in fake news detection research.*

Purpose	Tools / Frameworks	References
ML Models	Scikit-learn, XGBoost	[4, 32]
Deep Learning	TensorFlow, PyTorch, Keras	[4, 37]
NLP Preprocessing	spaCy, NLTK, Gensim	[4, 32]
Transformers	Hugging Face Transformers, BERT, RoBERTa	[8, 25]
Evaluation Tools	Seaborn, Matplotlib, Optuna, GridSearchCV	[38]
Programming Languages	Python (dominant), R, Java	[4, 32]

3

Design and Methodology

This chapter describes the design rationale and methodological steps to implement and evaluate transformer-based models for fake news detection, as introduced in Section 2.7. The focus is applying BERT variants (BERT Base, BERT Large, and RoBERTa) across three classifier architectures: a Fully Connected (FC) head, a CNN-augmented architecture, and the multi-branch FakeBERT model.

An overview of the complete workflow, from raw text preprocessing to final classification and evaluation, is shown in Figure 3.1. It captures the entire pipeline through standard metrics, including transformer-based embedding, classification using FC, CNN, or FakeBERT architectures, loss optimisation, and performance evaluation.

3.1 BERT for Fake News Detection

As detailed in Sections 2.1.2 and 2.1.3, fake news detection presents unique challenges that traditional machine learning methods often fail to overcome. Transformer-based models such as BERT have been recognised for achieving state-of-the-art performance in text classification tasks [42], including fake news detection [25].

Sections 2.5–2.6 explained several advantages of BERT. We can summarise them as following:

- **Bidirectional Contextual Understanding**: BERT simultaneously captures context from both directions (as detailed in Section 2.5.3), enabling deeper comprehension of nuanced language [8, 25].
- **Transformer Architecture**: Its self-attention mechanism models long-range dependencies (Sections 2.5.1 and 2.5.2), aiding the detection of subtle linguistic patterns [31].
- **Pre-training and Transfer Learning**: BERT's pre-training on large corpora enables effective transfer to downstream tasks. Fine-tuning on news datasets supports accurate fake/true classification (Sections 2.5.4 and 2.7.1).
- **Variant Flexibility**: BERT Base, BERT Large, and RoBERTa, differing in architecture and training objectives, allow a comparative study of model strengths (Section 2.7).

As outlined in Section 1.2, the methodology leverages BERT's capabilities through

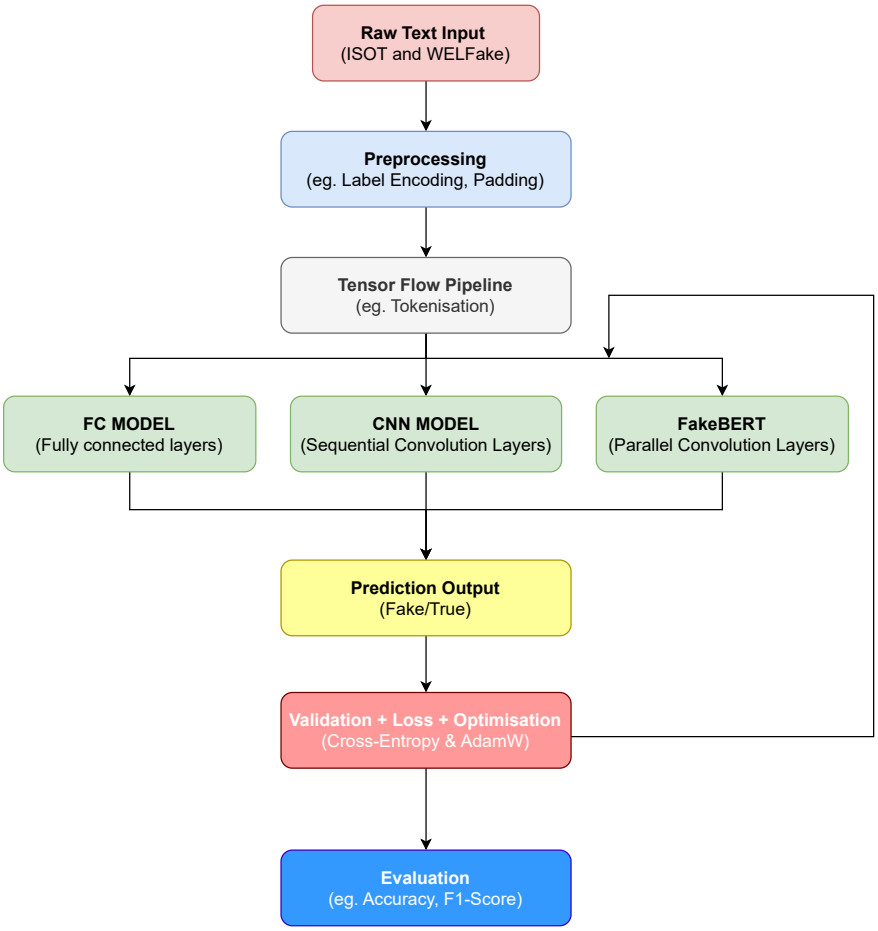


Figure 3.1: ***An overview of our fake news detection workflow.****The figure outlines the full pipeline, including preprocessing, transformer-based embedding via the Tensorflow pipeline (see Figure 3.5) followed by loss optimisation and evaluation stages.*

variant comparison and architectural experimentation to construct an effective fake news detection system.

3.2 Implementation Environment

The BERT-based fake news detection models were implemented and evaluated using a Python-based deep learning stack and a high-performance computing environment. This section outlines the software frameworks and hardware platforms utilised.

3.2.1 Programming Frameworks

Development was conducted in Python, employing several key libraries:

- **PyTorch (v2.0.1)**: The primary deep learning framework for model implementation, training, and inference, offering dynamic computation graphs and intuitive tensor operations suited for NLP tasks.

- **Hugging Face Transformers (v4.31.0)**: Used to load pre-trained BERT models, perform tokenisation, and facilitate fine-tuning, providing pre-trained weights and high-level APIs [10].
- **Scikit-learn (v1.2.2)**: Employed for evaluation metrics including accuracy, precision, recall, F1-score, and confusion matrix generation.
- **Pandas (v1.5.3)** and **NumPy (v1.23.5)**: Utilised for dataset loading, manipulation, label encoding, and statistical computations.
- **Matplotlib (v3.7.1)** and **Seaborn (v0.12.2)**: Applied to visualise dataset distributions, loss curves, and performance trends.

3.2.2 CSF3 High-Performance Computing Facility

Model training was conducted on the **Computational Shared Facility (CSF3)** at The University of Manchester, which provides access to high-performance computing resources [33]. The following configurations were used:

- **Hardware**: Training utilised **NVIDIA Tesla V100 16GB GPUs**, with access to up to two GPUs concurrently, enabling efficient experimentation with larger BERT variants.
- **Software Environment**: The environment included pre-installed deep learning frameworks supplemented by user-space installations of specific packages where necessary.
- **Job Scheduling**: Training jobs were submitted via the CSF3 batch scheduler to ensure efficient resource allocation and reproducibility.
- **Storage**: Temporary data, including model checkpoints and caches, was stored on the scratch filesystem by CSF3 policies to manage storage quotas.

Utilising CSF3 enhanced training efficiency and enabled the exploration of computationally intensive model configurations.

3.3 Data Collection and Preprocessing

This section describes the two main datasets used: the ISOT Fake News Dataset and the WELFake Dataset. The datasets' origin, size, class labels, and quality are outlined, followed by a description of the preprocessing steps applied.

3.3.1 ISOT Fake News Dataset

The ISOT Fake News Dataset, created by the Information Security and Object Technology (ISOT) Lab at the University of Victoria [21, 22], is a widely used benchmark for fake news detection.

Dataset Composition The dataset comprises two CSV files: `True.csv` and `False.csv`. After merging, the data is approximately balanced between true and fake news. Table 3.1 summarises the number of instances per class post-preprocessing. As shown in Figure 3.2, there is a slight skew towards fake news, which is acceptable for model training.

Table 3.1: *ISOT Fake News Dataset class labels after preprocessing*

Class Label	Number of Instances
True	21417
Fake	23481
Total	44898

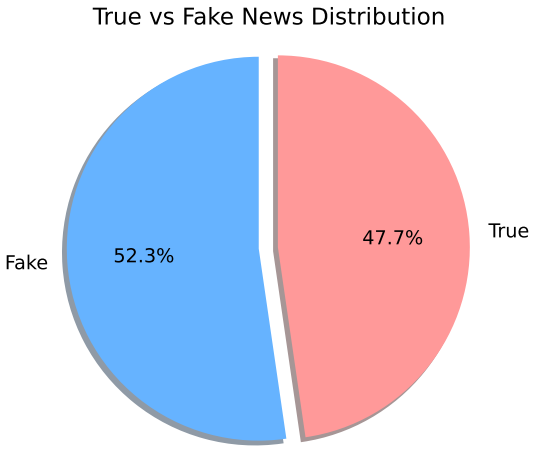


Figure 3.2: ***True vs Fake News Distribution (ISOT).*** *The dataset is roughly balanced with a slight skew toward fake news (52.3% fake, 47.7% true).*

Data Quality Minimal cleaning was required. A total of 209 duplicate articles (identical texts) were identified and removed. No missing text fields were found. After preprocessing, the dataset contained 44,898 unique news articles, providing a clean and reliable basis for model training.

3.3.2 WELFake Dataset

The WELFake dataset is an extensive collection of news articles compiled from multiple sources for fake news research [19]. It originally contained over 70,000 entries. After removing 9,415 duplicates and articles with missing text, 62,719 unique articles remained. The cleaned class distribution is shown in Table 3.2, while Figure 3.3 visualises the near balance between classes. Compared to ISOT, WELFake is larger and more diverse, covering a broader range of topics.

Usage Note The WELFake dataset was used exclusively for qualitative analysis and generalisation testing. It was not involved in training or validation, allowing models

Table 3.2: *WELFake News Dataset class distribution after preprocessing*

Class Label	Number of Instances
True	34621
Fake	28098
Total	62719

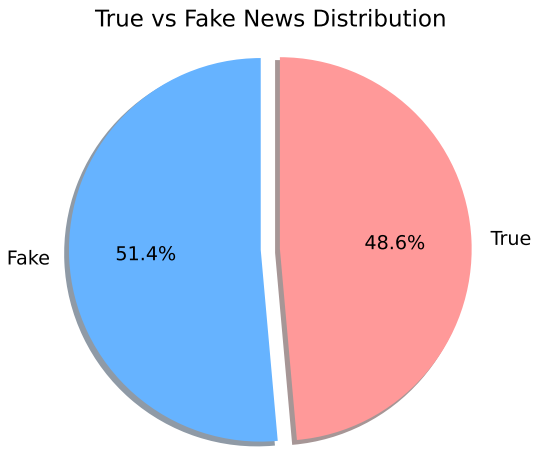


Figure 3.3: ***True vs Fake News Distribution (WELFake).*** *The dataset is nearly balanced, with slightly more fake news (approximately 51.4%) than true news (48.6%).*

trained on ISOT to be evaluated on entirely unseen data (see Section 3.9).

3.3.3 Data Preprocessing

To ensure compatibility with BERT models and maintain data integrity, both datasets underwent standard preprocessing steps:

- **Label Encoding:** Textual labels were mapped to binary values: `1` for fake news and `0` for true news. WELFake labels were inverted for consistency across datasets.
- **Duplicate Removal:** Duplicate articles were removed using `.drop_duplicates()`, eliminating 209 duplicates from ISOT and 9,415 from WELFake to prevent repeated content during training or evaluation.
- **Handling Missing Data:** ISOT contained no missing fields. WELFake entries with empty or missing text were discarded using `.dropna()`.
- **Text Cleaning:** Minimal text normalisation was applied. Original casing, punctuation, and common filler words were preserved, relying on the BERT tokeniser to handle tokenisation. Only non-content rows and excess whitespace were removed, as aggressive preprocessing (e.g., stop-word removal) is undesirable for transformer-based models.
- **Train/Validation/Test Split (ISOT):** Following cleaning, ISOT was stratified into training (70%), validation (15%), and test (15%) sets, maintaining class balance:
  - **Training:** 31,428 samples (16,436 fake, 14,992 true)

- **Validation:** 6,735 samples (3,523 fake, 3,212 true)
- **Test:** 6,735 samples (3,522 fake, 3,213 true)

These steps ensured that both datasets were clean, well-structured, and suitably prepared for fine-tuning transformer-based models, reducing potential noise and inconsistencies.

3.3.4 Sequence Length and Padding

A key preprocessing decision was setting the input sequence length to BERT's maximum of 512 tokens, justified by several considerations:

- **Coverage:** Most news articles fell below 512 tokens. Setting the maximum avoided truncating informative samples, while handling longer inputs through controlled truncation.
- **Simplicity:** Using `padding='max_length'` and `truncation=True` produced uniform input lengths, enabling efficient GPU batching and streamlined preprocessing.
- **Alternative Considerations:** Shorter limits (e.g., mean plus one standard deviation) could reduce resource usage but risk discarding important content from outlier samples. Given the distribution, full coverage was prioritised over marginal efficiency gains.
- **Implementation:** Inputs exceeding 512 tokens were truncated, while shorter inputs were padded with `[PAD]` tokens to maintain consistent input shapes.

Figure 3.4 shows the distribution of article lengths, supporting the selection of the 512-token maximum.

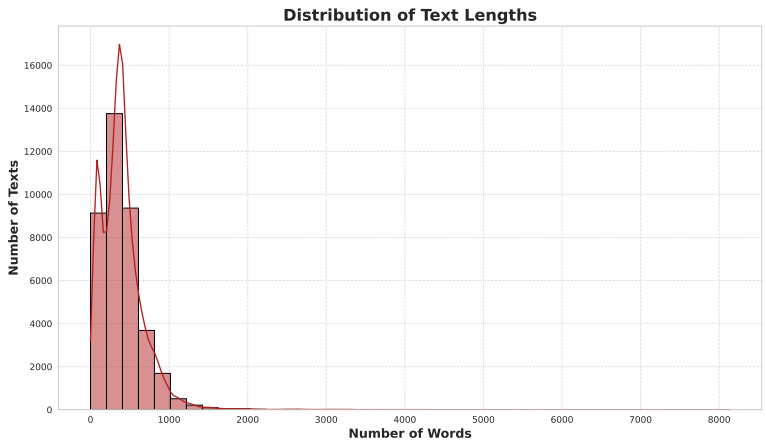


Figure 3.4: ***Distribution of Text Lengths in ISOT Dataset.*** *The histogram shows the number of news articles (Y-axis) corresponding to different word counts (X-axis). The red line represents the smoothed density curve. Most articles fall below 1000 words, with a peak between 400 and 600 words. The long tail reflects a few outliers with unusually long content.*

3.4 Input Encoding and Feature Engineering

After dataset preprocessing, raw news articles were transformed into numerical input compatible with BERT-based models. This process involved a combination of **tokenisation**, **embedding**, and **tensor conversion**, as illustrated in Figure 3.5. This section provides a task-specific summary of how these steps were applied within the implementation.

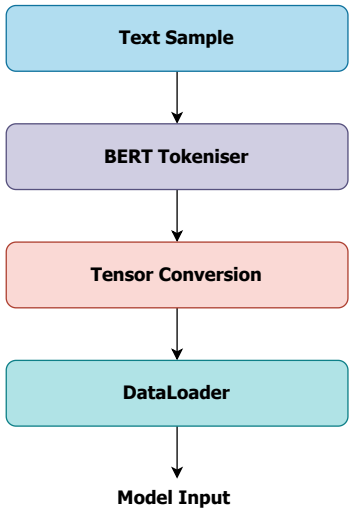


Figure 3.5: ***Text-to-Tensor Conversion Pipeline.*** *This figure shows the complete process from raw news input through tokenization, tensor creation, batching, and input into the BERT model for classification.*

3.4.1 Word Embeddings and Tokenisation

Tokenisation is the initial step that converts raw text into discrete units (tokens) using BERT's WordPiece algorithm. This subword-based approach (explained in Section 2.6.1) manages rare or unseen terms by decomposing them into familiar sub-units, enabling better generalisation.

For each article, tokenisation was performed using Hugging Face's Transformers library [10], following these steps:

- Insertion of special tokens: `[CLS]` at the beginning and `[SEP]` at the end of each sequence, with `[CLS]` serving as the primary marker for sequence-level classification.
- Splitting into WordPiece tokens and mapping each token to a unique *token ID* from BERT's vocabulary.
- Application of padding or truncation to fix sequence length at 512 tokens, as justified in Section 3.3.4.

This procedure produces a uniform input format across all samples, facilitating efficient parallel batch processing. The resulting token IDs and attention masks are inputs for the subsequent tensor conversion stage, as illustrated in Figure 3.5.

3.4.2 BERT Embeddings for Contextualised Representations

After tokenisation, each token ID is mapped into a 768-dimensional vector (for BERT-base) using a combination of three embedding types (see Figure 2.11 and Section 2.6.2):

- **Token embeddings**: capture individual word identity.
- **Positional embeddings**: encode word order.
- **Segment embeddings**: distinguish input segments (uniform in this case).

The combined embeddings are passed to BERT's multi-layer Transformer encoder Section 2.5.1, which is refined through self-attention into contextualised representations. Each token's final embedding encodes its meaning relative to the entire input sequence.

3.4.3 Tensors Conversion Pipeline

The final step involves converting preprocessed input into PyTorch-compatible tensors for efficient training:

- Token IDs, attention masks, and labels are converted into `PyTorch tensors` with shapes $(B, 512)$, where B is the batch size.
- Tensors are transferred to GPU memory using `torch.device('cuda')` to enable parallel computation.
- `TensorDataset` and `DataLoader` utilities are employed to construct **iterable, shuffled batches** of training samples, minimising CPU-GPU data transfer overhead.

This conversion finalises the pipeline from raw text to model-ready inputs, encapsulated within an efficient, GPU-accelerated data loader. As illustrated in Figure 3.5, the implementation automates the entire transformation, from raw CSV files to structured batches, using Hugging Face and PyTorch utilities.

3.5 Model Architectures

Having prepared the data and input representations, three neural network architectures were designed and implemented for fake news detection based on BERT embeddings. These architectures are introduced conceptually in Section 2.9, and this section details their implementation.

Each model employs a pre-trained Transformer encoder (BERT Base, BERT Large, or RoBERTa), differing in the additional feature extraction layers applied after BERT. The architectures are:

- **Fully Connected (FC) Model**: A baseline feed-forward classifier that uses BERT's `[CLS]` token embedding, passing it through fully connected layers for classification.
- **BERT+CNN Hybrid Model (BERTCNNModel)**: A convolution-enhanced model applying 1D CNN layers to BERT's sequence output to extract localised patterns before classification.
- **Multi-branch FakeBERT-inspired Model**: A parallel-branch architecture with multiple CNN filters of varying sizes operating concurrently to capture multi-scale textual features.

In all cases, the input text is first tokenised and encoded, as described in Section 3.4. The resulting token IDs are processed by the Transformer encoder, producing contextualised embeddings, which serve as input to the respective classifier designs described below.

3.5.1 Fully Connected (FC) Model

The Fully Connected model serves as the baseline architecture. As illustrated in Figure 3.6, the pooled output corresponding to the `[CLS]` token is extracted and passed through a stack of dense layers with ReLU activations, followed by a final output layer for binary classification.

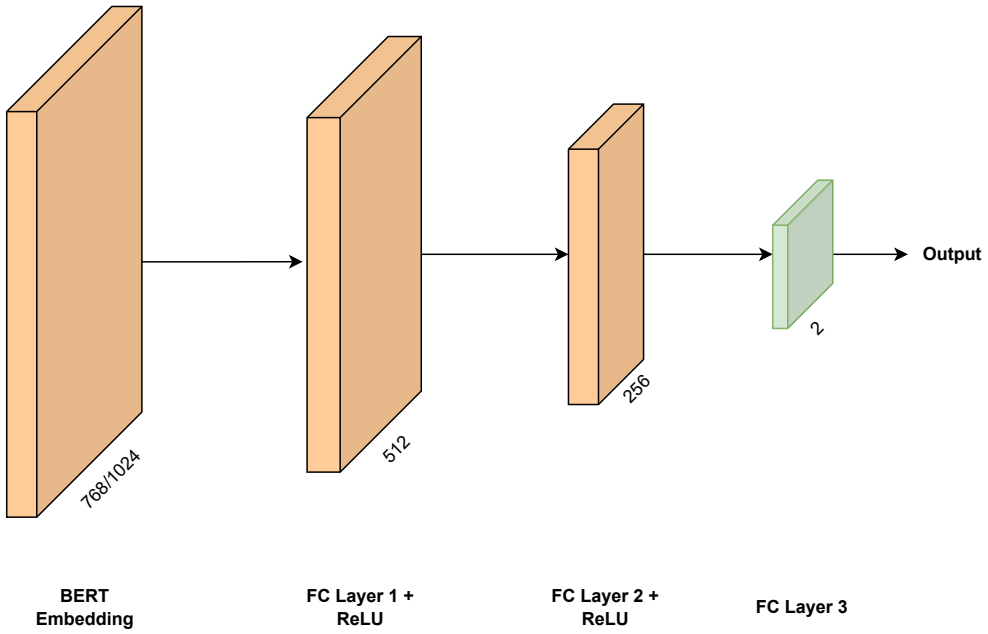


Figure 3.6: ***BERT FC model with a 3-layer fully connected classifier.*** *The `[CLS]` embedding from BERT is passed through two ReLU-activated dense layers and an output layer to predict fake vs true news.*

The layer structure for BERT Base and RoBERTa is summarised in Table 3.3, while BERT Large follows the configuration shown in Table 3.4.

Table 3.3: *Layer structure of the Fully Connected (FC) model using BERT Base and RoBERTa. The [CLS] token embedding is passed through two dense layers followed by an output layer for binary classification.*

Layer	Input Size	Output size
BERT Embedding	768	768
Fully Connected Layer 1	768	512
Fully Connected Layer 2	512	256
Output Layer	256	2

Table 3.4: *Layer structure of the Fully Connected (FC) model using BERT Large. Increased embedding size (1024) is accounted for in the first dense layer.*

Layer	Input Size	Output size
BERT Embedding	1024	1024
Fully Connected Layer 1	1024	512
Fully Connected Layer 2	512	256
Output Layer	256	2

3.5.2 BERTCNN Model

The BERT+CNN hybrid model enhances feature extraction by applying 1D convolutional layers over BERT's token sequence output. The whole sequence of token embeddings is processed to capture localised patterns indicative of deceptive writing.

The embeddings are passed through a series of convolutional and max-pooling layers, acting as local feature detectors that identify short phrases and stylistic patterns. The resulting feature maps are flattened and fed into fully connected layers for classification, as illustrated in Figure 3.7.

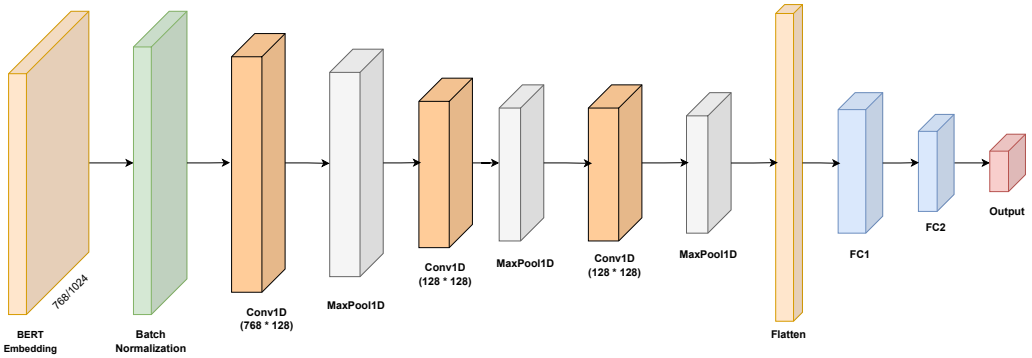


Figure 3.7: ***BERT + CNN model architecture.*** *The sequence of embeddings from BERT is processed by several 1D convolution and max-pooling layers, then flattened and fed into fully connected layers. This allows local textual patterns to be captured by the CNN before classification.*

The layer dimensions for the Base and RoBERTa models are summarised in Table 3.5, while those for BERT Large appear in Table 3.6.

Table 3.5: *The layer structure of the BERT+CNN model using BERT Base and RoBERTa. The input comprises token-level embeddings of size 768 produced by the Transformer encoder. Batch normalisation is applied before passing the embeddings through three sequential Conv1D layers (kernel size $k = 3$), each followed by max pooling to reduce the sequence length progressively. After three convolution-pooling stages, the resulting feature maps are flattened and processed through two fully connected layers before final binary classification. Here, M_i denotes the sequence length after the i-th pooling operation and N represents the total number of features after flattening.*

Layer	Input Size	Output Size
BERT Embedding (per token)	768	768
Batch Normalisation	768	768
Conv1D Layer 1 ($k = 3$)	768	128
Max Pooling Layer 1	$128 \times M_0$	$128 \times M_1$
Conv1D Layer 2 ($k = 3$)	128	128
Max Pooling Layer 2	$128 \times M_1$	$128 \times M_2$
Conv1D Layer 3 ($k = 3$)	128	128
Max Pooling Layer 3	$128 \times M_2$	$128 \times M_3$
Flatten	$128 \times M_3$	N
Fully Connected Layer 1	N	512
Fully Connected Layer 2	512	256
Output Layer	256	2

Table 3.6: *Layer structure of the BERT+CNN model using BERT Large. The input comprises token-level embeddings of size 1024 produced by the Transformer encoder. Batch normalisation is applied before passing the embeddings through three sequential Conv1D layers (kernel size $k = 3$), each followed by max pooling to reduce the sequence length progressively. After three convolution-pooling stages, the resulting feature maps are flattened and processed through two fully connected layers before final binary classification. Here, M_i denotes the sequence length after the i-th pooling operation and N represents the total number of features after flattening.*

Layer	Input Size	Output Size
BERT Embedding (per token)	1024	1024
Batch Normalisation	1024	1024
Conv1D Layer 1 ($k = 3$)	1024	128
Max Pooling Layer 1	$128 \times M_0$	$128 \times M_1$
Conv1D Layer 2 ($k = 3$)	128	128
Max Pooling Layer 2	$128 \times M_1$	$128 \times M_2$
Conv1D Layer 3 ($k = 3$)	128	128
Max Pooling Layer 3	$128 \times M_2$	$128 \times M_3$
Flatten	$128 \times M_3$	N
Fully Connected Layer 1	N	512
Fully Connected Layer 2	512	256
Output Layer	256	2

3.5.3 Multi-branch FakeBERT-inspired Model (FakeBERT Model)

The FakeBERT model extends the BERT+CNN design by incorporating multiple convolutional branches operating in parallel. As illustrated in Figure 3.8, three parallel Conv1D layers with differing kernel sizes are used to detect patterns at various scales. The outputs from these branches are concatenated and passed through additional convolutional and fully connected layers for final classification.

This architecture, inspired by Kaliyar et al. [25], enables the model to capture a wider range of local features while still leveraging BERT's contextual embeddings for global understanding.

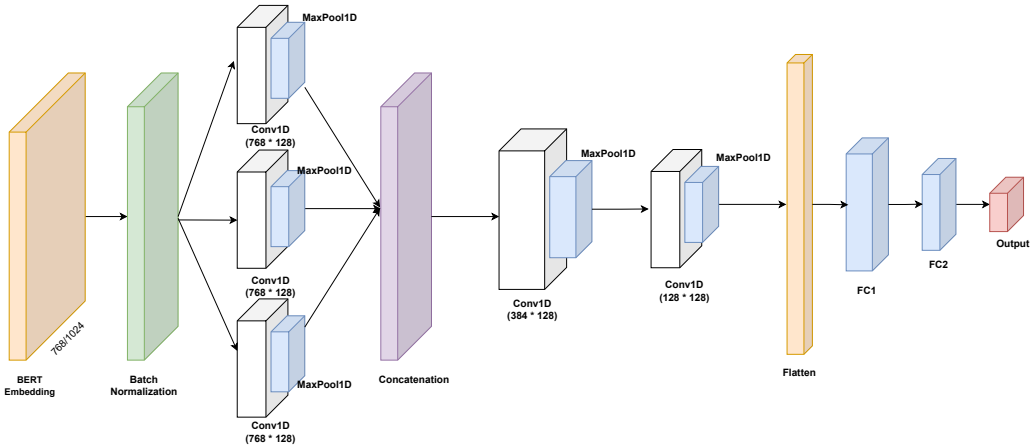


Figure 3.8: ***FakeBERT Architecture.*** *The model integrates BERT Base embeddings with parallel convolutional branches of different kernel sizes, followed by further convolutional and fully connected layers for fake news classification.*

The detailed layer structures are summarised for BERT Base and RoBERTa in Table 3.7, and for BERT Large in Table 3.8.

3.6 Training, Optimisation and Hyperparameter Tuning

With the data prepared and model architectures defined, the next stage involved establishing the training procedure and the strategies employed to optimise model performance. Fine-tuning large models such as BERT for a specific downstream task requires careful consideration of several factors, including which parts of the model are updated during training, the choice of loss function and optimiser, methods for preventing overfitting, and approaches to hyperparameter selection. Each of these aspects is detailed in the following subsections.

3.6.1 Fine-tuning Strategy

All models were trained end-to-end for fake news classification, with consideration given to both the pre-trained BERT weights and the newly added classifier layers. As discussed in Section 2.5.4, fine-tuning enables the adaptation of BERT's generic

Table 3.7: *Layer structure of the FakeBERT model using BERT Base and RoBERTa. Token embeddings of size 768 are processed through three parallel Conv1D branches with kernel sizes $k = 3$, $k = 5$, and $k = 7$. Each branch undergoes separate max pooling before concatenation. The concatenated features are further processed through sequential convolutional and pooling layers, followed by two fully connected layers for final binary classification. Here, M_i denotes the sequence length after the i-th pooling operation.*

Layer	Input Size	Output Size
BERT Embedding (per token)	768	768
Batch Normalisation	768	768
Conv1D Branches ($k = 3, 5, 7$)	768	128×3
Max Pooling (per branch)	$128 \times M_0$	$128 \times M_1$
Concatenation	-	384
Conv1D Layer 4 ($k = 3$)	384	128
Max Pooling Layer 4	$128 \times M_1$	$128 \times M_2$
Conv1D Layer 5 ($k = 3$)	128	128
Global Max Pooling	$128 \times M_2$	128
Fully Connected Layer 1	128	512
Fully Connected Layer 2	512	256
Output Layer	256	2

Table 3.8: *Layer structure of the FakeBERT model using BERT Large. Token embeddings of size 1024 are processed through parallel convolutional branches and sequential CNN layers, identical in structure to the Base variant, but scaled for the higher input dimensionality.*

Layer	Input Size	Output Size
BERT Embedding (per token)	1024	1024
Batch Normalisation	1024	1024
Conv1D Branches ($k = 3, 5, 7$)	1024	128×3
Max Pooling (per branch)	$128 \times M_0$	$128 \times M_1$
Concatenation	-	384
Conv1D Layer 4 ($k = 3$)	384	128
Max Pooling Layer 4	$128 \times M_1$	$128 \times M_2$
Conv1D Layer 5 ($k = 3$)	128	128
Global Max Pooling	$128 \times M_2$	128
Fully Connected Layer 1	128	512
Fully Connected Layer 2	512	256
Output Layer	256	2

language representations to the news domain. However, updating all parameters (see Tables 2.6, 2.7, and 2.8) is computationally expensive and may lead to overfitting, particularly with limited training data.

A frozen BERT encoder strategy was adopted to address this: the Transformer weights were kept fixed, and only the classifier layers (fully connected or convolutional) were trained. This approach reduced computational costs and memory usage while mitigating overfitting. Although fine-tuning the entire model can yield additional gains, it demands greater resources and careful regularisation. Training only the classifier layers over frozen BERT embeddings produced competitive results, consistent with previous findings regarding the strength of pre-trained representations.

3.6.2 Loss Function and Class Weighting

The models were optimised using the standard `cross-entropy loss` for binary classification, discussed theoretically in Section 2.2.3. In practice, `nn.CrossEntropyLoss` from PyTorch [37] was employed, comparing the model's output logits for the "Fake" and "True" classes against the ground-truth labels.

Given the slight class imbalance in the ISOT dataset, a weighting scheme was applied to emphasise the minority class. Class weights were calculated using `compute_class_weight` from `scikit-learn` [38], inversely proportional to class frequencies in the training set. This adjustment penalised misclassification of the rarer "True" articles more heavily than "Fake" articles, discouraging the model from biasing toward the majority class.

Formally, if w_c is the weight for class c, and (x, y) represents an input sample with true label $y \in \{0, 1\}$ and predicted probability $p(c \mid x)$, the weighted cross-entropy loss is given by:

$$\mathcal{L}(x, y) = -\sum_{c \in \{0,1\}} w_c \cdot \mathbb{1}(y = c) \cdot \log p(c \mid x)$$

where $\mathbb{1}(y = c)$ is the indicator function equal to 1 when the true label is c, and 0 otherwise. This weighted formulation was consistently applied across all models during training to mitigate the effects of class imbalance.

3.6.3 Optimiser

All models were trained using the **AdamW optimiser**, a variant of Adam that decouples weight decay from gradient-based updates, thereby improving regularisation and generalisation performance [30]. AdamW is widely regarded as the default choice for fine-tuning Transformer-based models, as recommended in the original BERT [8] and RoBERTa [29] papers. Iterative optimisation principles are discussed theoretically in Section 2.2.3.

The AdamW update rule is defined as:

$$\theta_{t+1} = \theta_t - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} - \eta \cdot \lambda \cdot \theta_t$$

where θ_t denotes the model parameters at iteration t, η is the learning rate, $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected estimates of the first and second moments of the gradients, and λ is the weight decay coefficient. In this implementation, λ was set to 0.01, consistent with established best practices.

3.6.4 Regularisation

To mitigate overfitting, particularly given the capacity of BERT-based models and the added CNN layers in hybrid architectures, the following regularisation strategies were employed:

- **Dropout**: A dropout rate of 0.3 was initially used across all models, later increased to 0.5 for the FakeBERT model following early signs of overfitting during validation (see Section 3.6.5).
- **Weight Decay**: Incorporated within AdamW, serving as an L_2 regularisation term to penalise large weights and encourage simpler model representations [30].
- **Batch Normalisation**: Applied within the convolutional layers of CNN-based models to stabilise learning and act as a mild regulariser.
- **Early Stopping**: Training was terminated if validation loss did not improve over three consecutive epochs, thereby reducing overfitting and computational cost.

These strategies are further explained in Section 2.2.2.

3.6.5 Overfitting Reduction Techniques in FakeBERT

To enhance the generalisation of FakeBERT and mitigate overfitting, as observed in Section 4.2.3, two complementary approaches were implemented. The first focused on architectural modifications, while the second introduced additional powering techniques.

First Approach: Structural Modifications Three architectural adjustments were incorporated:

- **Increased Dropout Rate**: The dropout rate was raised from 0.3 to 0.5 to reduce neuron co-adaptation and enhance generalisation.
- **Reduced Model Complexity**: The size of the fully connected layers was decreased to limit over-parametrisation.
- **Increased Weight Decay**: The weight decay coefficient was increased to strengthen L_2 regularisation and suppress overfitting.

Second Approach: Input Noise Injection To further overcome overfitting, Gaussian noise with a standard deviation of 0.1 was added to the BERT input embeddings during training. The noise layer was implemented as `self.noise_layer = GaussianNoise(stddev=0.1)`, applying random perturbations via `torch.randn_like(x) * stddev` during the forward pass. By simulating minor input variability, this technique encouraged the model to learn more generalisable features rather than memorising training examples. The noise layer was automatically disabled during evaluation to ensure stable inference. The effects of both strategies are evaluated in Section 4.2.3.

3.6.6 Grid Search for Hyperparameter Selection

A grid search was conducted to determine optimal values for two key hyperparameters: the learning rate and batch size. This focus was motivated by their sensitivity in Transformer fine-tuning, as reported in [8]. The grid search was performed using the simpler Fully Connected (FC) model architecture with BERT Base as the encoder, while all other model components were kept fixed, as summarised in Table 3.9.

The search space was defined as:

- Learning Rates: {1e-6, 5e-6, 1e-5, 5e-5, 1e-4}
- Batch Sizes: {8, 16, 32, 64}

All combinations of learning rate and batch size were evaluated, and the optimal configuration was selected based on validation loss and classification metrics. The detailed results of the grid search are presented in Section 4.1.2.

Table 3.9: *Fixed hyperparameters during grid search. Only learning rate, batch size, and number of epochs were varied.*

Hyperparameter	Fixed During Search
Number of layers	✓
Activation function	✓
Optimiser	✓
Loss function	✓
Learning rate	-
Epochs	✓
Batch size	-
Dropout rate	✓

3.7 Implementation Issues

3.7.1 Mismatch in Fully Connected Layer Dimensions

A matrix dimension mismatch was encountered between the output of the convolutional layers and the input expected by the fully connected (FC) layer. The error, shown in Listing 3.1, occurred during matrix multiplication within the FC layer:

```
RuntimeError: mat1 and mat2 shapes cannot be multiplied (16x7680 and 256x128)
```

Listing 3.1: *Matrix dimension mismatch error encountered in the fully connected layer.*

The input tensor (`mat1`) possessed a shape of $(16, 7680)$, whereas the FC layer (`mat2`) was configured to expect an input of size (batch_size, 256). Since $7680 \neq 256$, the matrix multiplication operation failed, resulting in a runtime error.

Cause: Incorrect Computation of Flattened Feature Size The error originated from an incorrect calculation of the flattened feature size following convolution and pooling operations. As the sequence length changes at each pooling stage, an inaccurate assumption regarding the final shape led to the mismatch.

Solution: Dynamic Computation of Feature Size To address this, the flattened feature size was computed dynamically based on the actual transformations performed by the convolutional and pooling layers. This dynamic calculation ensured that the FC layer was correctly initialised, as illustrated in Listing 3.2.

```
self.fc1 = nn.Linear(self.flatten_size, 128)  # Dynamically determined size
```

Listing 3.2: *Correct initialisation of the fully connected layer using dynamically determined flattened size.*

3.7.2 Mismatch in CNN Output Sizes During Concatenation

A tensor size mismatch was encountered when attempting to concatenate the outputs of multiple convolutional layers with different kernel sizes. The error, shown in Listing 3.3, arose during the `torch.cat()` operation:

```
RuntimeError: Sizes of tensors must match except in dimension 1.
Expected size 102 but got size 101 for tensor number 1 in the list.
```

Listing 3.3: *Tensor size mismatch error in FakeBERT during concatenation.*

This error occurred because `torch.cat()` requires tensors to have identical sizes along all dimensions except the concatenation axis. In the FakeBERT model, parallel `Conv1D` layers with different kernel sizes produced outputs of varying sequence lengths, violating this requirement.

Cause: Varying Kernel Sizes in Convolutional Layers FakeBERT utilises multiple convolutional branches with kernel sizes 3, 4, and 5. Given an input sequence of length $L = 512$, the output length after convolution is computed as:

$$\text{Output Length} = (L - K) + 1$$

where K denotes the kernel size. Consequently:

- $K = 3 \Rightarrow$ Output Length: 510
- $K = 4 \Rightarrow$ Output Length: 509
- $K = 5 \Rightarrow$ Output Length: 508

Subsequent application of `MaxPool1D` further reduced sequence lengths inconsistently, resulting in the observed mismatch.

Resolution: Applying Padding to Convolutions Padding was applied to each convolutional layer to maintain consistent output lengths, as shown in Listing 3.4.

```
self.convs = nn.ModuleList([
    nn.Conv1d(in_channels=embed_dim, out_channels=num_filters, kernel_size=k,
    ↪  padding=k//2)
    for k in kernel_sizes
])
```

Listing 3.4: *Applying symmetric padding to Conv1D layers to maintain consistent output sizes.*

This adjustment ensured that all convolutional outputs had identical sequence lengths, enabling successful concatenation.

3.8 Evaluation Metrics

Model performance was assessed using standard classification metrics computed via Scikit-learn: accuracy, precision, recall, F1-score, and the confusion matrix. Definitions and theoretical background for these metrics are provided in Section 2.2.3.

For this binary classification task, precision and recall offered insights into the model's ability to identify fake versus real news correctly. At the same time, the F1-score provided a balanced measure incorporating both false positives and false negatives. The confusion matrix further supported performance analysis by illustrating the distribution of correct and incorrect predictions across both classes.

3.9 Generalisation Testing

To evaluate model strength, generalisation testing was conducted by applying the final models to the out-of-domain WELFake dataset. The concept of domain testing is discussed in Section 2.2.3. No fine-tuning or retraining was performed on WELFake.

The same evaluation metrics described above were computed, and qualitative analysis of misclassified articles was undertaken to investigate error patterns and signs of overfitting. The detailed results are presented in Chapter 4 and discussed further in Chapter 5.

4

Results and Evaluation

This chapter presents the evaluation of the model configurations, combining three classifier architectures with three Transformer backbones. All models were trained on the ISOT dataset and evaluated on its test split to assess in-domain performance. Generalisation capability was tested using the out-of-domain WELFake dataset. Consistent training hyperparameters were applied across all experiments, as specified in Chapter 3. Model performance was assessed using accuracy, precision, recall, and F1-score, alongside confusion matrices to illustrate error distributions, following the definitions provided in Section 2.2.3.

All experiments were conducted using PyTorch implementations of pre-trained encoders and executed on two GPUs within the CSF3 high-performance computing environment.

4.1 Hyperparameter Tuning

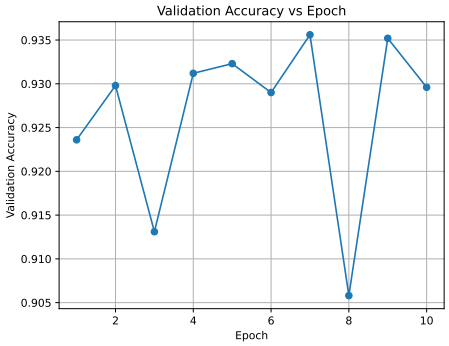
Effective hyperparameter selection was crucial to ensure that each model was appropriately trained on ISOT, providing a fair basis for comparison. As detailed below, a combination of manual epoch tuning and an extensive grid search for learning rate and batch size was performed.

4.1.1 Epoch Selection

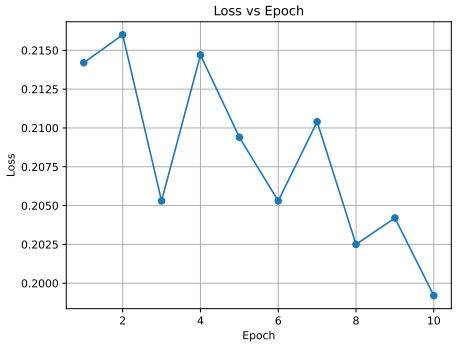
Models were trained for up to 10 epochs, with training loss and validation accuracy monitored at each step. As shown in Figure 4.1 for the BERT Base + FC model as an example, convergence typically occurred early, with limited gains beyond epoch 5. Specifically, the BERT Base + FC model peaked at 93.56% validation accuracy by epoch 7, although minor fluctuations suggested the onset of overfitting. Consequently, the number of training epochs was capped at 5 for all models.

4.1.2 Grid Search for Learning Rate and Batch Size

A grid search was conducted using the BERT Base + FC model to evaluate combinations of learning rate and batch size, as part of hyperparameter tuning on the ISOT dataset.



(a) *Validation accuracy for BERT Base + FC over epochs.*



(b) *Training loss for BERT Base + FC over epochs.*

Figure 4.1: ***Training performance of BERT Base + FC on the ISOT dataset over 10 epochs.*** *Figure 4.1a shows the progression of validation accuracy, while Figure 4.1b illustrates the corresponding reduction in training loss.*

Search Space The following search space was explored:

- **Learning Rates**: 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}
- **Batch Sizes**: 16, 32, 64, 128
- **Fixed Parameters**: AdamW optimiser, 5 training epochs, 0.3 dropout

The validation accuracies obtained are summarised in Table 4.1.

Table 4.1: *Grid search validation accuracies for different learning rates and batch sizes on the ISOT dataset.*

Learning Rate	Batch 16	Batch 32	Batch 64	Batch 128
10^{-6}	86.13%	92.65%	92.54%	94.48%
10^{-5}	93.63%	93.30%	93.19%	94.73%
10^{-4}	**95.26%**	94.04%	93.99%	94.90%
10^{-3}	94.82%	**94.97%**	**94.90%**	**95.02%**
10^{-2}	92.14%	93.08%	94.41%	94.55%

Overall, the best configuration was achieved with a learning rate of 10^{-4} and a batch size of 16, resulting in a validation accuracy of 95.26%. It was observed that excessively small learning rates (10^{-6}) led to underfitting, whereas higher learning rates (10^{-2}) introduced instability. In addition, smaller batch sizes tended to yield marginally better generalisation performance, suggesting that the optimiser benefited from noisier gradient estimates.

Visualisation The relationship between learning rate and validation accuracy for different batch sizes is illustrated in Figure 4.2, where a logarithmic scale is applied to the x-axis.

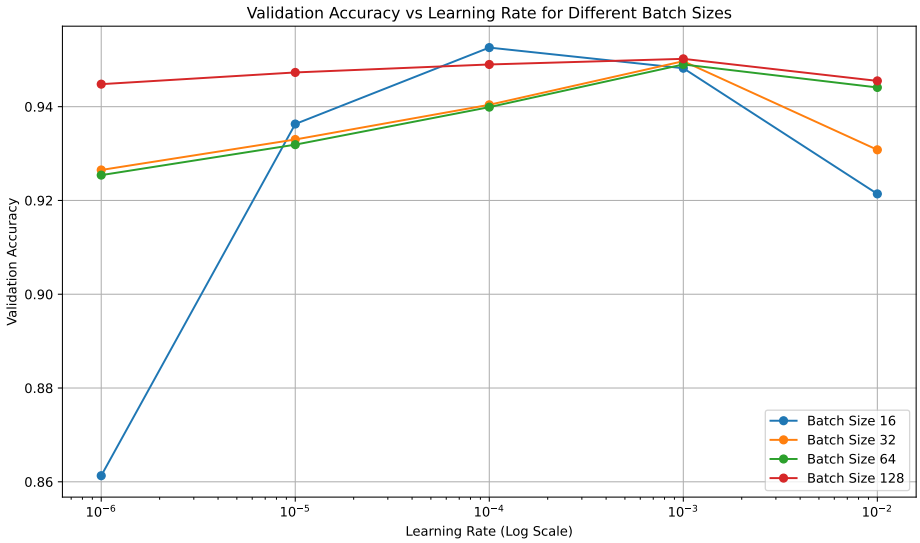


Figure 4.2: ***Grid search validation accuracies across learning rates on the ISOT dataset.*** *A logarithmic x-axis is used to better visualise the explored learning rate range.*

Analysis Key observations from the grid search are summarised below:

- **Best configuration**: Learning rate of 10^{-4} and batch size of 16.
- **Learning rate sensitivity**: Very small learning rates (10^{-6}) led to underfitting; excessively large rates (10^{-2}) induced instability.
- **Batch size effect**: Smaller batch sizes (16) resulted in marginally better generalisation.
- **Final hyperparameters used throughout**: Learning rate = 10^{-4}, batch size = 16, training epochs = 5.

4.2 BERT Base

This section evaluates the three classifiers on the BERT Base encoder (see Section 2.7.2). All models were trained for 5 epochs on the ISOT dataset using the optimised settings described earlier in Section 4.1. In-domain performance is reported on the ISOT test set, with classification errors analysed through confusion matrices. To assess generalisation, each model is further evaluated on the WELFake dataset, which differs in topic and writing style (see Section 3.3).

4.2.1 Fully Connected (FC) Model

The BERT Base + FC model achieved an accuracy of **98.1%** on the ISOT test set, correctly classifying **6609 out of 6735** articles. Class-specific performance metrics are summarised below:

- **True class**: Precision = 0.97, Recall = 0.99, F1-score = 0.98
- **Fake class**: Precision = 0.99, Recall = 0.97, F1-score = 0.98

- **Macro average**: Precision = 0.98, Recall = 0.98, F1-score = 0.98

The confusion matrix is shown in Table 4.2, highlighting a strong concentration of predictions along the diagonal, indicative of high classification accuracy.

Table 4.2: *Confusion matrix for the BERT Base + FC model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3429	93
Fake	33	3180

BERT Base + FC model exhibited strong performance on the ISOT dataset, achieving high precision and recall across both classes. Misclassifications were rare, with a slightly higher error rate when identifying true news articles compared to fake news, suggesting a marginal bias towards detecting deception.

4.2.2 BERTCNN Model

The BERT Base + CNN model achieved an accuracy of **95.0%** on the ISOT test set, correctly classifying **6429 out of 6735** articles. The class-wise performance metrics are summarised below:

- **True class**: Precision = 0.93, Recall = 0.98, F1-score = 0.95
- **Fake class**: Precision = 0.98, Recall = 0.93, F1-score = 0.96
- **Macro average**: Precision = 0.95, Recall = 0.96, F1-score = 0.95

The confusion matrix for this model is shown in Table 4.3, where 306 total misclassifications were recorded: 58 false positives and 248 false negatives.

Table 4.3: *Confusion matrix for the BERT Base + CNN model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3155	58
Fake	248	3274

Although the BERT Base + CNN model performed well, a slight imbalance was observed between precision and recall across classes. The model demonstrated strong precision when predicting fake news but produced a higher number of false negatives, indicating that some fake articles were misclassified as true. This suggests that while local feature extraction through CNN layers was beneficial, it may have been insufficient to fully capture the broader contextual cues present in deceptive content.

4.2.3 FakeBERT Model

The BERT Base + FakeBERT model achieved an accuracy of **99.3%** on the ISOT test set, correctly classifying **6699 out of 6735** articles. Performance was highly balanced across both classes, as shown below:

- **True class**: Precision = 0.99, Recall = 0.99, F1-score = 0.99
- **Fake class**: Precision = 0.99, Recall = 0.99, F1-score = 0.99
- **Macro average**: Precision = 0.99, Recall = 0.99, F1-score = 0.99

The confusion matrix is presented in Table 4.4, with only **15 false positives** and **21 false negatives** recorded.

Table 4.4: *Confusion matrix for the BERT Base + FakeBERT model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3198	15
Fake	21	3501

The BERT Base + FakeBERT model exhibited near-perfect classification on the ISOT dataset, achieving exceptionally high precision and recall across both classes. Nevertheless, concerns regarding overfitting prompted the application of regularisation techniques, as detailed below.

Overfitting Mitigation Despite its excellent in-domain performance, the model demonstrated signs of overfitting during validation experiments. To address this, three variants were implemented, as described in Section 3.6.5:

- **FakeBERT-Base**: No additional regularisation.
- **FakeBERT-Reg1**: Higher dropout, increased weight decay, and simplified classifier structure.
- **FakeBERT-Reg2**: Incorporation of Gaussian noise alongside Reg1 modifications.

Table 4.5 compares the test performance of these variants on the ISOT dataset.

Table 4.5: *Performance of BERT Base + FakeBERT variants with overfitting mitigation on the ISOT dataset.*

Model Variant	Accuracy	Precision	Recall	F1-score
FakeBERT-Base	99.3%	0.994	0.994	0.994
FakeBERT-Reg1	96.5%	0.964	0.972	0.968
FakeBERT-Reg2	96.6%	0.962	0.967	0.964

Although slight reductions in accuracy were observed following regularisation (approximately 2–3%), the variants maintained strong performance. Importantly, as discussed later in Section 4.2.4, these regularised models demonstrated markedly improved generalisation to the WELFake dataset, validating the effectiveness of the overfitting reduction strategies.

4.2.4 Generalisation Testing

To assess the model beyond the ISOT dataset, zero-shot generalisation testing was conducted on the WELFake dataset, which differs substantially in writing style, structure,

and content sources. No additional fine-tuning or adaptation was applied prior to evaluation.

The generalisation performance of all BERT Base models, including the regularised FakeBERT variants, is summarised in Table 4.6. Metrics reported include accuracy, and macro-averaged precision, recall, and F1-score.

Table 4.6: *Generalisation performance of BERT Base models on the WELFake dataset (macro-averaged).*

Model	Accuracy	Precision	Recall	F1-score
FC (Baseline)	88%	0.88	0.88	0.88
CNN (Baseline)	84%	0.87	0.84	0.84
FakeBERT-Base	45%	0.22	0.50	0.31
FakeBERT-Reg1	82%	0.85	0.83	0.82
FakeBERT-Reg2	82%	0.85	0.83	0.82

The results demonstrate that FakeBERT-Base failed to generalise, achieving only 45% accuracy and displaying significant class imbalance. In contrast, both regularised FakeBERT variants maintained substantially stronger performance, with balanced precision and recall scores exceeding 82%. The baseline FC model achieved the highest generalisation accuracy (88%), suggesting that simpler architectures may confer greater sturdiness under domain shift. Although the regularised FakeBERT models incurred a slight reduction in true class performance relative to the baseline, they provided improved detection capabilities for fake news articles.

4.3 BERT Large

The evaluation procedure was repeated for the BERT Large backbone. With a greater number of parameters and richer internal representations, BERT Large was expected to potentially improve performance. However, the larger model size also increased the risk of overfitting. All three architectures were trained for 5 epochs on the ISOT dataset using the same optimised hyperparameters established earlier. The results on the ISOT test set are presented below.

4.3.1 Fully Connected (FC) Model

The BERT Large + FC model achieved an accuracy of **93.7%** on the ISOT test set, correctly classifying **6307 out of 6735** articles. Compared to the BERT Base + FC model, this reflected a modest reduction in performance. The class-wise evaluation metrics are as follows:

- **True class**: Precision = 0.96, Recall = 0.90, F1-score = 0.93
- **Fake class**: Precision = 0.91, Recall = 0.97, F1-score = 0.94
- **Macro average**: Precision = 0.94, Recall = 0.93, F1-score = 0.94

The confusion matrix is shown in Table 4.7, revealing **318 false positives** and **110 false negatives**, resulting in a total of **428 misclassifications**, which is more than three times higher than that observed for the BERT Base + FC model.

Table 4.7: *Confusion matrix for the BERT Large + FC model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	2895	318
Fake	110	3412

While the BERT Large + FC model still achieved high overall accuracy, the increased misclassification rate suggested greater difficulty in adapting to the ISOT dataset compared to BERT Base. In particular, a notable imbalance was observed: the model exhibited stronger precision in identifying true articles but showed comparatively lower recall for fake news, indicating a potential loss in sensitivity towards deceptive content.

4.3.2 BERTCNN Model

The BERT Large + CNN model achieved an accuracy of **98.1%** on the ISOT test set, correctly classifying **6605 out of 6735** articles. This represents one of the strongest performances observed across all model configurations. The class-specific performance metrics are as follows:

- **True class**: Precision = 0.97, Recall = 0.99, F1-score = 0.98
- **Fake class**: Precision = 0.99, Recall = 0.97, F1-score = 0.98
- **Macro average**: Precision = 0.98, Recall = 0.98, F1-score = 0.98

The confusion matrix for this model is presented in Table 4.8, indicating a total of **130 misclassifications**, comprising **23 false positives** and **107 false negatives**.

Table 4.8: *Confusion matrix for the BERT Large + CNN model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3190	23
Fake	107	3415

Overall, the BERT Large + CNN model demonstrated excellent predictive capabilities, achieving a balanced precision-recall profile across classes. Compared to the BERT Large + FC model, the addition of convolutional layers significantly reduced the number of misclassifications, particularly by improving sensitivity to fake news patterns while maintaining strong generalisation within the ISOT domain.

4.3.3 FakeBERT Model

The BERT Large + FakeBERT model (FakeBERT-Reg2) achieved an accuracy of **95.2%** on the ISOT test set, correctly classifying **6413 out of 6735** articles. This configuration

incorporated the same regularisation strategies as used in the BERT Base variant, namely increased dropout (0.5), enhanced weight decay, reduced fully connected layer size, and Gaussian noise injection (Section 3.6.5).

The class-specific performance metrics are presented below:

- **True class**: Precision = 0.92, Recall = 0.99, F1-score = 0.95
- **Fake class**: Precision = 0.99, Recall = 0.92, F1-score = 0.95
- **Macro average**: Precision = 0.95, Recall = 0.95, F1-score = 0.95

The confusion matrix is shown in Table 4.9, with **41 false positives** and **281 false negatives**, resulting in **322 misclassifications** in total.

Table 4.9: *Confusion matrix for the BERT Large + FakeBERT model (FakeBERT-Reg2) on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3172	41
Fake	281	3241

Although the BERT Large + FakeBERT model demonstrated strong overall performance, the number of false negatives was noticeably higher compared to the false positives. This indicates that while the model maintained excellent precision when identifying fake news, it showed reduced sensitivity in detecting true news articles.

4.3.4 Generalisation Testing

The BERT Large models were evaluated on the WELFake dataset to assess performance under domain shift. All models were tested in a zero-shot setting using the same hyperparameters as applied during in-domain training.

The macro-averaged performance across all three architectures is presented in Table 4.10.

Table 4.10: *Generalisation performance of BERT Large models on the WELFake dataset (macro-averaged).*

Model	Accuracy	Precision	Recall	F1-score
FC (Baseline)	81%	0.82	0.82	0.81
CNN (Baseline)	82%	0.84	0.83	0.82
FakeBERT-Reg2	55%	0.61	0.50	0.36

The BERT Large + CNN model demonstrated moderate generalisation capabilities, achieving an accuracy of 82% on WELFake. In contrast, the FakeBERT-Reg2 model exhibited poor generalisation, achieving only 55% accuracy with substantially reduced F1-score. These results suggest that, despite regularisation efforts, the increased capacity of BERT Large models made them more prone to overfitting to the ISOT domain.

4.4 RoBERTa

Finally, the evaluation was extended to models utilising the RoBERTa Base encoder. All three RoBERTa models were trained under identical conditions to ensure comparability with the previous experiments. The results on the ISOT test set are presented below.

4.4.1 Fully Connected (FC) Model

The RoBERTa Base + FC model achieved an accuracy of **98.0%** on the ISOT test set, correctly classifying **6697 out of 6735** articles. Performance remained strong across both true and fake news classes, as summarised below:

- **True class**: Precision = 0.98, Recall = 0.97, F1-score = 0.98
- **Fake class**: Precision = 0.97, Recall = 0.97, F1-score = 0.97
- **Macro average**: Precision = 0.98, Recall = 0.97, F1-score = 0.98

The confusion matrix is shown in Table 4.11, indicating a total of only **30 misclassifications**, comprising **12 false positives** and **18 false negatives**.

Table 4.11: *Confusion matrix for the RoBERTa Base + FC model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3201	12
Fake	18	3504

The RoBERTa Base + FC model demonstrated outstanding classification performance on the ISOT dataset, achieving the lowest misclassification rate among all evaluated FC models. The results confirm that RoBERTa's pre-training strategy provided enhanced robustness and allowed the model to capture subtle distinctions in linguistic patterns between true and fake news articles.

4.4.2 BERTCNN Model

The RoBERTa Base + CNN model achieved an accuracy of **98.0%** on the ISOT test set, correctly classifying **6613 out of 6735** articles. The class-specific performance metrics are as follows:

- **True class**: Precision = 0.96, Recall = 1.00, F1-score = 0.98
- **Fake class**: Precision = 1.00, Recall = 0.97, F1-score = 0.98
- **Macro average**: Precision = 0.98, Recall = 0.98, F1-score = 0.98

The confusion matrix for this model is presented in Table 4.12, showing a total of **132 misclassifications**, composed of **12 false positives** and **120 false negatives**.

Overall, the RoBERTa Base + CNN model exhibited a balanced performance across classes, achieving very high recall for true news and high precision for fake news. Although slightly more false negatives were recorded compared to the RoBERTa FC model,

Table 4.12: *Confusion matrix for the RoBERTa Base + CNN model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3201	12
Fake	120	3402

the results still demonstrated strong reliability within the ISOT domain, confirming the effectiveness of combining convolutional layers with RoBERTa embeddings.

4.4.3 FakeBERT Model

The RoBERTa Base + FakeBERT model achieved an accuracy of **98.0%** on the ISOT test set, correctly classifying **6624 out of 6735** articles. This configuration combined RoBERTa's contextual representations with the regularised FakeBERT classifier, incorporating dropout, weight decay, and Gaussian noise.

The class-specific performance metrics are summarised below:

- **True class**: Precision = 0.97, Recall = 1.00, F1-score = 0.98
- **Fake class**: Precision = 1.00, Recall = 0.97, F1-score = 0.98
- **Macro average**: Precision = 0.98, Recall = 0.98, F1-score = 0.98

The confusion matrix for the model is shown in Table 4.13, indicating **15 false positives** and **96 false negatives**.

Table 4.13: *Confusion matrix for the RoBERTa Base + FakeBERT model on the ISOT dataset.*

True Label	Predicted: True	Predicted: Fake
True	3198	15
Fake	96	3426

The RoBERTa Base + FakeBERT model demonstrated highly balanced classification performance, maintaining near-perfect recall for true news articles and excellent precision for fake news. Although a slightly higher number of false negatives were observed compared to the RoBERTa FC model, the overall performance confirmed the strength of the regularised FakeBERT architecture when applied to RoBERTa embeddings.

4.4.4 Generalisation Testing

To evaluate the model cross-domain, the RoBERTa Base models were tested on the WELFake dataset in a zero-shot setting, without any additional fine-tuning. The macro-averaged performance metrics are presented in Table 4.14.

The RoBERTa Base models demonstrated strong cross-domain generalisation, with all three architectures maintaining accuracy close to 89% on the WELFake dataset. Notably, the FakeBERT-Reg2 model achieved performance equivalent to the fully connected baseline, confirming that the regularisation techniques introduced did not compromise

Table 4.14: *Generalisation performance of RoBERTa Base models on the WELFake dataset (macro-averaged).*

Model	Accuracy	Precision	Recall	F1-score
FC (Baseline)	89%	0.89	0.89	0.89
CNN (Baseline)	88%	0.88	0.88	0.88
FakeBERT-Reg2	89%	0.89	0.89	0.89

generalisability. Overall, the results reinforce RoBERTa's superior pre-training as a key factor for robust fake news detection across differing domains.

4.5 Training Times

Training times varied depending on the model architecture and encoder complexity. Table 4.15 summarises the approximate total training duration for each configuration. All models were trained using two NVIDIA A100 GPUs on the CSF3 high-performance computing cluster.

Table 4.15: *Training times for different model–architecture combinations.*

Model	Rough Training Time (HH:mm)
BERT Base + FC	10:20
BERT Base + CNN	11:10
BERT Base + FakeBERT	12:00
BERT Large + FC	12:40
BERT Large + CNN	13:20
BERT Large + FakeBERT	14:50
RoBERTa Base + FC	11:30
RoBERTa Base + CNN	12:10
RoBERTa Base + FakeBERT	14:00

Overall, careful monitoring of training time and GPU memory usage influenced experimental planning, helping to balance model complexity against available computational resources.

4.6 Comparison Summary

Table 4.16 presents a consolidated view of all models, including their in-domain performance on ISOT and generalisation accuracy on WELFake. Precision, recall, and F1-scores are macro-averaged. A shorter ranking for the best-performing models is shown in Table 4.17.

Table 4.16: *Performance of all models on ISOT test set and WELFake dataset (macro-averaged). Using Reg2 for FakeBERT throughout.*

Model	Accuracy	Precision	Recall	F1-score	WEL Accuracy
BERT Base + FC	98.1%	0.98	0.98	0.98	88%
BERT Base + CNN	95.0%	0.95	0.96	0.95	84%
BERT Base + FakeBERT	96.6%	0.96	0.97	0.96	82%
BERT Large + FC	93.7%	0.94	0.93	0.94	81%
BERT Large + CNN	98.1%	0.98	0.98	0.98	82%
BERT Large + FakeBERT	95.2%	0.95	0.95	0.95	55%
RoBERTa Base + FC	98.0%	0.98	0.97	0.98	89%
RoBERTa Base + CNN	98.0%	0.98	0.98	0.98	88%
RoBERTa Base + FakeBERT	98.0%	0.98	0.98	0.98	**89%**

Table 4.17: *Top 3 performing models based on ISOT and WELFake results (macro-averaged).*

Model	Accuracy	Precision	Recall	F1-score	WEL Accuracy
RoBERTa Base + FakeBERT	98.0%	0.98	0.98	0.98	89%
RoBERTa Base + FC	98.0%	0.98	0.97	0.98	89%
RoBERTa Base + CNN	98.0%	0.98	0.98	0.98	88%

Best In-Domain Performance: Several models achieved 98.0% test accuracy on ISOT, including RoBERTa FC, CNN, FakeBERT, and BERT Large CNN. However, the RoBERTa Base + FakeBERT configuration achieved slightly superior balance across precision, recall, and F1-score.

Best Generalising Model: RoBERTa Base + FakeBERT and RoBERTa Base + FC both reached 89% WELFake accuracy, the highest across all models. FakeBERT maintained slightly stronger class balance, demonstrating that regularisation can significantly enhance cross-domain robustness.

Trade-offs: Simpler architectures such as FC achieved competitive accuracy with lower computational cost but were more sensitive to domain shift. FakeBERT models, while more complex, consistently improved generalisation, particularly when paired with RoBERTa. Conversely, BERT Large + FakeBERT showed that complexity without sufficient regularisation may degrade cross-domain performance.

Conclusion: The RoBERTa Base + FakeBERT model demonstrated the best performance, combining excellent in-domain accuracy (98%) with strong generalisation (89%). These findings highlight the critical importance of both encoder selection and careful model design when building fake news detection systems for real-world deployment.

5

Discussion and Conclusion

5.1 Overview of Findings

Table 4.16 summarises the performance of all transformer-based models across both in-domain (ISOT) and out-of-domain (WELFake) datasets. RoBERTa-based models consistently outperformed BERT Base and BERT Large, regardless of classifier type. RoBERTa+FakeBERT and RoBERTa+FC achieved the highest WELFake accuracy (89%), while maintaining strong ISOT performance (98%).

RoBERTa+CNN also performed competitively, reaching 98% on ISOT and 88% on WELFake. By contrast, BERT Base models demonstrated slightly weaker generalisation. For example, BERT Base+FC achieved 88% on WELFake, while BERT Base+FakeBERT fell to 82%. FakeBERT enhancements were less effective for BERT Large, where WELFake accuracy dropped significantly to 55%.

BERT Large emerged as the weakest encoder overall. Despite its higher parameter count, BERT Large+FC reached only 93.7% accuracy on ISOT and 81% on WELFake. This suggests that larger model capacity did not translate to better performance, likely due to overfitting and the limited size of ISOT.

Classifier architecture also impacted generalisation. While fully connected (FC) heads offered strong in-domain accuracy, CNN and FakeBERT heads demonstrated strong cross-domain performance, particularly when paired with RoBERTa. The FakeBERT design, using parallel convolutional filters, captured varied local patterns (e.g., clickbait phrases, emotional expressions), promoting better generalisation. These trends are illustrated in Figure 5.1.
In summary:

- **RoBERTa consistently outperformed BERT Base and BERT Large**, particularly in cross-domain settings.
- **Classifier design influenced generalisation**: CNN and FakeBERT heads offered more accurate performance.
- **BERT Large was the least effective**, likely due to overfitting.
- **FakeBERT improved RoBERTa's robustness**, but offered limited gains with BERT Large.

Figure 5.1: ***Accuracy of model–architecture combinations on ISOT and WELFake datasets.*** *RoBERTa-based models consistently achieved the strongest results, with CNN and FakeBERT architectures improving generalisation.*

5.2 Strengths of the FakeBERT Approach

Our findings highlight several strengths of the FakeBERT architecture, particularly when paired with RoBERTa. Most notably, FakeBERT improved generalisation across domains. The RoBERTa+FakeBERT model achieved 98% accuracy on the ISOT test set and 89% on WELFake, matching the highest cross-domain performance among all models tested (Table 4.17). This suggests that FakeBERT's convolutional layers helped capture domain-invariant cues, such as stylistic markers and common deceptive phrases.

FakeBERT also enabled multi-resolution text analysis. While standard transformers rely on token-level embeddings, FakeBERT's CNN layers applied filters of varying widths, detecting unigrams, bigrams, and trigrams. These local patterns, often weak individually but stronger together, likely contributed to the consistently high F1-scores (0.98) observed for RoBERTa+FakeBERT.

Finally, FakeBERT introduced an implicit regularisation effect. Although it added parameters, its structured design encouraged the model to focus on meaningful local features, reducing the risk of overfitting to dataset-specific noise. This is reflected in the comparison between RoBERTa+FC and RoBERTa+FakeBERT: while both achieved 98% on ISOT, FakeBERT generalised slightly better to WELFake (89% vs. 88%).

However, Figure 5.1 also highlights a limitation: FakeBERT did not improve all encoders. When combined with BERT Large, FakeBERT achieved only 55% accuracy on WELFake, the lowest cross-domain result observed. This suggests that combining convolutional layers with very large transformer outputs may lead to instability or overfitting, particularly with limited training data.

5.3 Limitations of the Present Study

While this study yielded promising results, several limitations must be acknowledged.

First, the analysis was confined to two English-language datasets (ISOT and WELFake) both comprising formal news articles with a political or general focus. Although the datasets differ in origin and style, their similar format limits text diversity. As such, the findings may not generalise to other domains such as health misinformation, social media posts, or multimodal fake news incorporating images or videos.

Second, the models were evaluated only in English. Fake news is a global phenomenon, yet this study did not explore multilingual scenarios. It remains unclear whether the performance trends observed, particularly the advantages of RoBERTa+FakeBERT, would hold in languages such as Spanish, Hindi, or Mandarin. Addressing this would require access to cross-lingual datasets and the use of multilingual transformer models.

Finally, computational constraints (mentioned in Table 4.15) limited deeper experimentation. BERT Large, in particular, posed significant challenges due to its memory demands, contributing to instability in its FakeBERT variant, which achieved only 55% accuracy on WELFake (Table 4.16, Figure 5.1). Resource limitations also prevented repeated trials and exhaustive hyperparameter searches.

In summary, while this study provides valuable insights into model performance for fake news detection, its conclusions are bounded by dataset scope, language coverage, and computational capacity. Future work should address these limitations through broader datasets, multilingual testing, and interpretability-focused analysis.

5.4 Personal Reflection

This project provided a valuable opportunity for both technical and professional development. Motivated by the societal impact of fake news, the work quickly revealed the complexity involved in applying state-of-the-art natural language processing (NLP) techniques. Engaging with foundational literature on transformers and deception detection required a steep learning curve, particularly in mastering concepts such as self-attention and model fine-tuning.

Hands-on model development was central to the learning process, involving the implementation of transformer architectures using PyTorch and the Hugging Face Transformers library. Practical challenges, such as managing GPU memory limitations, addressing training instability, and integrating convolutional layers with transformer outputs, deepened understanding of modern deep learning frameworks.

A key breakthrough was recognising that the overfitting observed in BERT Large was a modelling limitation rather than a coding error. This insight led to the application of techniques such as layer freezing and data order shuffling, reinforcing the importance of analytical problem-solving and experimental design. Regular supervision meetings also supported the development of critical reasoning and scientific communication

skills.

Throughout the project, time management, academic writing, and evaluation abilities were progressively strengthened. Skills in interpreting model behaviour, comparing architectures, and explaining performance trade-offs improved over time. Ultimately, this project not only resulted in a rigorous comparative study but also instilled greater confidence and competence in addressing complex machine learning challenges for future research or professional work.

5.5 Future Developments

Several promising directions exist for extending this work. Based on the insights and limitations discussed, the following developments are suggested, each suitable for exploration at an undergraduate or early research level:

- **Multilingual Fake News Detection:** Extending this study by training models on fake news datasets in other languages would evaluate whether transformer architectures generalise across linguistic and cultural boundaries. Models such as XLM-RoBERTa [6] or mBERT could be explored. Datasets like FakeNewsAMT for Arabic or LIAR-PLUS for multiple languages offer starting points.
- **Incorporating Social Context:** Future systems could combine textual features with metadata such as source credibility, publishing history, or user engagement metrics (e.g., shares, comments). Work by Shu et al. [49] highlights how social context improves fake news detection, especially when textual signals are ambiguous. Graph neural networks (GNNs) or multimodal fusion models could be investigated for this purpose.
- **Detecting Rephrased and Evasive Fake News:** Fake news writers often reword existing content to evade detection. Exploring adversarial training techniques [59] or contrastive learning [13] could improve model robustness to rephrased or paraphrased deceptive content. Techniques such as synonym substitution or sentence shuffling could be used to generate adversarial examples during training.
- **Exploring Lighter Transformer Variants:** Alternative models such as DeBERTa [16] and ALBERT [28] offer improved efficiency and generalisation through techniques like disentangled attention and parameter sharing. Investigating these models could lead to better trade-offs between performance, training time, and computational cost.
- **Semi-supervised and Weakly Supervised Learning:** Building reliable fake news detectors often requires large labelled datasets, which are costly to obtain. Semi-supervised learning approaches (e.g., pseudo-labelling) [65] or weak supervision using noisy labels could enable models to leverage unlabelled news data effectively. This area has shown promise for expanding dataset diversity without extensive manual annotation.

Each of these extensions would address key challenges highlighted in Figure 1.2,

including generalisability, hybrid modelling, interpretability, and resilience to adversarial manipulation. Several open datasets, pre-trained multilingual models, and reproducible training frameworks are publicly available, providing an accessible starting point for future researchers.

5.6 Conclusion

This dissertation addressed the challenge of fake news detection through a comparative evaluation of transformer-based architectures, BERT Base, BERT Large, and RoBERTa, each combined with three classifier heads: Fully Connected (FC), CNN, and the hybrid FakeBERT design. The primary objective was to identify model–architecture combinations that most effectively distinguished real from fake news, particularly with regard to cross-domain generalisation.

Experiments conducted on the ISOT (in-domain) and WELFake (out-of-domain) datasets demonstrated that RoBERTa paired with the FakeBERT classifier delivered the best overall performance. This combination effectively captured domain-invariant linguistic features, achieving strong results on both datasets. By contrast, BERT Large did not consistently outperform BERT Base, highlighting the risks of overfitting in large models when trained on relatively small datasets.

The integration of convolutional layers and dropout regularisation within the FakeBERT architecture proved valuable, improving the model's resilience to domain shifts. A key contribution of this work is the novel exploration of RoBERTa combined with a FakeBERT-style head, alongside rigorous cross-dataset evaluation, an approach not previously detailed in the literature.

Practically, the findings support the use of pre-trained models combined with targeted feature extraction mechanisms to build more reliable fake news detection systems. Such systems could assist news platforms and social media networks in identifying misleading content, contributing to wider efforts to address misinformation.

Although automated detection alone cannot eliminate fake news, this study illustrates the power of modern natural language processing models in mitigating the problem. It provides a foundation for future research, encouraging developments in multilingual detection, social-context modelling, and greater model interpretability, as outlined in Section 5.5.

Bibliography

[1] Merriam-Webster. (n.d.) "In Merriam-Webster.com dictionary". In: (2024). URL: https://www.merriam-webster.com/dictionary/news.

[2] H. Ahmed, I. Traore, and S. Saad. "Detecting opinion spams and fake news using text classification". In: *Security and Privacy* 1.1 (2018), e9. URL: https://doi.org/10.1002/spy2.9.

[3] Esma Aïmeur, Sara Amri, and Gilles Brassard. "Fake news, disinformation and misinformation in social media: a review". In: *Social Network Analysis and Mining* 13.1 (2023), p. 30. URL: https://doi.org/10.1007/s13278-023-01028-5.

[4] Muhammad Ali, Syed Afaq Shah, and Riaz Nawaz. "Fake News Detection Techniques on Social Media: A Survey". In: *Complexity* 2022 (2022), pp. 1–19. URL: https://doi.org/10.1155/2022/6072084.

[5] S. Anderson. *Individuals using the Internet*. URL: https://data.worldbank.org/indicator/IT.NET.USER.ZS.

[6] Alexis Conneau et al. "Unsupervised Cross-lingual Representation Learning at Scale". In: *Proceedings of ACL* (2020).

[7] DataReportal. *Digital 2024: Global Overview Report*. https://datareportal.com/reports/digital-2024-global-overview-report. Accessed: 2025-04-16. 2024.

[8] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805.

[9] Richard Ling Edson C. Tandoc Jr. Zheng Wei Lim. "DEFINING FAKE NEWS A typology of scholarly definitions". In: *Digital* (2018). URL: https://www.tandfonline.com/doi/full/10.1080/21670811.2017.1360143.

[10] Hugging Face. *Chapter 6: Transformer Models*. Accessed: 2025-01-27. 2025. URL: https://huggingface.co/learn/nlp-course/en/chapter6/6.

[11] Áureo Figueira, Nuno Guimaraes, and Luís Torgo. "Current State of the Art to Detect Fake News in Social Media: Global Trendings and Next Challenges". In: *Proceedings of the 14th International Conference on Web Information Systems and Technologies (WEBIST)*. SciTePress. 2018. URL: https://www.scitepress.org/Papers/2018/71885/71885.pdf.

[12] KFF (Kaiser Family Foundation). *COVID-19 Vaccine Misinformation and Uptake*. https://www.kff.org/coronavirus-covid-19/poll-finding/covid-19-vaccine-misinformation-and-uptake/. Accessed: 2025-04-16. 2021.

[13] Tianyu Gao, Xingcheng Yao, and Danqi Chen. "SimCSE: Simple Contrastive Learning of Sentence Embeddings". In: *Proceedings of EMNLP*. 2021.

[14] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. https://www.deeplearningbook.org/. MIT Press, 2016.

[15] Sufian Hakak et al. "Propagation of fake news on social media: challenges and opportunities". In: *Advances in Social Networks Analysis and Mining (ASONAM)*. Springer, 2020, p. 345. URL: https://link.springer.com/chapter/10.1007/978-3-030-66046-8_28.

[16] Pengcheng He et al. "DeBERTa: Decoding-enhanced BERT with Disentangled Attention". In: *arXiv preprint arXiv:2006.03654* (2021).

[17] Minh Hoque. *A Comprehensive Overview of Transformer-based Models: Encoders, Decoders, and More*. https://medium.com/@minh.hoque/a-comprehensive-overview-of-transformer-based-models-encoders-decoders-and-more-e9bc0644a4e5. Accessed: 2024-11-12. 2023.

[18] Christy G. Horner et al. "Emotions: The unexplored fuel of fake news on social media". In: *Fake News on Social Media*. This study highlights how emotionally charged messages tend to be shared more often due to their powerful impact on sharing behaviors. Routledge, 2023. URL: https://www.taylorfrancis.com/chapters/edit/10.4324/9781003433934-7/emotions-unexplored-fuel-fake-news-social-media-christy-galletta-horner-dennis-galletta-jennifer-crawford-abhijeet-shirsat.

[19] Md Nur Hossen et al. *WELFake Dataset for Fake News Detection*. https://www.kaggle.com/datasets/mdnurhossen/welfake-dataset. Accessed on 1 April 2025. 2022.

[20] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR. 2015, pp. 448–456.

[21] University of Victoria ISOT Research Lab. *Fake News Detection Datasets – ISOT Lab*. https://onlineacademiccommunity.uvic.ca/isot/2022/11/27/fake-news-detection-datasets/. Accessed on 1 April 2025. 2022.

[22] University of Victoria ISOT Research Lab. *ISOT Fake News Dataset*. https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset. Accessed on 1 April 2025. 2022.

[23] S. Mo Jang. "A computational approach for examining the roots and spreading patterns of fake news: Evolution tree analysis". In: *Computers in Human Behavior* (2018). URL: https://www.sciencedirect.com/journal/computers-in-human-behavior/vol/162/suppl/C.

[24] Daniel Jurafsky and James H. Martin. *Speech and Language Processing (3rd ed. draft)*. Available at https://web.stanford.edu/~jurafsky/slp3/. 2021.

[25] Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. "FakeBERT: Fake news detection in social media with a BERT-based deep learning approach". In: *Multimedia Tools and Applications* 80 (2021). URL: https://doi.org/10.1007/s11042-020-10183-2.

[26] Nitish Shirish Keskar et al. "On large-batch training for deep learning: Generalization gap and sharp minima". In: *arXiv preprint arXiv:1609.04836* (2017).

[27] Yoon Kim. "Convolutional Neural Networks for Sentence Classification". In: *EMNLP* (2014). URL: https://aclanthology.org/D14-1181/.

[28] Zhenzhong Lan et al. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations". In: *Proceedings of ICLR* (2020).

[29] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL]. URL: https://arxiv.org/abs/1907.11692.

[30] Ilya Loshchilov and Frank Hutter. "Decoupled Weight Decay Regularization". In: *International Conference on Learning Representations (ICLR)*. 2019. arXiv: 1711.05101. URL: https://arxiv.org/abs/1711.05101.

[31] Szczepanski M. "New explainability method for BERT based model in fake news detection". In: *Sci Rep* (2021). URL: https://doi.org/10.1038/s41598-021-03100-6.

[32] Mr. M.S.V.V.Ramesh. "FAKE NEWS DETECTION USING MACHINE LEARNING". In: *International Research Journal of Modernization in Engineering Technology and Science* (2022). ISSN: 2582-5208. URL: https://www.irjmets.com/uploadedfiles/paper//issue_1_january_2022/18707/final/fin_irjmets1643710144.pdf.

[33] University of Manchester. *Research IT Services - Computational Shared Facility (CSF3)*. Accessed: 2025-03-11. 2025. URL: https://ri.itservices.manchester.ac.uk/csf3/.

[34] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[35] T. Mu. *COMP34711 Lecture Notes on LLM Training and BERT*. Lecture notes. 2023.

[36] Raymond S. Nickerson. "Confirmation bias: A ubiquitous phenomenon in many guises". In: *Review of General Psychology* 2.2 (1998), pp. 175–220.

[37] Adam Paszke, Sam Gross, and Massa. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019. URL: https://papers.nips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[38] Fabian Pedregosa and Varoquaux. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830. URL: http://jmlr.org/papers/v12/pedregosa11a.html.

[39] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "GloVe: Global vectors for word representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543.

[40] Matthew E. Peters et al. *Deep contextualized word representations*. 2018. arXiv: 1802.05365 [cs.CL]. URL: https://arxiv.org/abs/1802.05365.

[41] Raghav Prabhu. *Understanding of Convolutional Neural Network (CNN) — Deep Learning*. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148. Accessed: 2025-04-14. 2018.

[42] Ye Qi et al. "When and Why Are Pre-Trained Word Embeddings Useful for Neural Machine Translation?" In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Ed.

by Marilyn Walker, Heng Ji, and Amanda Stent. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 529–535. URL: https://aclanthology.org/N18-2084/.

[43] Hannah Rashkin et al. "Truth of varying shades: Analyzing language in fake news and political fact-checking". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2017, pp. 2931–2937. URL: https://aclanthology.org/D17-1317/.

[44] Natali Ruchansky, Sungyong Seo, and Yan Liu. "CSI: A Hybrid Deep Model for Fake News Detection". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM)*. 2017, pp. 797–806. URL: https://dl.acm.org/doi/10.1145/3132847.3132877.

[45] Fadi Safieddine. "Chapter 1: History of Fake News". In: Feb. 2020. ISBN: 9781786614223.

[46] Victor Sanh et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. 2019. arXiv: 1910.01108 [cs.CL]. URL: https://arxiv.org/abs/1910.01108.

[47] Steven Seidenberg. "Lies and Libel: Fake News Lacks Straightforward Cure". In: *ABA Journal* (July 2017). URL: https://www.abajournal.com/magazine/article/fake_news_libel_law/history_fake_news.

[48] Upasna Sharma and Jaswinder Singh. "A comprehensive overview of fake news detection on social networks". In: *Social Network Analysis and Mining* 14.120 (2024). DOI: 10.1007/s13278-024-01280-3. URL: https://link.springer.com/article/10.1007/s13278-024-01280-3.

[49] Kai Shu et al. "Beyond News Contents: The Role of Social Context for Fake News Detection". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 13.2 (2019), pp. 1–25.

[50] Kai Shu et al. "Fake News Detection on Social Media: A Data Mining Perspective". In: *SIGKDD Explor. Newsl.* 19.1 (Sept. 2017), pp. 22–36. ISSN: 1931-0145. URL: https://doi.org/10.1145/3137597.3137600.

[51] Kai Shu et al. "FakeNewsNet: A Data Repository with News Content, Social Context and Dynamic Information for Studying Fake News on Social Media". In: *Companion Proceedings of The Web Conference 2018*. 2018. URL: https://dl.acm.org/doi/10.1145/3184558.3191529.

[52] Marina Sokolova and Guy Lapalme. "A systematic analysis of performance measures for classification tasks". In: *Information Processing & Management* 45.4 (2009), pp. 427–437.

[53] Nitish Srivastava et al. "Dropout: A simple way to prevent neural networks from overfitting". In: *Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[54] Quandt T. "Fake News". In: *Forms of Journalism* (2019). URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118841570.iejs0128.

[55] Hugging Face Team. *BERT 101: A Comprehensive Overview*. https://huggingface.co/blog/bert-101. Accessed: 2024-12-07. 2023.

[56] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[57] S. Vosoughi, D. Roy, and S. Aral. “The spread of true and false news online”. In: *Science* 359.6380 (2018), pp. 1146–1151.

[58] Alex Wang et al. *GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding*. 2019. arXiv: 1804.07461 [cs.CL]. URL: https://arxiv.org/abs/1804.07461.

[59] Di Wang et al. “InfoDetector: A Framework for Detecting Fake News through Multi-modal Content Understanding”. In: *Proceedings of AAAI*. 2020.

[60] William Y. Wang. “"Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection”. In: *Proceedings of ACL 2017*. 2017, pp. 422–426. URL: https://aclanthology.org/P17-2067/.

[61] Andrew Ward et al. “Naive realism in everyday life: Implications for social conflict and misunderstanding”. In: *Values and Knowledge*. Ed. by Elliot Turiel E. Reed and Thomas Brown. Lawrence Erlbaum Associates, 1997, pp. 103–135.

[62] CAROL A. WATSON. “Information Literacy in a Fake/False News World: An Overview of the Characteristics of Fake News and its Historical Development”. In: *International Journal of Legal Information* (2018). URL: https://www.cambridge.org/core/journals/international-journal-of-legal-information/article/information-literacy-in-a-fakefalse-news-world-an-overview-of-the-characteristics-of-fake-news-and-its-historical-development/674B5DCB6A3106FC50B5DE1CCF0646AA.

[63] Karl R. Weiss, Taghi M. Khoshgoftaar, and Dingding Wang. “A survey of transfer learning”. In: *Journal of Big Data* 3.1 (May 2016). URL: https://link.springer.com/content/pdf/10.1186%2Fs40537-016-0043-6.pdf.

[64] Yonghui Wu and Mike Schuster. *Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. 2016. arXiv: 1609.08144 [cs.CL]. URL: https://arxiv.org/abs/1609.08144.

[65] Qizhe Xie et al. “Self-training with Noisy Student improves ImageNet classification”. In: *Proceedings of CVPR* (2020).

[66] Wenpeng Yin et al. “Comparative study of CNN and RNN for natural language processing”. In: *arXiv preprint arXiv:1702.01923* (2017).

[67] Chiyuan Zhang et al. “Understanding deep learning requires rethinking generalization”. In: *arXiv preprint arXiv:1611.03530* (2017).

[68] Xinyi Zhou and Reza Zafarani. “Fake News: A Survey of Research, Detection Methods, and Opportunities”. In: *arXiv preprint arXiv:1812.00315* (2018). URL: https://arxiv.org/abs/1812.00315.