# Enhancing Fake News and Rumor Detection Using Metadata, Context, and User Interaction with a Hybrid GCN-Transformer

Lucas Romulo Zuin Gigli

A Thesis Submitted in Partial Fulfilment

of the requirements for the

Degree of

Master of Science in Data Analytics


August 2025

Supervisor: Maqsood Hussain

# List of Figures



# List of Tables

## List of Equations

## Abstract:

The increasing influence of social media has significantly impacted how news spreads among users, making the detection of misinformation, including fake news and rumours, a critical task. Previous research has explored content-based and metadata-driven approaches, while recent advancements have leveraged Graph Neural Networks (GNNs) such as GCNs, GATs, and SAGE to analyse user interactions and propagation patterns. This thesis investigates the effectiveness of a Hybrid Graph-Transformer Convolutional Neural Network (GCN-Transformer) for fake news detection, combining graph-based learning with Transformer attention mechanisms. Through in-depth data analysis and advanced detection techniques, this model aims to enhance predictive performance and mitigate the spread of misinformation.

**Keywords:** Fake News Detection, Misinformation Classification, Social Media Analysis, Graph Neural Networks (GNN), GCN-Transformer, Semi-Supervised Learning, Attention Mechanisms, Natural Language Processing (NLP), BERT Embeddings.

## Acknowledgements

I would like to sincerely thank my family for their unwavering support and encouragement throughout this journey. Their belief in me has been a constant source of strength and motivation.

I am also deeply grateful to my friends for their support, whether through moments of encouragement, much-needed distractions, or simply their presence during challenging times. Their companionship has made this process far more manageable.

To my classmates, I truly appreciate the shared discussions, late-night problem-solving, and collective effort that enriched this experience. Learning alongside you has been invaluable.

Finally, I extend my gratitude to everyone who contributed to this work in any way—through guidance, constructive feedback, or simply their support along the way. Your contributions have played a meaningful role in this achievement.

# Table of Contents

# 1. Introduction:

## 1.1. Problem Statement

Social media platforms like Twitter, Facebook, and YouTube have become essential in people's daily lives, providing a space for communication, content sharing, and news consumption. While these platforms offer many benefits, they have also facilitated the rapid spread of misinformation and fake news, making it difficult for users to differentiate between factual and misleading content.

A significant portion of the population now relies on social media as a primary source of news (Holcomb, Gottfried and Mitchell, 2013) yet research shows that false information spreads faster than verified facts. The virality of fake news is driven by algorithmic recommendations, user engagement, and the tendency of individuals to share emotionally charged content without verifying its authenticity. This leads to serious societal consequences, including political polarisation, distrust in media, and misinformation about critical issues such as public health and elections.

Despite efforts to combat fake news, several challenges remain:

- **Evolving misinformation tactics** – Fake news creators continuously refine their strategies to evade detection.
- **Diverse content formats** – Misinformation is no longer limited to text; it now includes manipulated images, deepfake videos, and multimodal content.
- **Propagation-driven influence** – Fake news often gains credibility through repeated exposure and social validation, making it crucial to analyse how misinformation spreads within networks.

Current detection models struggle with these challenges, as many rely solely on text analysis or content-based features, overlooking the role of user interactions and information propagation in misinformation dissemination. To address this, this study introduces a BGCN-Transformer model, which integrates BERT for text analysis, Graph Convolutional Networks (GCN) for user interaction modelling, and Transformer-based attention mechanisms to improve fake news detection accuracy. This hybrid approach enhances the ability to detect misinformation by leveraging both linguistic context and social network structure, making it more adaptable to the evolving nature of fake news.

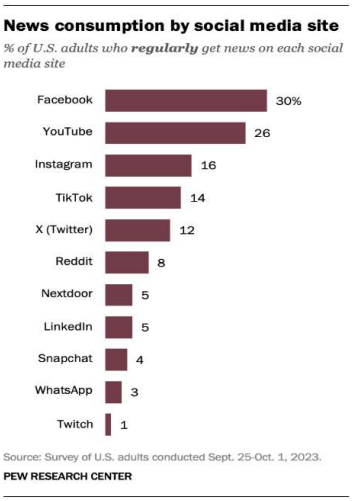



*Figure 1: News consumption through social media* ***(Holcomb, Gottfried and Mitchell, 2013)***

## 1.2. Research Motivation

The motivation for this thesis comes from the harmful effects that misinformation causes in our society. Fake news can mislead public opinion, create untrusted behaviour towards the media and directly influence critical areas like elections and public health.

This research aims to improve existing traditional methods for detecting and controlling misinformation and help to create a more trustworthy digital space.

Fake news tactics are always evolving, so there's an ongoing need for adaptable detection systems.

## 1.3. Research Objectives

This study aims to develop a BGCN-Transformer model for fake news detection, incorporating text-based, metadata, and graph-based features. The model follows a structured pipeline:

1. **BERT for text feature extraction** to capture linguistic patterns.
2. **Metadata mining** to extract all relevant user and content attributes.
3. **GCN layer** to model user interactions and metadata relationships.
4. **Transformer-based module** incorporating multi-head self-attention and feedforward layers to refine learned representations before classification.

This hybrid approach allows for effective information propagation while preserving textual context and structural dependencies, addressing key challenges in fake news detection.

## 1.4. Research Contribution

This research explores fake news and rumour detection through a structured pipeline, beginning with data collection and preprocessing, including BERT for text feature extraction. To establish a strong baseline, various machine learning and deep learning models were evaluated. A graph structure was then constructed to represent relationships between users.

Initially, a Graph Convolutional Network (GCN) was implemented to assess the effectiveness of graph-based learning. Building upon this foundation, this work introduces a Hybrid BGCN-Transformer model, which integrates Graph Neural Networks (GNNs) and Transformers to enhance feature extraction and improve classification accuracy.

## 1.5. Challenges in Fake News Detection

Fake News detection is a complex field with several challenges, such as:

- **Defining Fake News**: Not all information that might sound misleading can be considered Fake news. It can contain satire, biased reporting, or entirely fabricated stories. This wide range of content makes it difficult to set clear criteria for what can be classified as Fake news.
- **Developing new tactics:** Fake news creators constantly update their strategy to avoid existing pre-trained models.
- **Dataset**: Labelling fake news data is quite time-consuming, and collecting correctly labelled ones can be problematic. Also, the content can vary from simple texts to multimodal forms like video, photo, text, and audio combined
- **Virality**: Fake news often spreads more quickly and widely than factual information, especially if it triggers an emotional response. Even if detected, misinformation may have already influenced a broad audience.

- **Ethical and Political Implications**: Controlling fake news raises concerns about censorship and freedom of speech. Mistakenly labelling true content as fake can have a big impact on the platform's responsible trust.

### 1.6. Thesis Structure:

This thesis is organized as follows:

**Chapter 1 – Introduction:**

Introduces the problem of fake news, its impact on society, and the challenges in detection. It presents the research objectives, motivation, and contributions, outlining the need for a hybrid BGCN-Transformer model.

**Chapter 2 – Literature Review:**

Reviews existing fake news detection techniques, from traditional machine learning to deep learning, graph-based methods, and transformers. Identifies limitations in current approaches, leading to the justification for a hybrid model.

**Chapter 3 – Data Collection & Preprocessing:**

Describe the dataset used, its structure, challenges, and legal & ethical considerations. Covers data preprocessing techniques, including text embeddings (BERT), metadata processing, and graph construction to prepare data for model training.

**Chapter 4 – Methodology:**

This section briefly covers machine learning (SVM, XGBoost) and artificial neural networks (FFNN, RNN) before focusing on graph-based models (GCN, GNNs). The BGCN-Transformer is explored in detail, highlighting its combination of GCN for propagation learning and Transformers for global attention to enhance fake news detection.

**Chapter 5 – Experiments & Results:**

Presents the experimental setup, evaluation metrics, and performance comparison of different models, highlighting how the BGCN-Transformer model improves fake news classification.

**Chapter 6 – Discussion:**

Analyzes key findings, strengths, and limitations of the approach, discussing the real-world implications and potential improvements.

**Chapter 7 – Conclusion:**

Summarizes the research contributions, key findings, and limitations, offering recommendations for future research to enhance misinformation detection.

## 2. Literature Review and Background Study

### 2.1. Fake News Definition – Understanding Misinformation

Fake news/rumour is false information presented as real to mislead audiences. This type of deception is designed to look genuine, often mixing with real news to manipulate public opinion or behaviour on social media (Aïmeur, Amri and Brassard, 2023).

As for Phan (Phan, Nguyen and Hwang, 2023), fake news is intentionally false or misleading information designed to manipulate public perception, often spread through social media and other online platforms.

According to (Allcott and Gentzkow, 2017), fake news consists of intentionally false stories crafted to mislead, often for political purposes. Unlike simple mistakes, fake news involves deliberate deceptions, though it doesn't include satire, rumours, or biased yet factual reporting. Social media plays a major role in the rapid spread of fake news, providing an easy environment for circulation and creating insulated groups where users are rarely exposed to different and opposing points of view. The design of social benefits the spread of misinformation, extending its impact and reach.

(Pennycook and Rand, 2021) examines why people believe fake news, highlighting cognitive processing rather than political bias as the main factor. Key points include:

• **Cognitive Limitations**: People often struggle to identify fake news due to limited critical reasoning, poor background knowledge, and familiarity with content rather than its truthfulness.

• **Use of Heuristics**: Mental shortcuts, like trusting familiar or seemingly credible sources, can lead to false beliefs, especially when content is repeatedly encountered.

• **Belief vs. Sharing**: Many share news they don't necessarily believe, often due to inattention rather than intent to deceive. Simple prompts for accuracy can reduce this behaviour.

• **Interventions**: Accuracy prompts can encourage people to share more thoughtfully by tapping into their natural ability to recognise misinformation. This reduces dependence on constant fact-checking, allowing for more scalable solutions.

## 2.2. Trends in Fake News Propagation and Classification

In (Adriani, 2019)work, it examined how fake news has grown alongside technology lately, focusing on tech developments and psychological factors that boost its dissemination. Such as:

**Technological Advances:** Tools like AI and deep learning now allow realistic audio and video manipulation, creating "deepfakes" where public figures appear to say or do things they haven't.

**Psychological Factors:** People frequently share content without reading or verifying sources, driven by confirmation bias, favouring information matching their beliefs.

**Global Efforts and Challenges:** Several Countries are handling fake news through laws, education, and validation networks.

**Future Threats:** Artificial Intelligence may soon automate fake content creation on a large scale, making it even harder to identify and control its dissemination.

In (Zhou and Zafarani, 2018)extensive survey, fake news detection methods were described as follows:

**Knowledge-based detection** involves fact-checking by assessing if the information aligns with pre-established facts or trusted databases.

**Style-based Detection** analyses the language and tone of the article, assuming that Fake News often uses a particular style structure.

**Propagation-based** detection analyses how fake news spreads over social media by identifying the dissemination patterns.

**Source-based detection** checks the credibility of the news source itself, examining historical reputation and information accuracy.

## 2.3. Data Analytics for Misinformation Detection

The role of data analytics in misinformation detection has grown significantly as researchers develop data-driven methods to analyse text, user behaviour, and network structures. Studies have uncovered key insights into how misinformation spreads and how it can be detected by leveraging statistical models, feature engineering, and machine learning analytics.

## 2.4. Early Methods for Misinformation Classification

Early studies relied mostly on textual analysis through machine learning algorithms. These methods, unlike the multimodal, focus entirely on linguistic and statistical features

(Parikh and Atrey, 2018) classified in its survey the text-based approaches as simply the extraction of and analysis of linguistic features with some primary methods described:

**Linguistic Features**: These methods analyse word patterns like unigrams and bigrams, using TF-IDF (Term Frequency-Inverse Document Frequency) to highlight key terms that might indicate fake news.

**Punctuation**: The frequent use of punctuation, such as multiple exclamation points, can signal sensationalism, often associated with misleading news.

**Psycho-Linguistic Features**: Tools like LIWC (Linguistic Inquiry and Word Count) examine psychological aspects, such as emotional tone and cognitive complexity, that might reveal manipulative language common in fake news.

**Readability Metrics**: Features like syllable count, sentence length, and paragraph structure help assess readability. Fake news may use simpler sentence forms for easy, quick consumption.

**Syntax**: Some methods Use Context-Free Grammar (CFG) to examine sentence structure and production rules to spot fake news based on how the text is organized and constructed.

In machine learning, these linguistic features are used to build the dataset. Each news article is broken down into these feature values (like TF-IDF scores, punctuation frequency, and readability scores) and labelled as fake or real. The machine learning model is then trained to recognize patterns in these features typical of fake news. By learning these patterns, the model can apply this knowledge to classify new articles accurately. Combining feature extraction with machine learning allows these language-based indicators to become powerful tools for predicting whether news content is authentic.

(Khanam *et al.*, 2021) explored how machine learning methods enhance fake news detection by extracting and organizing linguistic features. Utilizing Python's scikit-learn library, they apply tools like CountVectorizer and TfidfVectorizer to capture patterns in the text. Essential steps include breaking down text into tokens, removing common but unhelpful words (stopwords), and stemming words to their base forms, making the text easier for models to analyse. Classifiers such as Naive Bayes, Random Forest, and SVM are then used to recognize linguistic patterns often appearing in fake news, drawing on structured data from these text-processing steps. Their study highlights how pre-processed text, combined with robust classifiers, significantly improves detection accuracy.

Similarly, (Sharma, Saran and Patil, 2021) described a model that accurately classifies fake news by combining text extraction and supervised machine learning. The method includes important Natural

Language Preprocessing steps like tokenization and stop word removal to clarify the text and improve the feature extraction. Machine Learning models like Naive Bayes, Logistic Regression, and Random Forest are trained on these features, recognizing patterns associated with fake news with great accuracy.

Beyond linguistic patterns, (Pennycook and Rand, 2021) focused on the psychological factors influencing fake news detection and machine learning. They suggest combining cognitive insights with machine learning helps capture the style and semantics that are often present in fake news. Methods like sentiment analysis and language modelling help machine learning models categorize content by analysing both style and factual elements. This highlights a growing trend of incorporating behavioural and linguistic cues into machine learning frameworks to improve misinformation detection.

## 2.5. Deep Learning and AI-Driven Fake News/Rumour Detection

### 2.5.1. Deep Learning Methods for Classification.

Deep learning became a popular tool for classification tasks because it can recognize complex text patterns, identify subtle relationships, and manage large material quite well (Lee *et al.*, 2019). Unlike traditional machine learning, deep learning models can extract hierarchical features from text, improving the detection of misinformation. Additionally, their ability to integrate multimodal data sources, such as images and metadata, has further enhanced their effectiveness in fake news classification.

Models like Convolutional Neural Networks(CNNs), Recurrent Neural Networks (RNNs), Bi-directional LSTM (Bi-LSTM) and attention mechanisms are widely referenced for their effectiveness in such tasks. (Thota *et al.*, 2018) applied CNN and RNN architecture with LSTM layers to detect stance alignment between headlines and article bodies, achieving high accuracy by capturing semantic relationships. Building on this approach, (Kumar *et al.*, 2020) further combined CNNs with Bi-LSTM and an attention mechanism to focus on contextual details and significantly improving classification accuracy.

Recognizing the limitations of text models only, researchers explored multimodal approaches that incorporate both text and visual elements.(Wang *et al.*, 2018) introduced an EANN: Event adversarial Neural Networks for Multi-Modal Fake news Detection, which uses CNNs for text and images with adversarial learning. Its innovative framework was built to be adapted to any new circumstances. (Wu *et al.*, 2021) implemented a Multimodal Co-Attention Network (MCAN). This network uses co-attention mechanisms to learn relationships between text and images, allowing the model to identify subtle interactions between images and text.

As deep learning approaches evolved, transformer-based architecture emerged. (Alarfaj and Khan, 2023) explored various deep learning methods for fake news detection, demonstrating that transformer-based models like RoBERTa/BERT outperform the traditional machine learning algorithms. It highlights the effectiveness of using ensemble methods to enhance the model performance. Their findings also highlight the effectiveness of ensemble learning methods, which further enhance model performance by combining multiple architectures. Expanding on the advancements in transformers and multimodal learning (Truică, Apostol and Karras, 2024) combined Bidirectional Encoder Representations from Transformers (BERT) Bidirectional Long Short-Term Memory (BiLSTM) and Convolutional Neural Networks (CNN) to classify articles based on their claim. Their model incorporates metadata and sentiment analysis, demonstrating that combining multiple feature types significantly improves classification accuracy compared to standalone NLP models.

### 2.5.2. Propagation Methods in Data Analytics: Graph Neural Networks (GNNs) for Classification

(Kipf and Welling, 2017) introduced Graph Convolutional Matrix Completion (GC-MC), a graph-based approach for recommendation systems. Their method reframes the recommendation task as a link prediction problem by representing users and items as nodes in a bipartite graph. The model uses Graph Convolutional Networks (GCNs) to learn embeddings, allowing it to predict missing interactions between users and items based on their connections.

Although originally applied in recommendation systems, their work demonstrated how GCNs can model relationships in networked data, which later influenced fake news and rumour detection techniques by enabling the analysis of propagation patterns and user interactions in misinformation spread.

Expanding GCN-based models, (Veličkovi´veličkovi´c *et al.*, 2018) introduced the Graph Attention Network (GAT) as an enhancement to the Graph Convolutional Network (GCN) by incorporating a self-attention mechanism for adaptive feature aggregation. This approach allows nodes to assign different importance levels to their neighbours, improving information propagation. Their model outperformed standard GCNs in various benchmark tasks, including those tested by (Kipf and Welling, 2017). Initially applied in general-based learning tasks, GAT's ability to identify weighted relationships across the nodes has influenced fake news classification by allowing models to prioritize key users and interactions in networks.

Built on attention-based models, (Yuan *et al.*, 2019) introduced GLAN (Global-Local Attention Network), a graph-based approach for rumour detection that models both local retweet interactions and global social network structures. The model first analyses how retweets interact with the source tweet, applying multi-head self-attention to capture dependencies within a discussion thread. Beyond individual threads, GLAN builds a heterogeneous graph linking users, tweets, and retweets, incorporating social connections and propagation patterns. A Graph Attention Network (GAT) is then used to refine node embeddings, allowing the model to learn from both direct and indirect interactions.

For textual analysis, GLAN employs a Convolutional Neural Network (CNN) to extract semantic and stylistic features from tweets. The model then combines the local tweet-retweet attention features with the global graph-based representations, feeding them into a classifier to distinguish rumours from non-rumours.

While GLAN integrates textual and propagation-based modelling, Shanika et al. (2020) demonstrated that GNNs can classify fake news only relying on by purely how it propagates through the network. This study highlights that such models can perform just like models that carry textual features. This finding suggests that exploring propagation can enhance robustness by tackling the issue of fake news disseminators that modify content to evade detection.

Recognizing the limitations of a single modality, (Han, Karunasekera and Leckie, 2020) introduced an enhanced Graph Attention Network (GAT) designed to process both textual and visual content in social media posts. Their model integrates multi-modal features, allowing it to analyse misinformation more effectively by considering both written content and accompanying images. By applying graph attention mechanisms, the model learns relationships between different data types, improving feature fusion and classification accuracy. Their findings emphasize the importance of multi-modal approaches in fake news detection, as relying solely on text-based methods may overlook key contextual clues in visual media. The paper "Fake News

Synthesizing advancements across graph-based fake news detection**,** (Phan, Nguyen and Hwang, 2023) conducted a comprehensive review of Graph Neural Network (GNN) approaches for fake news detection, highlighting how these models analyse propagation patterns, user interactions, and textual features. They categorized GNN-based methods into Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), and hybrid GNN-NLP models, explaining how each technique contributes to misinformation detection. The study emphasizes the effectiveness of propagation-based detection, demonstrating that fake news spreads differently from real news, which GNNs can learn. Additionally, they discuss the advantages of hybrid models that combine graph-based learning with text analysis using transformers. Their findings suggest that integrating multiple data sources (text, user interactions, and network structures) leads to more accurate and robust fake news detection models.

### 2.5.3. Transformer-based Methods for Misinformation Classification

Recent research has shown that Transformer-based models significantly outperform traditional machine-learning methods in misinformation detection. (Slovikovskaya, 2019) demonstrated that fine-tuned BERT and RoBERTa models achieve high accuracy by first identifying the stance of a news article before determining its authenticity. Expanding on this, (Wu *et al.*, 2021) enhanced Transformer-based fake news detection by integrating CNN and BiLSTM layers, improving contextual feature extraction. Further validating these advancements, (Pardamean and Pardede, 2021) reported that fine-tuned BERT models reached 99% accuracy on text-based misinformation datasets, significantly outperforming conventional approaches. However, despite their effectiveness, Transformer-based models require high computational resources and lack built-in mechanisms for tracking misinformation propagation, which has led to the exploration of hybrid and multimodal approaches.

One such refinement in Transformer-based fake news detection is seen (Li et al., 2021), who applied domain-specific adaptation and ensemble learning to detect COVID-19 misinformation. Their model combined five pre-trained text transformers through five-fold cross-validation, improving model robustness. To further enhance performance, they integrated pseudo-labelling, which expands the training dataset by labelling high-confidence test samples. Additionally, they incorporated domain-specific pretraining using Covid-Twitter-BERT to refine predictions on COVID-19-related content. To stabilize model training, they employed label smoothing to reduce overfitting and learning rate warm-up with cosine decay to optimize convergence. Their results showed that leveraging domain adaptation techniques significantly improves Transformer-based misinformation detection in specialized contexts.

While the previous work focused on text-based information, (Yang *et al.*, 2023) extended it to the Multi-Modal domain, focusing on textual and visual content. It uses transformers to extract visual and textual content. The text part is processed using GloVe embeddings and after a Transformer encoder. The imagery had its features extracted by a Vision Transformer (ViT) instead of the conventional Convolutional Neural Network models.

The text process is divided into two phases:

1- Using GloVe, it tokenises the text into words and sub-words and then maps into a pre-trained GloVe vector, capturing the semantics.
2- The embeddings are then passed through a Transformer Encoder (Instead of using RNNs/CNNS) to capture contextual dependencies. This Transformer Encoder works as a result of two steps:


*Figure 2: Transformer Layer (Yang* et al.*, 2023)*

- **A Multi-Head Self-Attention mechanism** that refines each word's representation by considering the relationship with the words in ahead.
- **Feedforward Layer** applies non-linearity to improve feature learning.

By incorporating Vision Transformers (ViT) for images and transformer encoders for text, their model demonstrated that multimodal fusion enhances fake news classification, capturing both linguistic context and visual cues more effectively than conventional models.

Following this, (Al-alshaqi, Rawat and Liu, 2024) proposed a multi-modal Fake news Detection framework combining text, images and videos using a combination of BERT for text and CNN-based for image classification, leveraging both modalities to improve detection performance.

The framework begins by using a pre-trained BERT model to analyse text and extract contextual features, while a ResNet CNN processes the visual content. To improve the model's ability to detect fake news, the authors integrated feature fusion techniques, allowing textual and visual data to be combined for a more thorough assessment. Their findings showed that using multiple data sources together led to greater accuracy in identifying misinformation compared to models that rely solely on text or images, highlighting the benefits of a multimodal approach.

Beyond text and multimodal learning, graph-based approaches have also been explored to analyze misinformation propagation. (Hoang, Pham and Ta, 2024) introduced a Graph Transformer Network (GTN) that combines Graph Convolutional Networks (GCNs) with Transformer layers to improve social-based recommendation systems. Their model represents recommendation tasks as a graph, where: In their model, the recommendation system is represented as a graph, where:

**Nodes (V)** correspond to users and items (such as products or movies).

**Edges (E)** indicate interactions, including:

- User-item interactions (such as purchases, ratings, or clicks).
- User-user relationships (such as following or trust connections).

**GCN for Learning User-Item Relationships:**

GCN distributes and refines feature representations across the graph by updating each node's information based on the behaviour of its connected nodes, such as users with similar preferences or previously rated items. By aggregating information from neighbouring nodes, GCN enables the system to make more informed predictions about user preferences.

**Transformer Layer for Feature Refinement:**

Once the GCN has generated initial embeddings for users and items, a Transformer layer is applied to enhance these representations.

- Multi-Head Self-Attention is used to capture complex relationships between users and items.

- This allows the model to identify influential connections between nodes, even if they are not directly linked within the graph.

By integrating GCN for structural learning and Transformers for attention-based refinement, the proposed model effectively improves the accuracy of social-based recommendations, outperforming traditional graph-based approaches.

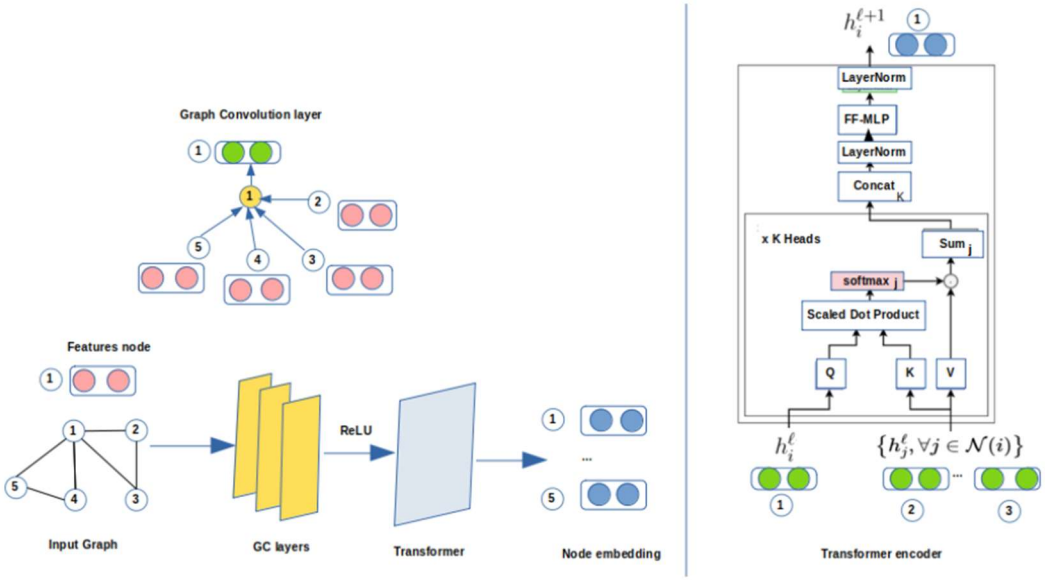



*Figure 3: GCN and Transformer Integration* ***(Hoang, Pham and Ta, 2024)***

## 2.6. The Need for Hybrid and Graph-Based Approach in Misinformation Classification

Fake news detection has evolved from simple text classification to multimodal and deep learning methods, driven by the outgrowing complexity of misinformation. Earlier methods focused on linguistic patterns (Parikh and Atrey, 2018), while recent approaches incorporate user interactions, metadata, and propagation features ((Yang *et al.*, 2019); (Han, Karunasekera and Leckie, 2020)). The limitations of using a singular approach have motivated researchers to build hybrid models that integrate multiple techniques for improved accuracy. This section summarizes the advantages and limitations of existing methodologies, leading to the justification for a BGCN-Transformer model.

### 2.6.1. The Role of Multimodal Analysis

Misinformation is rarely just about the text; it often includes misleading visuals or manipulated multimedia content to make it more convincing (Segura-Bedmar and Alonso-Bartolome, 2022). Relying on just one type of data, such as text-based classifiers, can lead to false negatives where misinformation goes undetected simply because the model does not consider images or contextual elements.

- (Wu *et al.*, 2021) developed the Multimodal Co-Attention Network (MCAN), which looks at how text and images interact to improve detection.
- (Dhawan *et al.*, 2022) proposed a Graph Attention Network (GAT) that combines text and visual data, making it better at identifying fake news.
- (Yang *et al.*, 2023) introduced a Multi-Modal Fake News Detection model using GloVe embeddings for text and Vision Transformers (ViT) for images, showing how effective combining different types of data can be.

- (Truică, Apostol and Karras, 2024) combined Bidirectional Encoder Representations from Transformers (BERT) and Bidirectional Long Short-Term Memory (BiLSTM) and Convolutional Neural Networks (CNNs).

*Table 1: Pros and Cons Multimodal*

| Pros | Cons |
|---|---|
| Improves accuracy when adding multiple data sources such as text, images, video, and metadata | Higher Computational Costs are needed to process all the data |
| Detects the inconsistencies between the data sources | Limited datasets are available |
| | Not all misinformation is multimodal |

The multimodal approach by itself improves the accuracy of the models compared to standalone models. However, nowadays, with social media and its easy propagation, it's necessary to develop methods that can capture how fake/rumours spread over the internet.

### 2.6.2. The Importance of Graph-Based Approaches for Misinformation Classification.

Fake news is not just about content; how it spreads is equally important. Social networks create bubbles where misinformation starts to gain credibility by being distributed repeatedly (Han, Karunasekera and Leckie, 2020) These propagation patterns led to the rise of graph-based models that can perfectly analyze this relationship between users and fake news being propagated. It showed that fake news follows distinct propagation patterns, making it detectable even without analyzing textual content.

(Yang *et al.*, 2019) introduced GLAN (Global-Local Attention Network), which integrates retweet relationships and social network structures to improve misinformation detection.

(Phan, Nguyen and Hwang, 2023) conducted a comprehensive review of GNN-based fake news detection models, highlighting how Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs) can enhance detection performance.

*Table 2: Pros and Cons Graph-Based*

| Pros | Cons |
|---|---|
| Capture users' interaction and how it spreads differently from real information | Text content is often not taken into consideration |
| GNNs can identify accounts and groups that are maliciously spreading the news as a campaign | Limited by sparsity because not all fake news/rumour spreads can be predictable |
| Improves the detection of deceptive tactics, as misinformation gains credibility through social approval rather than being false (Social Validation). | Highly computationally intensive, requires substantial data processing |

While graph-based methods excel at tracking how misinformation spreads, they do not consider the linguistic and contextual aspects of fake news content. Since misinformation involves both propagation patterns and textual manipulation, it is essential to incorporate both aspects to allow the model to analyze semantic meaning, contextual dependencies, and dissemination behaviour, leading to more comprehensive detection.

### 2.6.3. The Importance of Transformer-Based Approaches

Transformer-based models revolutionized classification tasks by capturing semantic meaning, structures, and contextual dependencies in the text. It differs from traditional machine learning

methods that rely only on linguistic features because Transformers leverage self-attention mechanisms to process long-range dependencies, making them suitable for identifying manipulation in misinformation (Slovikovskaya, 2019).

(Wang *et al.*, 2021) Developed a BERT-based model with CNN and BiLSTM layers, showing that context-aware text processing significantly improves fake news detection.

(Pardamean and Pardede, 2021) Demonstrated that fine-tuned BERT models outperform traditional machine learning approaches, achieving nearly 99% accuracy on text-based misinformation datasets.

(Li *et al.*, 2021) Applied domain-specific pretraining and pseudo-labeling, proving that Transformers generalize well across different misinformation domains.

(Yang *et al.*, 2023) Proposed a multi-modal fake news detection model, combining GloVe embeddings for text processing with a Transformer encoder to capture contextual dependencies.

*Table 3: Pros and Cons Transformer-based*

| **Pros** | **Cons** |
|---|---|
| Captures semantic and syntactic relationships within the text, improving accuracy | Requires high computational resources |
| Pre-trained Transformers are easy to handle and adapt | Understands context deeply but does not track how it spreads over the internet |
| Better performance in analysing text compared to traditional methods | Social Validation can be an issue for a Transformer-based model |

While pre-trained Transformers like BERT are powerful for analyzing textual features, they do not model user interactions, misinformation propagation, or network structures.

BERT embeddings are used for feature extraction, ensuring that linguistic nuances are captured.

Multi-head self-attention is applied after GCN to refine node representations, allowing the model to learn meaningful connections in social networks.

Feedforward layers enhance feature interactions, improving classification performance.

### 2.6.4. Hybrid Models: Combining Graph Network and Transformers

While Graph Neural Networks (GNNs) capture the network by themselves, Transformer-based models can capture the linguistic features precisely. They have shown very good results in the past, but alone they are facing a new challenge nowadays. The rise of multimodal information, including images, videos, and metadata might complicate the detection. To address this challenge, researchers have combined multiple techniques into a hybrid model that incorporates multimodal data, graph-based exploration, and attention-based mechanisms.

(Dhawan *et al.*, 2022) Proposed a Graph Attention Network (GAT) combining textual and visual data, showing that multimodal learning improves feature extraction across different formats.

(Yang *et al.*, 2023) Developed a Multimodal Fake News Detection model, leveraging GloVe embeddings for text and Vision Transformers (ViT) for images, showing that combining text and images enhances misinformation detection.

(Phan, Nguyen and Hwang, 2023) Conducted a review highlighting that hybrid GNN-Transformer models outperform standalone models by integrating graph propagation patterns and textual meaning.

(Alarfaj and Khan, 2023) Demonstrated that Transformer models like BERT and RoBERTa outperform traditional methods, but their effectiveness increases when integrated with social graph structures to model misinformation spread.

(Hoang, Pham and Ta, 2024) Introduced Graph Transformer Networks (GTN), showing that GNNs for network structure and Transformers for attention-based refinement lead to better social-based predictions.

*Table 4: Pros and Cons GCN-Transformer*

| **Pros** | **Cons** |
|---|---|
| Combines all techniques leading to a robust classifier | Requires as much as all technique's combined computational resources together |
| By combining propagation and content learning, it can adapt the detections to different tactics | Contains a very complex structure as a result of combining all the techniques |
| | Relies on complete datasets, containing context, propagation and metadata, making it hard to obtain due to legal and commercial restraints |

#### 2.6.5. Conclusion

When combining multimodal analysis, social network learning, and transformer-based attention mechanisms, our BGCN-Transformer model provides a more complete approach to detecting fake news and rumours. Traditional methods, such as machine learning, focus mainly on text-based classification, while graph-based models emphasise network propagation. In contrast, this hybrid solution integrates both aspects, enhancing accuracy and adaptability. By leveraging these combined techniques, the model not only improves detection precision but also addresses key challenges, such as misinformation evolution, deceptive tactics, and scalability issues. This approach contributes to the development of more reliable, real-time misinformation detection systems that can adapt to the ever-changing nature of online content.

# 3. Data Collection and Preprocessing

## 3.1 Data Description

### 3.1.1. Overview

This thesis was built upon the Twitter 15 and Twitter 16 datasets (Yang *et al.*, 2019) which are benchmarked for rumour and fake news detection. They contain real-world misinformation cases and their respective propagation trees.

**Source:** Twitter platform (X)

**Purpose**: Fake News and rumour detection

**Language**: English

**Period of collection**:

- Twitter 15: Data collected from 2015
- Twitter 16: Data collected from 2016

### 3.1.2. Data Structure

Each of the Datasets contains 2 different datasets:

**The Parent Dataset** :

- **ID:** Represents the unique ID of the original tweet, where the discussion started.
- **Content:** Represents the text of the tweet itself.
- **Status:** Represents the labels related to that tweet, each label means:

  **True:** Verified as real news.

  **False:** Misinformation of fake news.

  **Unverified:** Still not yet identified.

  **Non-**Rumour: Tweets that do not involve rumours.

After concatenating both datasets, it contains 2308 (1490 from Twitter 15 and 818 from Twitter 16) rows and 3 columns.

*Table 5: Parent dataset*

| ID | Content | Status |
|---|---|---|
| 0731166399389962242 | %ca kkk grand wizard =% endorses @hillaryclin... | unverified |
| 1714598641827246081 | an open letter to trump voters from his top st... | unverified |
| 2691809004356501505 | america is a nation of second chances @potus ... | non-rumour |
| 3693204708933160960 | brandon marshall visits and offers advice, sup... | non-rumour |
| 4551099691702956032 | rip elly may clampett: so sad to learn #beverl... | true |

**The Child Dataset**: It represents the propagation and also metadata.

- **File:** The filename associated with the tweet conversation thread.
- **Parent_uid**: The user ID of the original tweet author.
- **Parent_tid**: The tweet ID of the parent tweet (i.e., the original tweet being replied to or retweeted).
- **Parent_delay**: The time delay (in seconds) from the source tweet creation.
- **Child_uid**: The user ID of the replying or retweeting user.
- **Child_tid**: The tweet ID of the child tweet (i.e., a reply or retweet of the parent).
- **Child_delay**: The time delay (in seconds) from the source tweet creation until the reply/retweet.

After concatenating, it contains 957,968 rows (605,799 from Twitter 15 and 352,169 From Twitter 16) and 7 columns

*Table 6: Child Dataset*

| | |
|---|---|
| **File** | 532602376398462976.txt |
| **Parent UID** | 286742737 |
| **Parent TID** | 532602376398462976 |
| **Parent Delay** | 0.00 sec (source tweet) |
| **Child UID** | 2321563844 |
| **Child TID** | 532602516014264320 |
| **Child Delay** | 0.55 sec |

User 286742737 posted a tweet (ID: 532602376398462976) at time 0.00 sec.

User 2321563844 replied to (or retweeted) this tweet after 0.55 seconds.

The reply tweet had the ID: 532602516014264320.

### 3.1.3. Challenges in Data Collection for Fake News and Rumour Classification

Despite the increasing focus on misinformation detection, several challenges persist when it comes to data collection and analysis from social media platforms such as X (old Twitter).

- **Limitations of Twitter API for data retrieval:**

Recent policies were changed when Twitter was sold recently and have significantly restricted the access for research. It not only killed the Student/Researcher Account but also reduced drastically the free version while increasing the price of the paid APIs.

Previously, researchers could collect detailed metadata such as user verification, and engagement metrics (Tweets and Retweets). And network-based features (followers and relationships). However, due to this new API restriction, retrieving such metadata became almost impossible without a paid account. As a result of this, this study will rely on the available dataset without accessing additional user-level features.

- **Incomplete or Noisy Data:**

Fake News datasets rely on high-quality, labelled datasets, but misinformation datasets usually suffer from:

**Labelling:** Distinguishing fake and real news is usually assigned based on fact-checking websites, which may not be fully objective and can take time to address that content.

**Unstructured and messy data**: Datasets prevenient from social media always contain slang, emojis abbreviations and poorly written texts, making it hard for Natural Language models.

**Expensive and Time-Consuming Collecting Process**: Collecting and labelling large datasets requires an extensive amount of work and that can turn out to be very expensive, therefore, high-end fake news datasets are quite difficult to obtain.

**Multimodal complexity:** Misinformation isn't always text-based, it can be manipulated images, or videos. Datasets with such types of data can be extremely heavy to process and work with.

### 3.1.4. Ethical and Legal Considerations:

Developing fake news detection models raises ethical and legal concerns:

**Censorship x Free Speech**: Systems that are designed to detect fake news can sometimes classify controversial but real content as false, leading to concerns about censorship and bias. This challenge extends beyond content moderation to data labelling itself, as fact-checking sources may have their own biases or subjective criteria when determining what is true or false. Since misinformation is often context-dependent, news labelled as "fake" today might later be verified as true, making the labelling process inconsistent and sometimes unreliable. Ensuring fairness while combating misinformation remains a difficult but necessary balance.

**Privacy Risks**: Many models for fake news classification rely on analyzing sensitive data from users such as followers, likes, and engagement patterns, to identify the misinformation spreading. However, collecting this type of feature can raise serious data privacy concerns, as it could be misused, leading to a witch-hunt or exposing such users.

In Europe, data protection laws such as the General Data Protection Regulation (GDPR) set strict rules and guidelines on how the data that belongs to the user can be collected, stored and used.

#### 3.1.5. Conclusion

Given the challenges in data access, labelling, and privacy concerns, this study relies on the Twitter 15 and 16 datasets in their open-source format, avoiding the need for additional metadata from Twitter (X).

The decision to use these datasets is based on:

- API Restrictions – Recent policy changes limit access to essential metadata, making direct data collection impractical.
- Labelling Challenges – Fact-checking misinformation is time-consuming and subjective, making pre-labelled datasets more reliable.
- Privacy & Ethical Concerns – Collecting user engagement data risks violating privacy laws like GDPR, so this study avoids using personal information.

To adapt the dataset for fake news detection, we:

- Extracted text features using BERT embeddings to analyze misinformation patterns.
- Processed metadata while ensuring privacy compliance.
- Built a graph structure to model how fake news spreads, enabling Graph Convolutional Networks (GCNs).

By transforming the dataset this way, we overcome data access restrictions while ensuring effective misinformation detection.

### 3.2. Data Preprocessing

#### 3.2.1. Data Cleaning

The dataset was first examined for missing values in both the Parent and Child datasets, and no null values were found. Since the dataset is complete, no imputation or data removal was required at this stage.

Cleaning and normalizing texts have a big influence on the model's accuracy, according to (Esuli and Sebastiani, 2009), they are critical, and the key reasons are:

**Removing Noisy and Inconsistent Data:** Text collected from social media often contains special characters, poor grammar, and abbreviations.

**Standardizing Input for a better model:** Models like BERT rely on textual patterns, for example: uppercase and lowercase or misspellings can make the model identify the same word as two distinct words, therefore penalizing the model's accuracy.

**Handling Stopwords and Punctuation:** Removing stop words (e.g., “the”, “a”, “an”) ensures only words that can be meaningful go through the classification process.

By taking those actions, the proposed word embeddings and feature extractions might achieve a better performance. So, the actions taken to clean the text were as follows :

*Table 7: Cleaning text steps*

| Step | Action |
|---|---|
| 1 | Remove URLs |
| 2 | Remove HTML tags |
| 3 | Remove special characters, punctuation, and numbers |
| 4 | Reconstruct split characters into proper words |
| 5 | Normalize whitespace |
| 6 | Convert to lowercase |
| 7 | Remove the word "URL" explicitly |

Output:

*Table 8: Output cleaned text*

| Original Text | Cleaned Text |
|---|---|
| 🔥 ca kkk grand wizard 🔥 endorses @hillaryclinto... | ca kkk grand wizard endorses hillaryclinton ne... |
| An open letter to Trump voters from his top st... | An open letter to Trump voters from his top st... |
| America is a nation of second chances — @potus ... | America is a nation of second chances potus on... |

### 3.2.2. BERT Embeddings – Initial Embeddings Encoder

According to (Devlin *et al.*, 2018) in *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding BERT* or Bidirectional Encoder Representations from Transformers, is a pre-trained language model designed to understand the context between words by processing text bi-directionally. Unlike traditional word embeddings such as Word2Vec or GloVe, which assign a fixed meaning to words regardless of context, BERT dynamically adjusts word representations based on surrounding text.

For instance, consider the word "bank" in the following sentences:

*He sat by the bank of the river.*

*She deposited money in the bank.*

While traditional embeddings would represent "bank" as a single vector, BERT understands the different contexts and assigns different embeddings to "bank" in each case, improving language comprehension.

BERT is trained in two main phases: pre-training and fine-tuning. In this study, we leverage only the pre-training phase to convert raw Twitter text into numerical representations (embeddings) that preserve semantic and syntactic relationships while being machine learning.

- **Masked Language Modeling (MLM):** It masks randomly words in a sentence, and BERT predicts the missing words based on the context involving it.

**Example:** The bird [MASK] through the sky – BERT is trained then to predict the masked word as the sky, using the surrounding context. This process ensures that the previous and next words are taken into consideration.

- **Next Sentence Prediction (NSP):** Bert is trained to determine whether one sentence naturally follows another, improving the understanding between the sentences.
  **Example:**
  - Sentence A: "The man went to the store."
  - Sentence B: "He bought some milk." (Is a valid continuation of A)
  - Sentence C: "The sun is very bright today." (Not a valid continuation of A)

By understanding these connections, BERT improves its ability to recognize coherence and the logical progression of sentences.

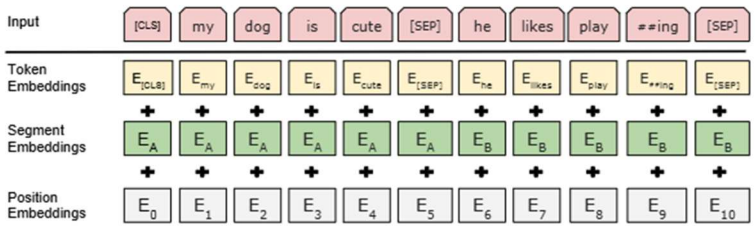



*Figure 4: BERT Tokenizer (Devlin* et al*., 2018)*

- Token Embeddings: These represent the meaning of each word or word in the text.
- Segment Embeddings: Used to differentiate between two parts of the input (like sentence pairs).
- Position Embeddings: These add information about the position of each word in the sequence, helping BERT understand word order.

In this study, we use only the pre-trained word task from BERT. Instead, we apply BERT's tokenizer to process tweet text, converting it into numerical values that represent linguistic patterns and meanings.

### 3.2.3. Metadata processing

To achieve better results in the proposed models, exploring and extracting all the features from our dataset is essential. The following features were retrieved by each Tweet ID:

- Total Number of Interacting Users: The total count of unique user IDs (child_uid) that interacted with the original tweet provides insights into the reach and engagement of the post.
- Average Interaction Time: The average time delay between the original post and subsequent interactions helps understand user engagement behaviour.
- Minimum and Maximum Interaction Time: These values highlight the fastest and longest delays in user interactions, indicating the spread dynamics of the post.

- List of Interacting User IDs: A comprehensive list of user IDs that engaged with a specific tweet allows for graph construction and propagation analysis.
- First User Interaction: Capturing the first interaction on a tweet identifies early adopters, which is critical for analysing how fake news spreads initially.

Since the metadata features have different numerical ranges, feature scaling was necessary before using them in the model. StandardScaler was chosen because it standardizes data to have zero mean and unit variance, ensuring consistency across features.

We used StandardScaler instead of MinMaxScaler because it:

- Handles diverse numerical distributions without distorting relationships.
- Preserves interpretability by keeping the original feature distribution.
- Improves model performance, enhancing convergence in GCNs and Transformers.

This ensures that all metadata features are scaled consistently, allowing the BGCN-Transformer model to learn effectively from propagation patterns and sentiment features.

To further explore the tweet text representations, RoBERTa was applied as part of the model. Building on its robust pre-trained architecture, Roberta provides very accurate sentiment analysis.

According to "RoBERTa: A Robustly Optimized BERT Pretraining Approach," RoBERTa is a variant of BERT that improves its robustness for several tasks, including Sentiment Analysis.

*Table 9: RoBERTa x BERT*

| Feature | Description |
|---|---|
| Increased Training Data | RoBERTa is trained on a larger and more diverse dataset compared to the original BERT. |
| Dynamic Masking | RoBERTa dynamically masks tokens for each training epoch, enabling more robust learning. |
| No Next Sentence Prediction (NSP) | RoBERTa eliminates the NSP task used in BERT's pre-training, focusing entirely on the masked language modelling (MLM) task. |
| Larger Batch Sizes and Learning Rates | It uses bigger batch sizes and fine-tuned learning rates for better optimization. |
| Longer Training Duration | RoBERTa is trained for a longer period, allowing it to capture deeper contextual information. |

These combinations of features are essential for constructing a robust graph and extracting patterns related to propagation dynamics, enabling the model to learn from both user behaviour and tweet characteristics, and sentiment behind those texts.

### 3.2.4. Exploratory Data Analysis (EDA) and Feature Insights

**Label Distribution:**

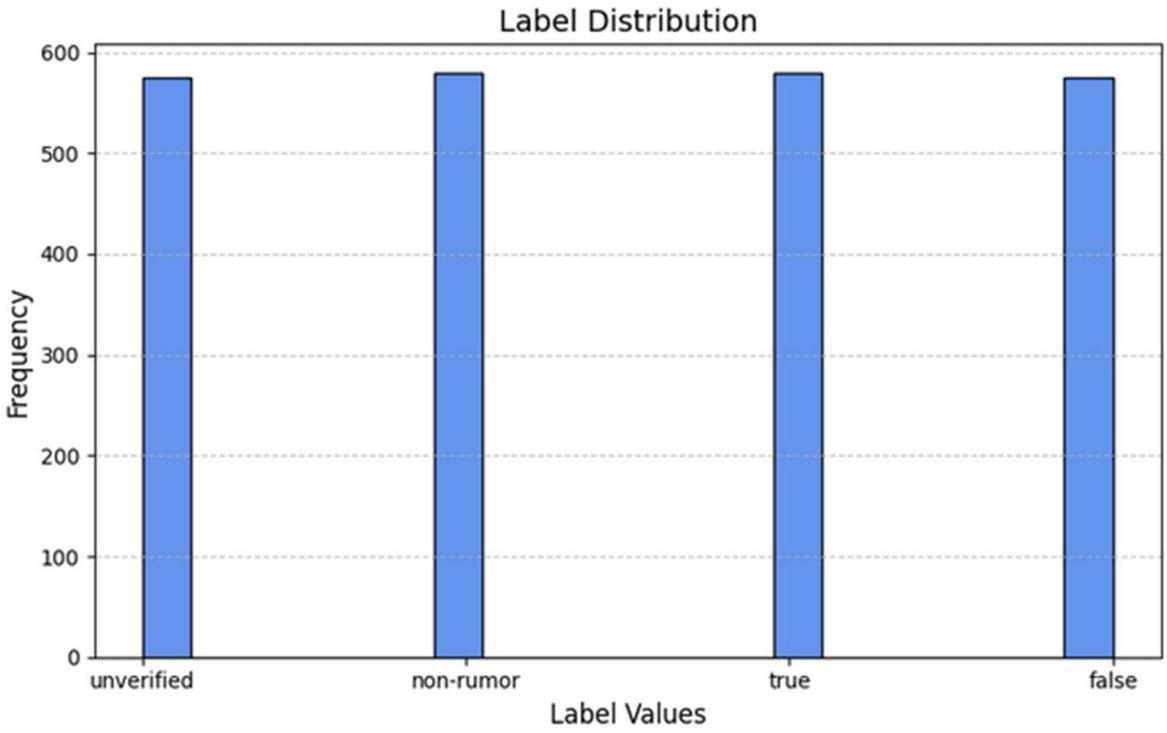



*Figure 5: Label Distribution*

The dataset appears to be balanced, with an equal number of observations across all label categories, ensuring fair representation in classification tasks

**Word Cloud Representation:**

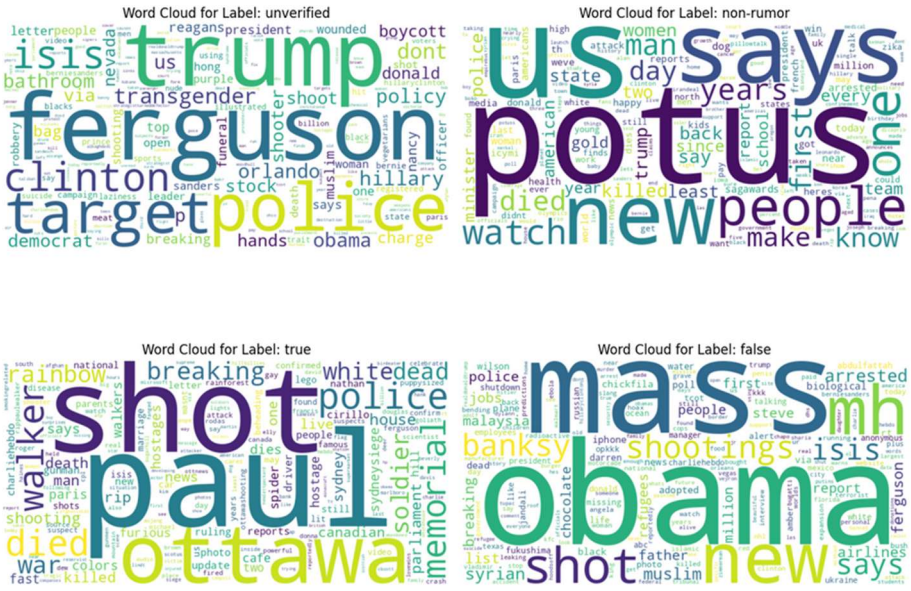



*Figure 6: Word Cloud per label*

The word cloud analysis for different tweet categories reveals distinct themes and patterns. Unverified tweets prominently feature terms like Trump, Ferguson, police, target, and Clinton, indicating discussions centred around politically charged topics and law enforcement incidents. The presence of targets and police suggests frequent discourse on law enforcement-related events. Non-rumour tweets frequently contain words such as POTUS, US, says, new, and people, reflecting a focus on factual reporting, particularly on government affairs. The word says appears often, suggesting that many of these tweets involve quoting or reported speech. True tweets are associated with keywords like shot, Paul, Ottawa, breaking, police, and died, highlighting their focus on crime-related events and significant breaking news. The mention of Paul and Ottawa suggests references to widely covered events during the dataset's period. False tweets, on the other hand, commonly include terms like mass, Obama, shot, new, and shootings, indicating a strong association with mass shootings and misinformation involving public figures. The repeated presence of shootings and mass implies that many false claims revolve around incidents of violence.

**Sentiment Distribution between each label category:**

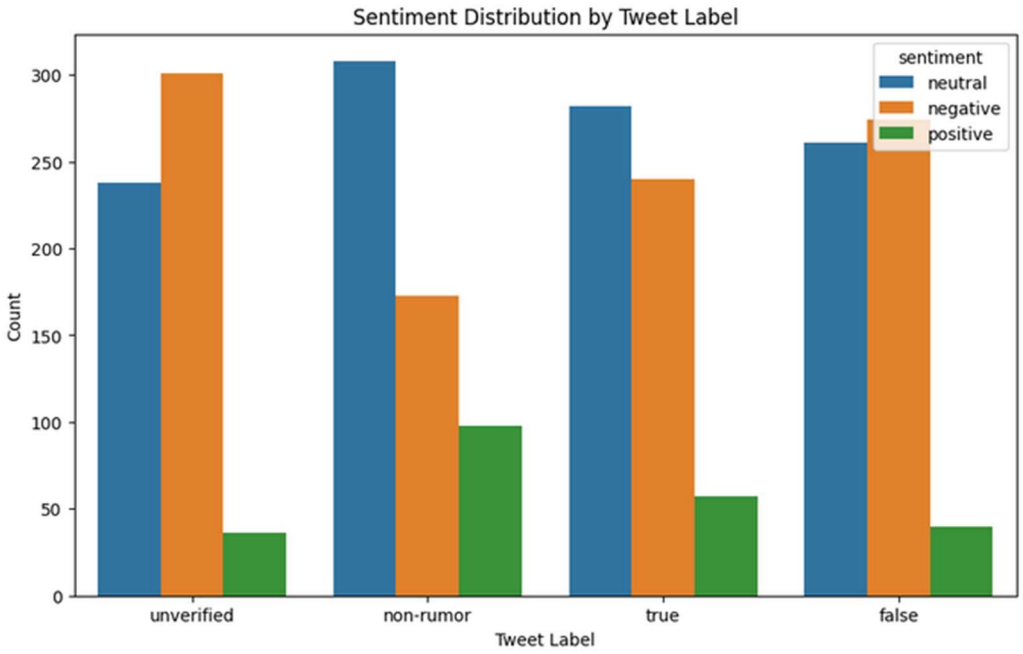



*Figure 7: Sentiment per label*

The chart shows sentiment distribution across different tweet labels. Negative sentiment is highest in unverified and false tweets, suggesting misinformation and uncertainty often carry a negative tone. Neutral sentiment dominates non-rumour and true tweets, indicating more factual reporting. Positive sentiment is rare but slightly higher in non-rumour tweets.

**Total Interaction per Labels:**

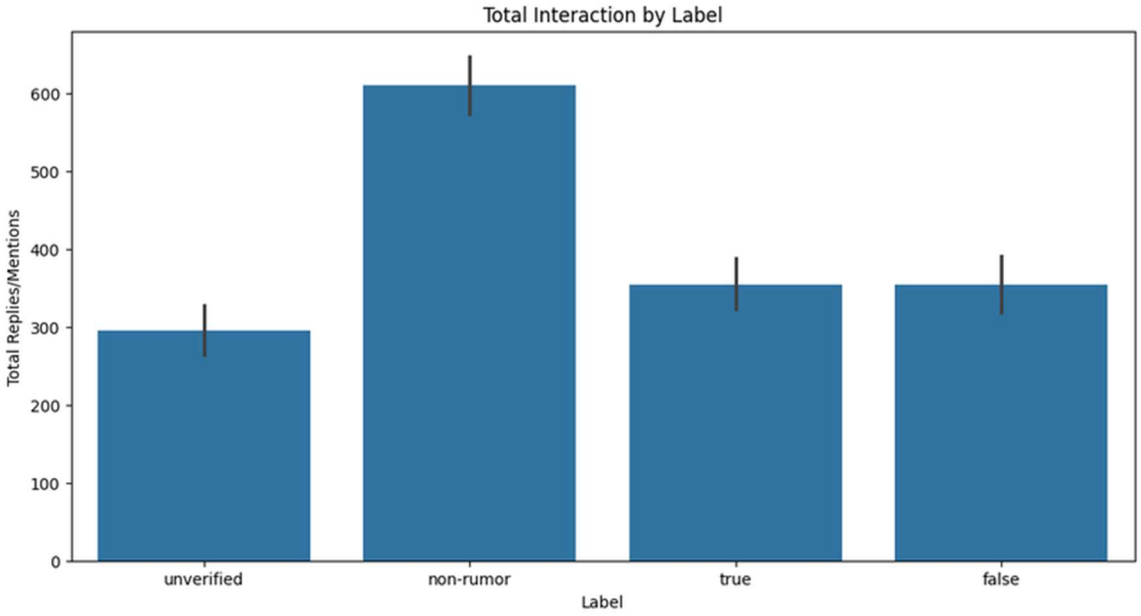



*Figure 8: Total Interaction by Label*

Our dataset shows that the total interaction between users over the labels tends to be similar but non-rumour tweets receive significantly higher engagement compared to the other categories.

**Engagement according to sentiment across labels:**

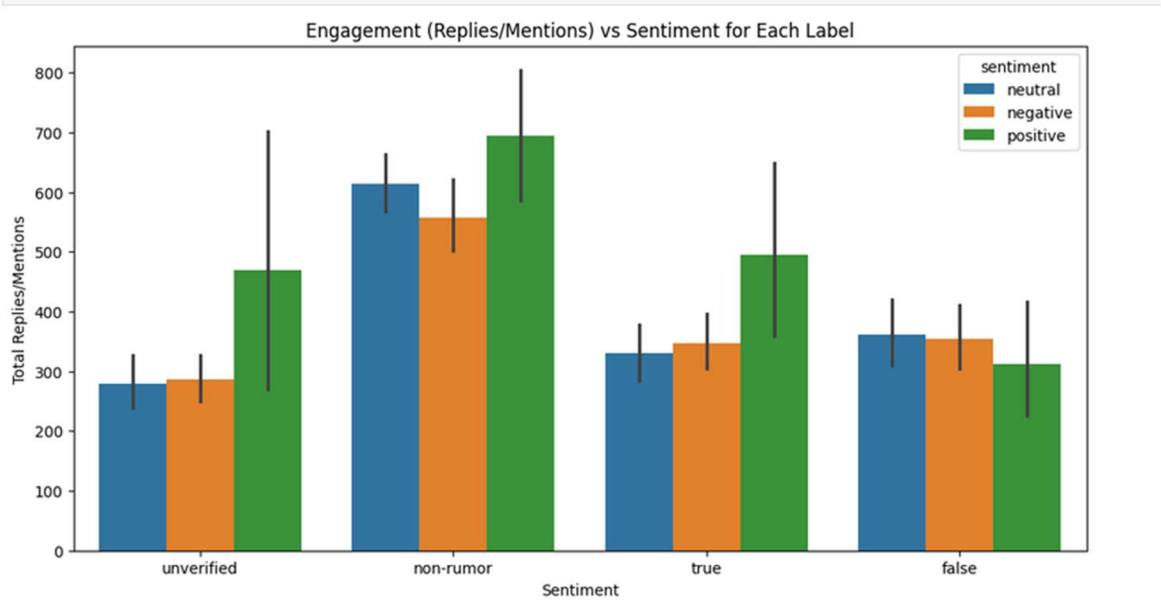



*Figure 9: Engagement vs Sentiment per Label*

Non-rumour tweets receive the highest engagement, especially for positive sentiment. True tweets also show a notable rise in interaction for positive sentiment. Unverified and false tweets have relatively balanced engagement across sentiments but at lower levels overall. The variation in engagement is more pronounced in non-rumour and true tweets.

**Total Interaction by Sentiment and Tweet Label:**

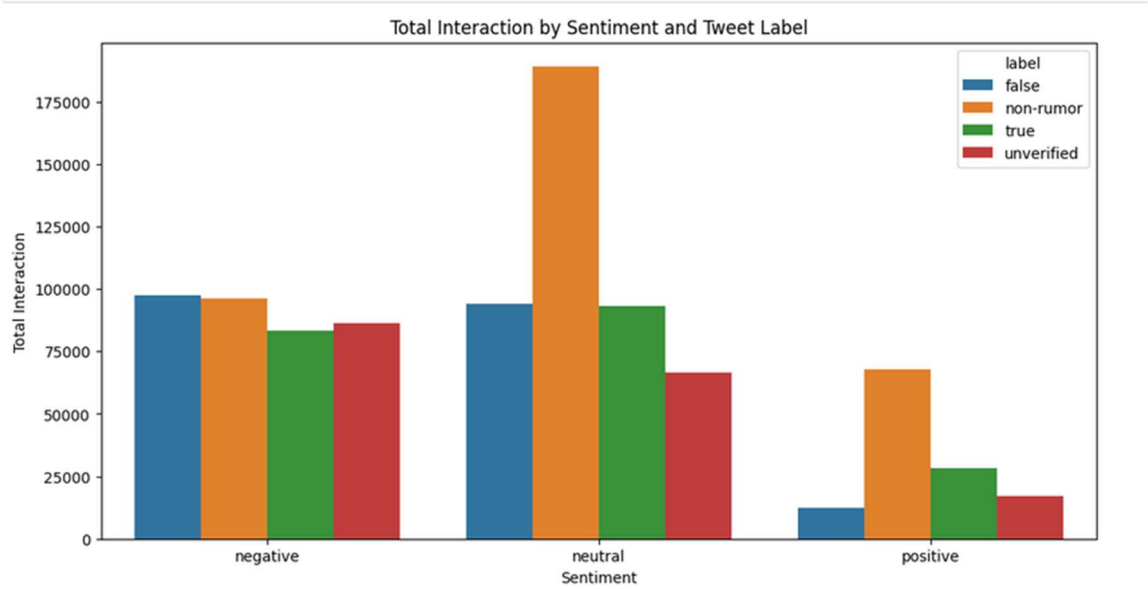



*Figure 10: Total interaction by Sentiment and Labels*

Neutral sentiment drives the highest engagement, particularly in non-rumour tweets, which show a significant spike. Negative sentiment also generates substantial interaction across all labels, with false and non-rumour tweets leading. Positive sentiment sees the least engagement overall, but non-rumour tweets still attract more interaction compared to other labels.

**Average Response Time by Tweet Labels:**

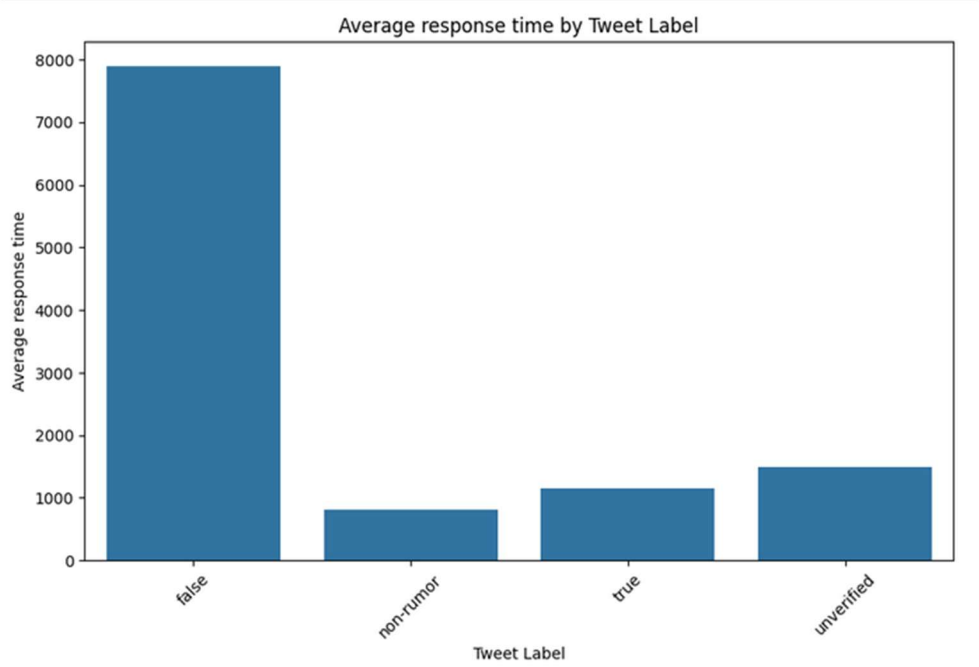



*Figure 11: Average Response Time*

False tweets have a significantly higher average response time compared to other labels, suggesting that misinformation lingers longer before being addressed. In contrast, non-rumour, true, and

unverified tweets have much shorter response times, indicating that verified and uncertain content gets quicker engagement or fact-checking

**Average Response per Label and Sentiment:**

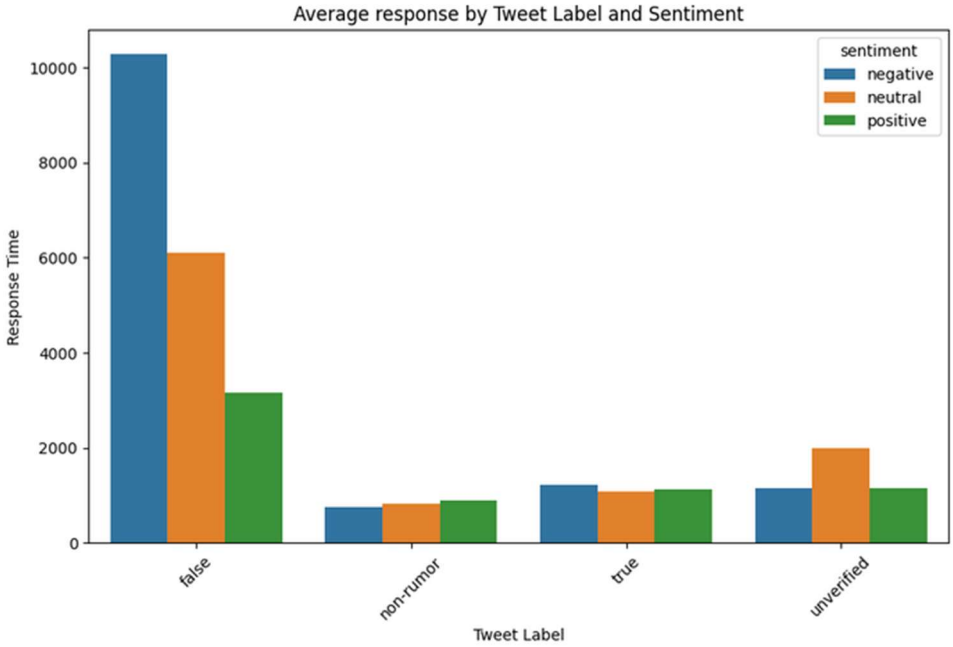



*Figure 12: Average Response Time by Label and Sentiments*

False tweets, especially with negative sentiment, have the longest response time, while non-rumour, true, and unverified tweets receive much quicker engagement. This reinforces the pattern that misinformation lingers longer before being addressed, whereas factual or uncertain content gets faster responses.

**Conclusion:**

While some trends are visible, such as misinformation taking longer to be addressed and non-rumour tweets receiving more engagement, these patterns alone are not enough. The high volume and constantly changing nature of online content make it difficult to track manually. Machine learning can help by processing large amounts of data efficiently, identifying patterns beyond what is immediately visible, and improving the detection of misinformation.

## 3.3. Graph Construction

To build a graph we need to understand its key components(Maekawa *et al.*, 2019), as follows:

**Node** is the context of a Graph Neural Network; it represents an entity that carries relevant attributes and may be associated with a pre-defined class label. Nodes connect through the Edges, defining the interaction between them.

**Node Attributes**:  Node Attributes are the features combined or characteristic of each node in the graph.

**Edges**: Edges are defined as the connection between the nodes in the graph. Representing relationships such as retweets and likes. Any sort of interaction between the nodes.

**Edge Attributes:** it holds additional information such as interaction throughout the time or how strong the relationship

**Edge Index**: This represents a structured way of storing the graph's connections, pointing to pairs of nodes that share an edge. This is fundamental for Graph Neural Network models to work efficiently.

**Edge Weight**: Represents the strength of the connection between nodes. It can be calculated by using diverse factors such as similarity scores, frequency, or domain-specific metrics.

To begin with our Graph Construction, this thesis will follow the following steps:

### 3.3.1. Node Mapping

First identify the possible nodes such as:

**Tweed ID (Posts):** Each unique tweet is defined as a node.

**Parent_uid and child_uid:** Each user interacting with a tweet (either being the owner of the tweet or a user that ) is assigned as a node as well.

**Mapping Process:**

The mapping process begins by creating an index mapping for tweet IDs from the parent node, ensuring each tweet is uniquely identified. Next, all users are mapped to corresponding node indices, allowing for structured representation within the graph. The total number of nodes is then determined by the combined length of both tweet and user mappings, forming the foundation for graph construction.

### 3.3.2. Edge Creation**:**

**Tweet-to-User Edges:** Connect each tweet to the user who posted it.

**Tweet-to-Commenter Edges:** Connect a tweet to users who commented on or interacted with it.

**User-to-User Edges:** Represent relationships between users based on interactions.

### 3.3.3. Feature Extraction:

The metadata generated during the preprocessing stage, including BERT embeddings and additional features, is normalized and stored for further processing. Similarly, delay-based user features are extracted, normalized, and stored to maintain consistency across different data types. Given that tweet nodes with BERT embeddings and additional metadata contain over 700 features, while user nodes have significantly fewer, padding is applied to the user feature matrix. This ensures uniformity in node features, allowing for a balanced graph structure. Padding helps standardize graph sizes, making computations more efficient, especially when utilizing GPUs. However, while this technique optimizes large-scale processing, it can introduce unnecessary calculations, increasing computational complexity (Zaman *et al.*, 2022)

### 3.3.4. Graph Feature Matrix Construction

After preparing the features from both users and tweets, a feature matrix x is initialized to store node attributes for all nodes in the graph. Tweet nodes are assigned BERT embeddings along with additional metadata extracted during preprocessing, ensuring that both linguistic context and tweet-specific information are incorporated. Since user nodes have fewer inherent features, they are padded to maintain consistency in feature dimensions and assigned delay-based interaction features. This structured representation allows the model to effectively capture both textual and interaction-based patterns, enabling better learning of misinformation propagation dynamics.

### 3.3.5. Edge Weights Computation

To capture relationships between connected nodes, Edge weights are computed using Cosine Similarity, which is a metric used to measure the relationship between two nodes by calculating the cosine of the angle between their feature vector. Cosine Similarity can be used to assign weights to the edges of a graph, in a way that nodes with similarities are strongly connected(Smutek *et al.*, 2024)

### 3.3.6. Symmetric Adjacency Matrix Construction

Adjacency Matrix is a representation of the graph structure using a square matrix. Each row and column corresponds to a node and the values inside of it represent the connection between each of the nodes. If there's a connection between node A and node B, the value assigned in the matrix is 1, otherwise 0 (Harary, 1962).

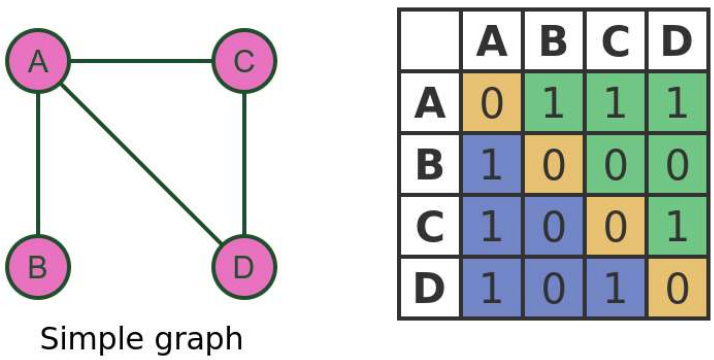



*Figure 13: Adjacency Matrix (McBride, 2023)*

Graph Convolutional Networks (GCNs) rely on an adjacency matrix to represent node connections and gather information from nodes around. A Key optimization of GCN implementations is the usage of symmetric adjacency matrix. This approach provides several advantages such as reducing redundancy, computation acceleration, and improving efficiency (G. R. Nair *et al.*, 2023)

### 3.3.7. Labeling Nodes

The labelling process is a key step in training the BGCN-Transformer model, ensuring that the model correctly identifies and classifies misinformation while also learning from user interactions. Since the dataset contains tweets (labelled) and users (unlabeled), a semi-supervised learning approach was applied.

**Labeling Tweets – Using Label Encoder:**

Tweets were already categorized into four distinct labels: Fake, Real, Unverified, and Non-Rumor. To make these labels compatible with machine learning models, they were converted into numerical values using a Label Encoder. This approach ensures a standardized representation of labels while minimizing potential biases that could arise from arbitrary label assignments, ultimately improving the consistency and reliability of the classification process.

**Assigning -1 to Unlabeled Users – Based on Semi-Supervised Learning:**

User nodes, which do not have explicit labels, were assigned a value of -1 following the approach outlined by (T. N. Kipf and Welling, 2016) This allows the Graph Convolutional Network (GCN) to propagate information from labelled tweets to connected users, enabling the model to learn patterns of misinformation spread. By structuring the dataset in this manner, the model can infer

user credibility and influence in misinformation dissemination without requiring direct user labels, as demonstrated in previous studies ((Shu et al., 2019; Phan, Nguyen and Hwang, 2023) This approach ensures that the model effectively utilizes both labelled content and network interactions to enhance the accuracy of fake news detection.

### 3.3.8. Final Graph Structure

Now that we have assigned labels to tweet nodes (y tensor), we can finalize the graph construction by defining the PyTorch Geometric (PyG) Data object. This step integrates node features, edge indices, edge attributes, and labels into a single format that can be fed into the model.

*Table 10: Graph Structure*

| Component from the Graph | Description |
|---|---|
| Nodes (x) | Tweets (BERT embeddings + metadata) and Users (delays, interactions) |
| Edges (edge index) | User-Tweet, Tweet-User, and User-User relationships |
| Edge Weights (edge attributes) | Cosine similarity + interaction delay |
| Labels (y) | Tweet classification labels, user nodes remain -1 |

**5. Graph**

The final Graph output is:

*Table 11: Final Graph design*

| Variable | Dimensions | Description |
|---|---|---|
| x or Nodes | [679779, 774] | 679,779 nodes and 774 features per node. |
| edge index | [2, 2901484] | 2 rows, 2,901,484 edges. |
| Edge attributes | [2901484] | 2,901,484 edge attributes. Weights are assigned to each edge |
| Y or Labels | [679779] | 679,779 labels. All nodes labelled |

# 4. Methodology

## 4.1. Model Design

Before detailing specific models, this thesis explores various approaches for fake news classification, comparing and evaluating their performance. The modelling phase progresses from traditional machine learning models to more advanced deep learning techniques. Additionally, graph-based models are implemented to capture network structures and propagation patterns.

Finally, a Hybrid GNN-Transformer model is designed to combine the strengths of textual analysis, metadata, and relational learning. In all models, both BERT embeddings and metadata attributes are used as input, ensuring a comprehensive representation of text, user interactions, and engagement patterns.

### 4.1.1. Traditional Machine Learning Models

Traditional machine learning models have been widely used in fake news detection, primarily leveraging numerical representations of text and metadata for classification tasks. In this study, two well-established models are implemented:

**Support Vector Machine (SVM)**

Support Vector Machine (SVM) is a supervised machine learning model used for classification tasks. It works by finding an optimal decision boundary (Hyperplane) that best separates data points of different classes. The main goal of SVM is to maximize the distance between the closest data points (support vectors) and the separating hyperplane (Yu and Kim, 2012)).

SVMs are widely used in text classification, image recognition, and other areas where clear decision boundaries between classes are needed

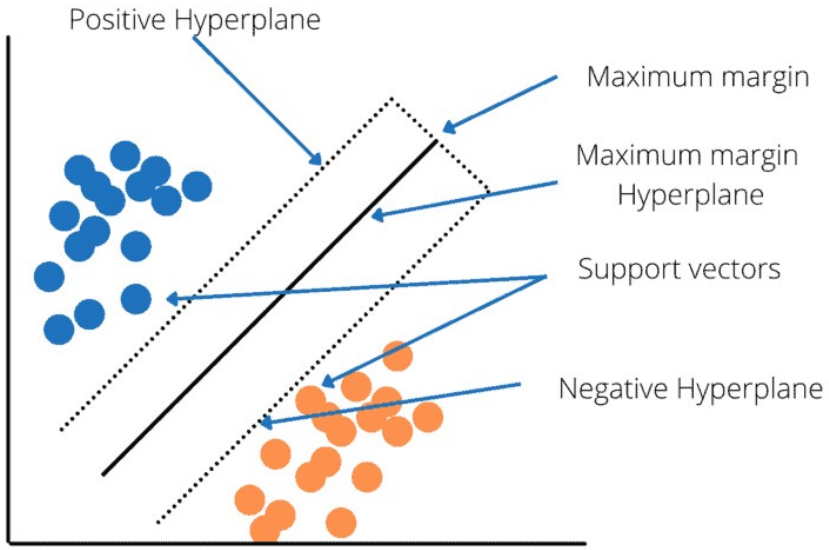



*Figure 14: SVM* ***(Bashir, 2022)***

This model was primarily designed for Binary classification tasks, classifying categories in two like the figure above. However, for multiclass (our case), were developed several strategies to address this challenge of classifying more than 2 outputs. One common approach is the one-vs-all method when multiple SVMs are trained apart, distinguishing between one class and the rest. Another technique is the one-vs-one, where a classifier is trained for each pair of classes, and a voting mechanism determines the final class.

**SVM Hyperparameter Selection for Multi-class Classification:**

To ensure the best model, a set of hyperparameters were tested such as:

Kernel Function: Determines how the model separates different types of news. Since misinformation often follows complex patterns, the Radial Basis Function (RBF) kernel helps the model detect hidden relationships that simpler methods might miss.

**Regularization Parameter (C):** Controls how strictly the model classifies news. A higher C forces the model to be more precise, while a lower C allows for slight misclassification, helping it adapt better to new data.

**Gamma (γ):** Affects how much attention the model gives to each data point. A high gamma value makes the model focus on fine details, while a low gamma value ensures broader generalization, avoiding excessive sensitivity to small variations.

**Decision Function Shape**: Since the model naturally distinguishes between only two categories, a One-vs-All (OvA) or One-vs-One (OvO) strategy is used to allow it to classify multiple types of news.

**Class Weights**: Adjusted to prevent the model from favouring more common news categories over misinformation, ensuring that fake news is correctly detected even when real news is more frequent in the dataset.

**XGBoost (eXtreme Gradient Boosting):**

This Machine Learning algorithm uses decision trees that are designed to be efficient, scalable and high-performing. It's widely used for classification and regression tasks.

XGBoost works by building an ensemble of decision trees, where each tree corrects the mistakes of the previous ones, improving predictive accuracy This approach, known as gradient boosting, allows XGBoost to minimize errors effectively while being computationally less intense. It also applies regularization techniques to prevent the model from overfitting(Baldo *et al.*, 2023).

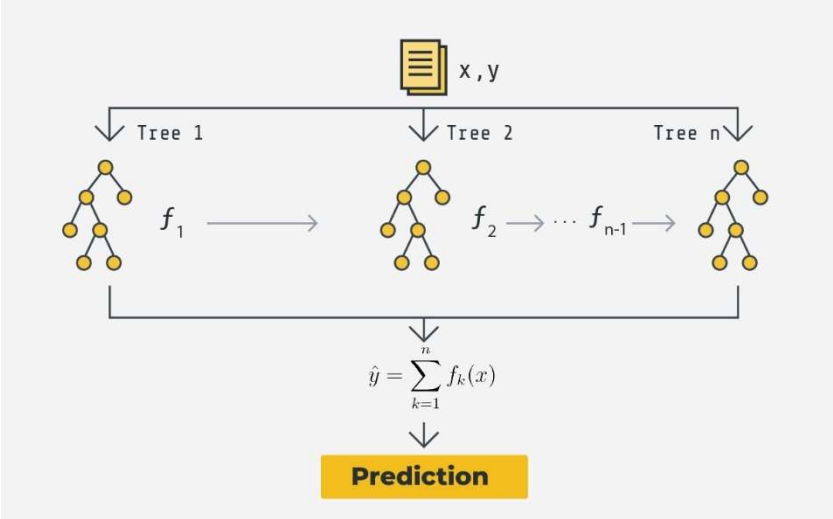



*Figure 15: XGBoost (Gao, 2023)*

SVM works well with BERT embeddings because it can handle high-dimensional data and classify complex patterns effectively. Meanwhile, XGBoost makes good use of structured metadata and runs efficiently, making it useful for understanding relationships in fake news detection.

**XGBoost Hyperparameter Selection for Multi-class Classification:**

To ensure the best model, a set of hyperparameters were tested such as (Baldo *et al.*, 2023)

**Learning Rate (learning_rate):** Controls how much the model updates with each iteration. A lower value ensures gradual learning, reducing the risk of overfitting while improving long-term accuracy.

**Maximum Depth (max_depth):** Limits the depth of individual decision trees, preventing overfitting while ensuring the model captures complex patterns in the data.

**Number of Estimators (n_estimators**): Specifies the number of boosting rounds, where more iterations can improve performance but may increase computational time.

**Number of Classes (num_class):** Defines the number of distinct categories in the dataset, ensuring the model correctly classifies multiple types of news.

**Objective Function (objective**): Set to "multi:softmax", which enables multi-class classification by outputting the class with the highest probability.

### 4.1.2. Deep Learning Models

Deep learning models offer a more advanced approach to fake news detection by learning hierarchical representations from text and metadata. Unlike traditional machine learning methods, deep learning can automatically extract patterns without relying on manually engineered features. In this study, two deep learning architectures were explored:

**Neural Network:**

Inspired by the network involving the neurons in the brain, the ANN is a powerful and versatile deep-learning technique used for large and complex problems.(with Scikit-Learn Keras and TensorFlow, 2019)

**Feedforward Neural Network (FFNN):**

FFNNs are a type of artificial neural network where connections between nodes do not form cycles. Information moves in one direction—from input nodes, through hidden nodes (if any), to output nodes. This structure is foundational in machine learning for tasks like classification and regression. Each neuron in one layer is connected to neurons in the next layer, where it applies a mathematical transformation using weights and activation functions (like ReLU or Sigmoid) to understand patterns in the data (Murat H. Sazli, 2006).

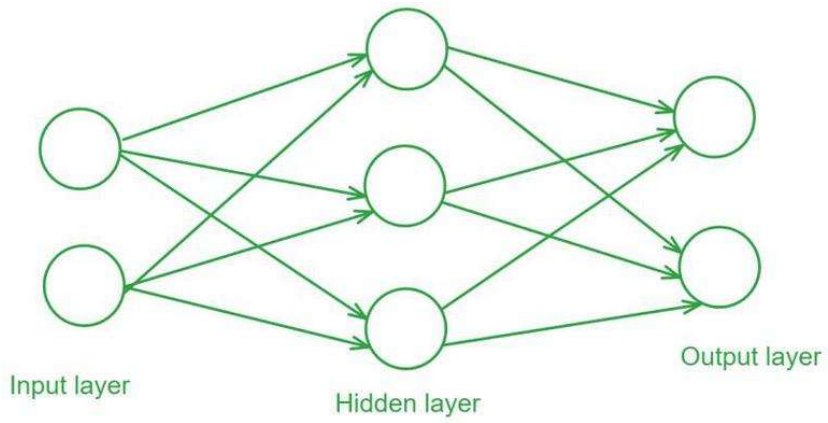



*Figure 16: FFNN (GeeksforGeeks, 2024)*

According to (Murat H. Sazli, 2006), FFNNs are well-suited for structured data classification tasks. In this paper context, provides:

Robust Generalization: FFNs are effective in learning complex patterns such as BERT embeddings from our texts

Efficient Training:  It doesn't require complexity while training, being computationally affordable due to the lack of sequential data.

It is also a good and strong benchmark model to compare with more advanced algorithms.

**FFNN Hyperparameter Selection for Multi-class Classification:**

Activation Function (activation_func): Determines how neurons process input data. ReLU is commonly used for its efficiency in deep networks, while Sigmoid and Tanh are alternatives for specific cases.

Batch Size (batch_size): Defines how many samples are processed before updating model weights. A larger batch size stabilizes training, while a smaller batch size updates more frequently but may introduce noise.

Hidden Layer Size (hidden_size): The number of neurons in the hidden layers, affects the model's ability to capture complex patterns. Too many neurons can lead to overfitting, while too few may cause underfitting.

Learning Rate (learning_rate): Controls how fast the model adjusts its weights. A higher value speeds up training but risks instability, while a lower value ensures stable learning but requires more time.

Number of Epochs (num_epochs): The number of times the model goes through the dataset. More epochs can improve learning, but excessive training may lead to overfitting.

**Recurrent Neural Network (RNN):**

A Recurrent Neural Network (RNN) is a variant of an artificial neural network,  specifically designed to process sequential data by maintaining a memory of previous inputs through hidden states. Unlike traditional feedforward networks, RNNs have loops that keep track of past information, making them a good choice for tasks like text classification, language modelling, and speech recognition (Liu, Qiu and Huang, 2016).

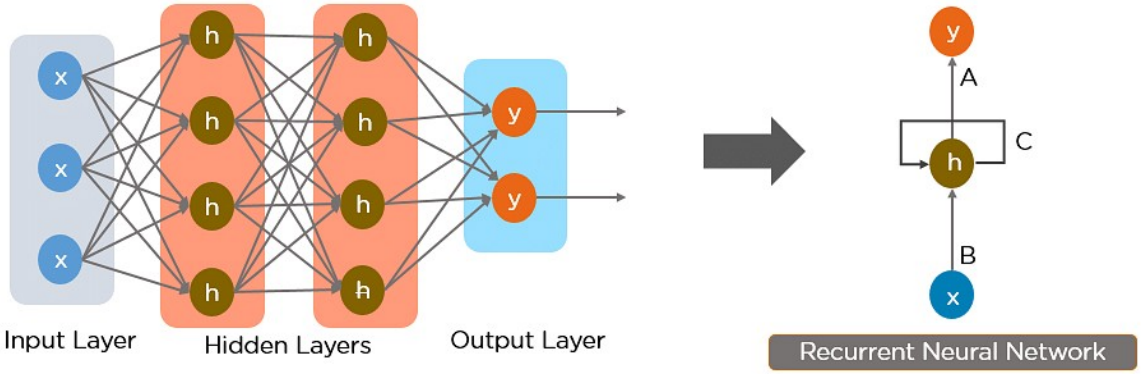



*Figure 17: Simple RNN* ***(Biswal, 2023)***

**RNN Hyperparameter Selection for Multi-class Classification:**

Hidden Size: Specifies the number of units in the hidden layer, affecting how much information the model retains and processes.

Learning Rate: Determines how quickly the model updates during training. A higher value speeds up learning but may lead to instability, while a lower value ensures gradual, more stable improvement.

Batch Size: Defines the number of samples processed together before updating the model. Larger batch sizes provide smoother updates, while smaller batch sizes allow more frequent adjustments.

Number of Epochs: Refers to how many times the model goes through the full dataset during training. More epochs improve learning but can lead to overfitting if excessive.

Activation Function: Defines how the model processes data. Tanh and ReLU are commonly used, helping the network capture patterns and relationships in the data.

### 4.1.3. Graph-Based Models (GNNs)

Traditional machine learning and deep learning models focus only on textual content but fail to capture the relational structure of information propagation. Since fake news spreads differently from real news (Shanika et al., 2020; Yuan et al., 2019), a graph-based approach is necessary.

GNNs help:

- Model user interactions (retweets, replies, mentions) as relationships.
- Learn propagation patterns that are characteristic of fake news.
- Integrate metadata, user activity, and social network behaviour alongside text features.

**Graph Convolutional Network (GCN):**

Graph Convolutional Networks (GCNs) gained a lot of attention in recent years for being capable of capturing relationships and structures in graph architectures. Unlike traditional models where data points are analysed individually, GCNs are designed to take advantage of the relationship and connections between the nodes to enhance classification accuracy (Kipf and Welling, 2017).This property makes GCNs particularly suitable for tasks where relationships among data points are crucial, such as fake news detection in social media networks.

**Fundaments of GCNs:**

*Equation 1: Graph Equation*

Given a Graph $G = (V, E)$ with:

- **V** representing the nodes (e.g., tweets, users).
- **E** representing the edges (e.g., interactions, retweets, replies).
- An Adjacency Matrix A, where $A\_ij = 1$ if nodes i and j have a connection.

GCNs apply a message-passing mechanism, where each node aggregates information from the closest nodes to refine feature representation. It's achieved by performing the graph convolutional operation repeatedly,

**Graph Convolutional Mechanism:**

At each layer of a GCN, node features are updated as follows(Kipf and Welling, 2017):

*Equation 2: GCN Mechanism:*

$$\boldsymbol{H^{(l+1)} = \sigma(\widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}H^{(l)}W^{(l)})}$$

- $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ is the adjacency matrix with self-loops added.
- $\widetilde{\boldsymbol{D}}$ is the degree matrix.
- $\boldsymbol{H}^{(l)}$ is the node feature matrix at layer l.
- $\boldsymbol{W}^{(l)}$ is the trainable weight matrix.
- $\boldsymbol{\sigma}$ is an activation function (e.g., ReLU).

This equation ensures that each node's new representation is a weighted sum of its neighbour's features, allowing the model to capture relational patterns in the data.

**Model:**

The model consists of:

- Input Layer: Receives BERT embeddings and metadata features.
- GCN Layers: Perform neighbour aggregation to refine node representations.
- Output Layer: Uses softmax activation for classification.

Final GCN Model:

*Equation 3: Final GCN Model:*

$$\boldsymbol{Z} = \boldsymbol{softmax}\big(\widetilde{\boldsymbol{A}} \times \boldsymbol{ReLU}\big(\boldsymbol{AXW}^{(\boldsymbol{0})}\big)\boldsymbol{W}^{(\boldsymbol{1})}\big)$$

Where:

- X is the initial feature matrix (BERT + metadata)
- $W^{(0)}$ and $W^{(1)}$ are learnable parameters
- Z is the predicted probability distribution over classes

**Features Integration:**

To fully leverage GCNs for fake news detection, multiple feature types were integrated:

*Table 12 - GCN features:*

| Feature Type | Description |
|---|---|
| Textual Features | BERT embeddings from tweet content. |
| User Features | Interaction delay, engagement metrics, sentiment. |
| Graph Structure | Connectivity between tweets and users (propagation). |

The GCN propagates these features, allowing the model to detect fake news based on both textual cues and propagation behaviour.

**Training the model:**

The GCN model is trained using the Cross-Entropy Loss function:

*Equation 4: Cross-Entropy Loss*

$$L = -\sum_{i=1}^{N} y_i \, log(\hat{y}_i)$$

Where:

- $y_i$ : True class label
- $\hat{y}_i$ : Predicted class probability

**GCN illustration:**

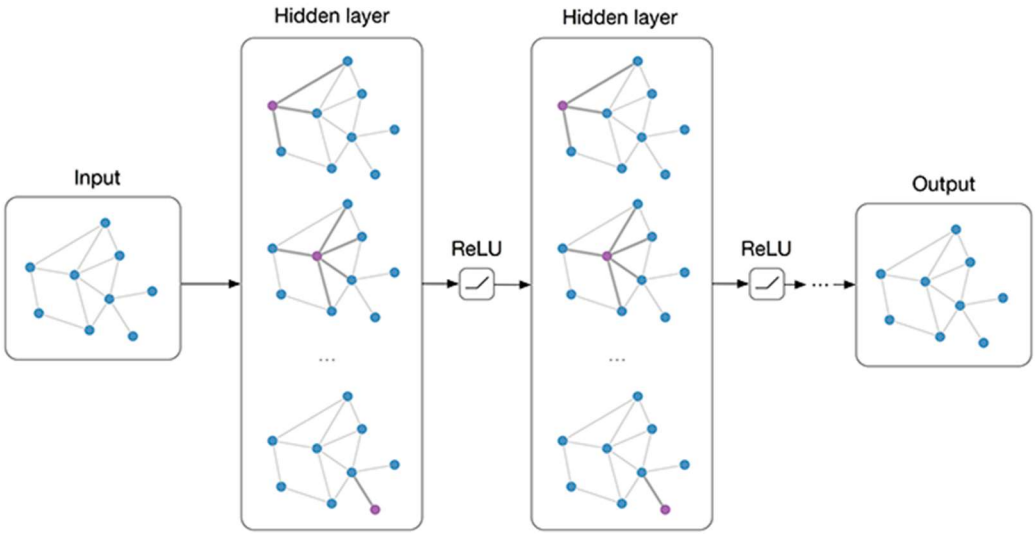



*Figure 18: Multi-layer Graph Convolutional Network (GCN) - (T. Kipf and Welling, 2016)*

This image demonstrates how a Graph Convolutional Network (GCN) operates. Here's a breakdown of the process:

**Input:**

The graph consists of nodes (blue circles) connected by edges. Each node contains input features, such as BERT embeddings and metadata, which are essential for tasks like fake news detection.

**Hidden Layers:**

In each hidden layer, the network performs neighbourhood aggregation or message passing, where each node updates its features by combining information from its neighbouring nodes (those directly connected to it). This process refines the node's representation, capturing its local context and relationships within the graph.

**Activation Function (ReLU):**

After each aggregation step, a Rectified Linear Unit (ReLU) activation function is applied. This introduces non-linearity, allowing the model to learn and represent more complex patterns in the data.

**Output Layer:**

The final output layer generates refined node representations or class predictions, depending on the specific task. For example, in fake news detection, this could involve classifying tweets as either fake or real based on the learned features.

### 4.1.4. Hybrid GCN-Transformer for Fake News Detection

Traditional GCN models are effective at capturing node-level relationships but struggle with long-range dependencies (Kipf and Welling, 2017) On the other hand, Transformer-based models capture global dependencies but do not naturally handle graph structures (Yun, Jeong, Kim, Kang and Hyunwoo J Kim, 2019). This study integrates both approaches to enhance fake news detection, leveraging local graph relationships and global attention mechanisms.

**GCN-Transformer benefits:**

Fake news spreads differently from real news, forming distinct propagation patterns (Han, Karunasekera and Leckie, 2020; (Yun et al., 2019). While GCNs learn how information propagates, they struggle with capturing complex textual and contextual dependencies across a network. In contrast, Transformers can capture semantic relationships across all nodes, regardless of direct graph connectivity (Dwivedi et al., 2021).

This hybrid approach enables the model to effectively integrate multiple learning techniques. The Graph Convolutional Network (GCN) allows it to understand graph-based structures, capturing the relationships between nodes. Meanwhile, the self-attention mechanisms in the Transformer enhance the model's ability to analyze global interactions, ensuring a more comprehensive understanding of misinformation spread. By combining textual content, metadata, and propagation patterns, this approach improves classification accuracy, making the detection of fake news more robust and adaptable to evolving misinformation tactics.

**Graph Representation & Feature Extraction**

The model starts by representing social media interactions as a graph. Nodes represent users and tweets, while edges define the relationships between them, such as retweets, replies, and interactions. Each node contains feature embeddings derived from BERT (for textual features) and metadata (for engagement and user characteristics). This graph structure captures how information propagates across different entities.

**GCN-Transformer layers**

GCN is responsible for learning node embeddings by aggregating information from neighbouring nodes. Following (Kipf and Welling, 2017) this step updates node representations using message passing.

The GCN layer is formulated as Equation 1 above:

Processing Steps:

Information Aggregation: The nodes in the graph don't rely on their structure only but also on the surrounding neighbours. That grants the model the ability to instead of treating each node separately, enhance its representation by considering also its connections.

Updating Node Representation: After collecting information from its neighbours, it updates its features based on what is learned from them. This allows the model to capture relationships rather than just individual node characteristics.

Weighted influence of Neighbours: Nodes that are strongly connected have a greater influence on each other.

Non-Linearity with ReLU: After the process of updating the node features, we apply the ReLU activation function to introduce non-linearity. This step is crucial because real-world relationships aren't just as simple as linear combinations. By acknowledging it, ReLU helps to identify them by filtering unnecessary or less important information.

In essence, GCN layers allow each node to iteratively refine its representation by incorporating knowledge from its local graph structure while ensuring the model can capture deeper, more complex patterns.

**Transformer layers: Capturing Long-Range Dependencies**

Once GCN embeddings are generated, they are passed into a Transformer encoder, which captures relationships beyond local neighbourhoods.

*Equation 5: The multi-head self-attention mechanism is defined (Vaswani et al., 2017):*

$$\boldsymbol{Attention}\ (\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{softmax}\left(\frac{\boldsymbol{Qk^T}}{\sqrt{\boldsymbol{d_k}}}\right)\boldsymbol{V}$$

Where:

- Q, K, V are query, key and value metrics from our node features.
- $d_k$ is the dimensionality of the key vectors.
- The softmax function determines the importance of each node's relationship with all other nodes.

After passing through the graph convolutional layers, the node embeddings are processed by the Transformer module. This step ensures that nodes do not rely solely on local information aggregated by the GCN but also capture long-range dependencies within the graph. Instead of strictly following predefined graph structures, the Transformer layer allows nodes to dynamically focus on the most relevant neighbouring nodes, assigning varying levels of importance to each connection. By incorporating a multi-head attention mechanism, the Transformer breaks down the representation into multiple subspaces, extracting different relational aspects and improving the model's ability to detect patterns in misinformation propagation.

*Equation 6: Multi-head attention mechanism:*

$$\boldsymbol{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{Concat}(\boldsymbol{head1}, \ldots, \boldsymbol{headh})\boldsymbol{W^O}$$

Where:

- $head_i = Attention(QWiQ, KWiK, VWiV)$ for each attention head.
- $W^O$ is the final output weight matrix.

Inside the Transformer encoder, several key operations refine the node embeddings to enhance the model's ability to capture meaningful relationships. The self-attention mechanism adjusts each node's representation based on its relevance to other connected nodes, ensuring that important interactions are highlighted. The scaled dot product attention dynamically assigns weights to neighbouring nodes, allowing the model to focus on essential relationships within the graph. Following this, a multi-layer perceptron (MLP) further refines the extracted features, improving representation learning. Lastly, layer normalization stabilizes the learned embeddings, preventing drastic shifts in data distribution and ensuring consistent performance across different graph structures.

**Final Node Embeddings & Classification.**

The final node embeddings, enriched by both local structures (GCN) and global interactions (Transformer), pass through a fully connected layer with softmax activation for classification:

*Equation 7: Final Classification:*

$$\boldsymbol{Z = softmax\left(A \times ReLU\left(XW^{(0)}\right)W^{(1)}\right)}$$

Where:

- A is the learned graph structure with attention-enhanced adjacency matrices.
- X is the input feature matrix (e.g., BERT embeddings + metadata).
- $W^{(0)}, W^{(1)}$) $are\ trainable\ weight\ matrices.$

After passing through both the GCN and Transformer layers, each node receives an enhanced embedding that combines both local graph structure and global contextual relationships. The GCN component captures propagation patterns by aggregating information from neighbouring nodes, while the Transformer refines these embeddings by identifying important connections beyond direct neighbours. This integration ensures that the model learns both structural and semantic dependencies, making it more effective in detecting misinformation. The resulting refined embeddings serve as inputs for the final classification step, where each node (tweet) is categorized as Fake, Real, Unverified, or Non-Rumor based on the learned features.

# 5. Experiments and Results

## 5.1. Experimental Setup:

Processing Graph Neural Networks (GNNs), especially GCN-Transformer hybrid models, demands substantial computational resources due to high-dimensional embeddings and large graph-based architecture. The experimental setup consists of:

*Table 13: Experimental Setup*

| Operational System | GPU | CPU | RAM | Storage | Python | PyTorch |
|---|---|---|---|---|---|---|
| Ubuntu 22.04 (WSL and ROOT) | Radeon RX 7900 XTX | AMD Ryzen 9 7900X 12-Core Processor 4.70 GHz | 32GB DDR5 | NVMe SSD for faster data access | 3.10+ | 2.5.1+rocm6.2 & PyTorch Geometric |

Additionally, all necessary libraries and dependencies used in this setup are listed in the requirements.txt file, ensuring reproducibility and easy installation for future experiments.

## 5.2. Evaluation Metrics

The Confusion Matrix offers a detailed breakdown of correct and incorrect predictions across all labels—Fake, Real, Unverified, and Non-Rumor—helping identify specific areas where the model struggles.

The Classification Report provides key performance metrics, including precision, recall, and F1-score, allowing a deeper assessment of class-wise performance. By analyzing these metrics, we can

determine whether the model performs consistently across all labels and detect any discrepancies in classification accuracy(M and M.N, 2015).

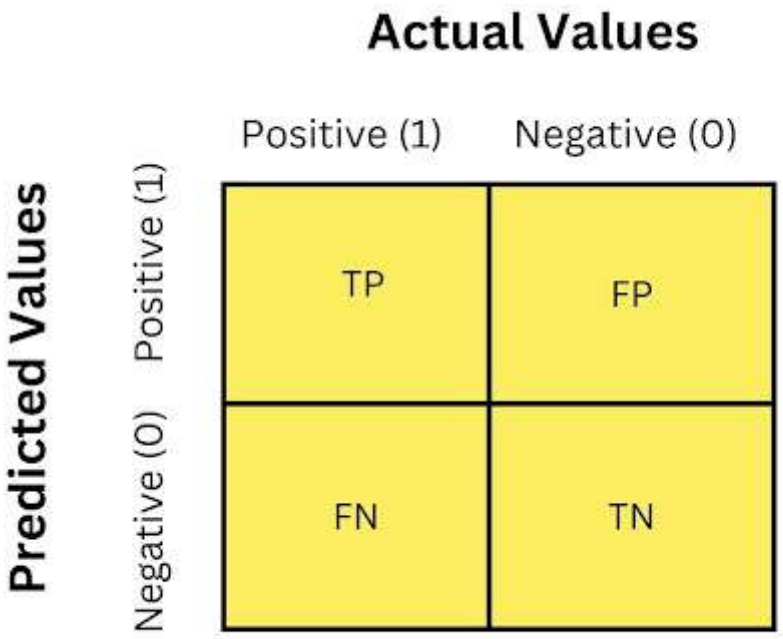



*Figure 19: Confusion Matrix (H2O.ai, no date)*

**True positive(TP)**: Predicted a positive outcome and the prediction is true
**True negative(TN)**: Predicted a negative outcome, and that prediction is true
**False positive(FP)**: Predicted a positive outcome and the prediction is false.
**False-negative(FN)**: Predicted a negative outcome and the prediction is false.

This thesis applies several models to analyse Fake news dissemination and evaluates them using five different metrics:

- **Accuracy:** measures the proportion of correctly classified instances, offering a general performance indicator. Since our dataset is balanced, it remains a reliable metric.

It consists of the ratio of predicted observation to the total of comments:

*Equation 8: Accuracy:*

$$\frac{TP\ +\ TN}{TP + TN + FP + FN}$$

- **Precision:** How much positive out of all positive predictions:

*Equation 9: Precision:*

$$\frac{TP}{TP + FP}$$

- **Recall:** Ratio of correctly predicted positive observations to all observations in actual class:

*Equation 10: Recall:*

$$\frac{TP}{TP + FN}$$

- **F1score:** Weighted average of Precision and Recall. It has a better result than Accuracy when having uneven distribution:

*Equation 11: F1-Score:*

$$\frac{2 * (Precision * Recall)}{precision + recall}$$

As discussed in (M and M.N, 2015), ROC-AUC is more effective for binary classification and imbalanced datasets. Since our dataset is multi-class and balanced, ROC-AUC does not provide meaningful insights compared to a confusion matrix and class-wise performance metrics. Instead, we focus on interpretable metrics that highlight model strengths and misclassification patterns.

By using accuracy, confusion matrix, and classification report, we ensure a clear and practical evaluation that supports model improvements.

### 5.3. Cross-Validation Strategy:

Cross-validation is a statistical method used to evaluate the performance of a Machine Learning model by dividing the original dataset into multiple subsets. This technique helps to identify how well the model performs in new, unseen data and therefore helps to avoid overfitting. The main objective is to check if the model keeps consistent accuracy across various parts of the dataset.

Knowing our data isn't unbalanced, we opted for one of the most common Cross-Validation techniques, the K-Fold Cross-Validation (scikit-learn, 2018).

*Table 14: Cross Validation methods:*

| Method | How It Works | Best For |
|---|---|---|
| Stratified K-Fold | Like K-Fold but preserves class distribution in each fold. | Imbalanced classification problems. |
| Leave-One-Out (LOO) | Uses 1 sample as the test set, and trains on the rest. Repeats for all samples. | Small datasets, very computationally expensive. |
| Leave-P-Out (LPO) | Similar to LOO but leaves P samples out for testing. | Custom validation strategies. |
| Shuffle & Split | Randomly shuffles and splits data into train/test multiple times. | Fast and simple evaluation. |
| Time-Series Split | Keeps chronological order, trains on past data, and tests on future data. | Time-series |

In K-Fold Cross-Validation the dataset is split into k equally sized subsets (or folds). The model is trained on k-1 and then tested on the remaining folds. The process repeats until every fold is tested individually. The performance measure is then the average of each iteration.

This method ensures each data point is included in training and validation, helping the model develop a stronger and more reliable understanding of the data, leading to improved accuracy and generalization (scikit-learn, 2018).

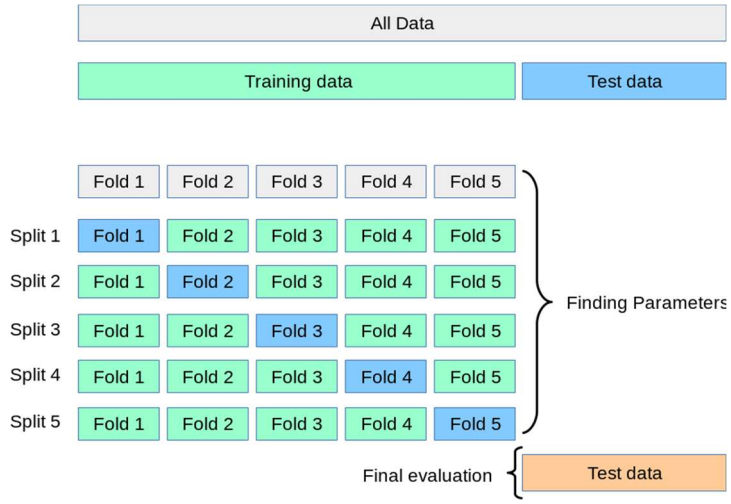



*Figure 20: Cross-Validation (scikit-learn, 2018)*

For PyTorch techniques, the Cross-Validation Process require to be manually performed, unlike the automated version of scikit-learn.

## 5.4. Traditional Methods Performance

**Support Vector Machine Model:**

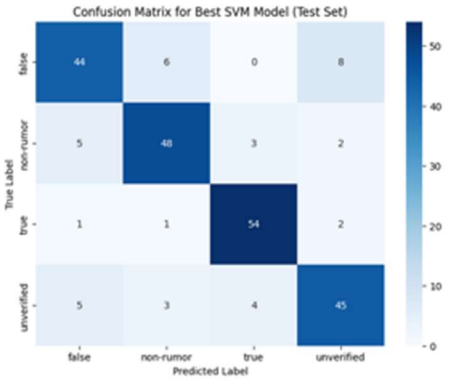
In the experimental phase, we first performed the Support Vector Machine Classifier. Before training our model we first ran a hyperparameter tunning so we could have the best result possible from this model.

After running a GridSearchCV for SVM, the results obtained were:

*Table 15: SVM Best Hyperparameters*

| Parameter | Value |
|---|---|
| **C** | 10 |
| **decision_function_shape** | ovo |
| **gamma** | scale |
| **kernel** | rbf |

After determining the optimal hyperparameters, the model was trained and evaluated on the validation set, achieving an accuracy of 82.68%. To further assess its performance, the model was tested on the test set, with results analysed through a Confusion Matrix and Classification Report.




```
Classification Report for GridSearch SVM (Test Set):
              precision    recall  f1-score   support

       false       0.80      0.76      0.78        58
  non-rumor        0.83      0.83      0.83        58
        true       0.89      0.93      0.91        58
  unverified       0.79      0.79      0.79        57

    accuracy                           0.83       231
   macro avg       0.83      0.83      0.83       231
weighted avg       0.83      0.83      0.83       231
```

*Figure 21: SVM Performance*

With an overall accuracy exceeding 82%, the SVM model performed well, particularly in classifying truthful tweets. According to the Classification Report and Confusion Matrix, 54 out of 58 truthful tweets were correctly classified, with only four misclassifications. However, the model struggled more with fake news and rumour tweets, where accuracy was 9% lower, and recall was the weakest at 76%, indicating a challenge in correctly identifying these categories.

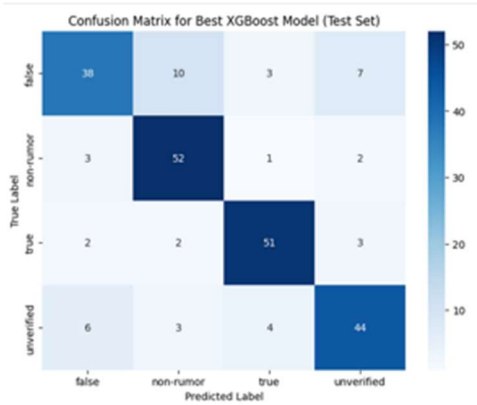
**XGBoost Model:**

For the second proposed model, XGBoost, hyperparameter tuning was conducted using GridSearchCV. The best-performing hyperparameters were found to be:

*Table 16: XGBoost Best Hyperparameters*

| Parameter | Value |
|---|---|
| learning_rate | 0.2 |
| max_depth | 3 |
| n_estimators | 200 |
| num_class | 4 |
| objective | multi:softmax |

The model's performance was optimized through cross-validation, ensuring the highest possible accuracy. After training on the validation set, the XGBoost model achieved an accuracy of 74.9%. To further evaluate its effectiveness, it was tested on the test set, with results analysed using a Confusion Matrix and Classification Report.




```
Classification Report for GridSearch XGBoost (Test Set)
              precision    recall  f1-score   support

       false       0.78      0.66      0.71        58
  non-rumor        0.78      0.90      0.83        58
        true       0.86      0.88      0.87        58
  unverified       0.79      0.77      0.78        57

    accuracy                           0.80       231
   macro avg       0.80      0.80      0.80       231
weighted avg       0.80      0.80      0.80       231
```

*Figure 22: XGBoost Performance*

With an overall accuracy of 80%, the XGBoost model demonstrated strong performance in classifying truthful tweets, achieving an accuracy of 86% and a recall of 88%. However, it struggled with detecting false information, reaching 78% accuracy but with the weakest recall of 66%, indicating difficulty in correctly identifying misinformation.

## 5.5. Neural Network Model Performance

- **Feed Forward Neural Network (FFNN)**

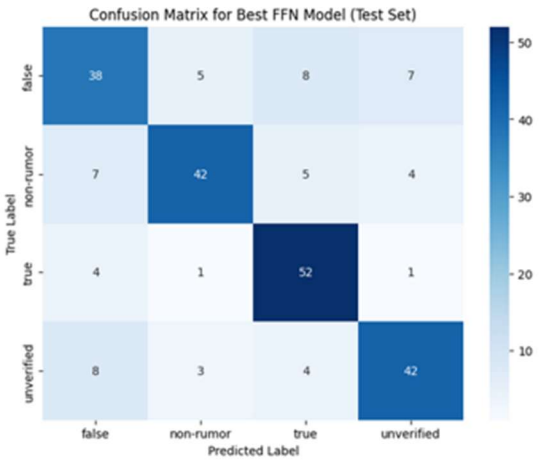
For the proposed FFNN model, we also conducted performance optimization by hyperparameter tunning. The best set of parameters is as follows:

*Table 17: FFNN Best Hyperparameters*

| Parameter | Value |
|---|---|
| activation_func | ReLU |
| batch_size | 32 |
| hidden_size | 256 |
| learning_rate | 0.001 |
| num_epochs | 10 |

The optimal set of hyperparameters was determined through cross-validation, leading to the final training and evaluation of the best-performing model on the validation set, achieving a validation accuracy of 71%

Following this, the model was trained on the full training set and evaluated on the test set, with results analyzed through a Confusion Matrix and Classification Report.




Classification Report for Best FFNN Model:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| false | 0.67 | 0.66 | 0.66 | 58 |
| non-rumor | 0.82 | 0.72 | 0.77 | 58 |
| true | 0.75 | 0.90 | 0.82 | 58 |
| unverified | 0.78 | 0.74 | 0.76 | 57 |
| accuracy | | | 0.75 | 231 |
| macro avg | 0.76 | 0.75 | 0.75 | 231 |
| weighted avg | 0.76 | 0.75 | 0.75 | 231 |

*Figure 23: FFNN Performance*

The Feed Forward Neural Network (FFNN) achieved an overall accuracy of 75%. It performed well on non-rumour tweets, attaining 82% accuracy and 72% recall, and demonstrated strong performance in classifying truthful tweets with 75% accuracy and 90% recall. However, the model struggled with detecting false information, reaching 78% accuracy but the weakest recall of 66%, highlighting difficulties in correctly identifying misinformation.

**Recurrent Neutral Network (RNN)**

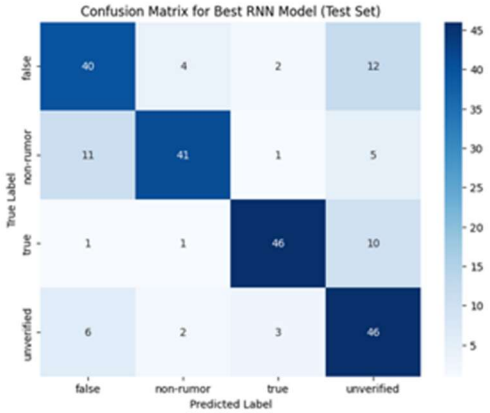
Similarly, for the proposed RNN model, after conducting fine-tuning hyperparameters the most effective configuration obtained was:

*Table 18: RNN Best Hyperparameters*

| Parameter | Value |
|---|---|
| activation_func | sigmoid |
| batch_size | 96 |
| dropout_rate | 0.3 |
| hidden_size | 128 |
| learning_rate | 0.01 |
| num_epochs | 10 |

After performing cross-validation and selecting the optimal hyperparameters, The Recurrent Neural Network (RNN) was trained on the training set and validated on the validation set, achieving an accuracy of 69%.

Once fully trained, the model was evaluated on the test set, with performance measured using Confusion Matrix and Classification Report:




Classification Report for Best RNN Model:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| false | 0.69 | 0.69 | 0.69 | 58 |
| non-rumor | 0.85 | 0.71 | 0.77 | 58 |
| true | 0.88 | 0.79 | 0.84 | 58 |
| unverified | 0.63 | 0.81 | 0.71 | 57 |
| accuracy | | | 0.75 | 231 |
| macro avg | 0.76 | 0.75 | 0.75 | 231 |
| weighted avg | 0.77 | 0.75 | 0.75 | 231 |

*Figure 24: RNN Performance*

The RNN achieved an overall accuracy of 75%, performing well in classifying truthful tweets with 88% accuracy and 79% recall. However, its performance in detecting false information was notably weaker, with an accuracy of 69% and a recall of 69%, indicating difficulties in distinguishing misinformation effectively.

## 5.6. Graph Convolutional Network Performance:

Instead of using cross-validation and hyperparameter tuning, the GCN and GCN-transformer models were only trained and tested within a certain set of parameters. This approach was chosen to evaluate under standard conditions while maintaining computational resources manageable. Future research could take advantage of this gap.

**Model Architecture Hyperparameters:**

*Table 19: Architecture Hyperparameters*

| Hyperparameter | Value | Description |
|---|---|---|
| in_channels | data.x.size(1) | Number of input features per node |
| hidden_channels | 128 | Number of hidden layer features |
| out_channels | torch.unique(data.y[data.y != -1]).size(0) | Number of unique output classes (excluding -1 for unlabelled nodes) |

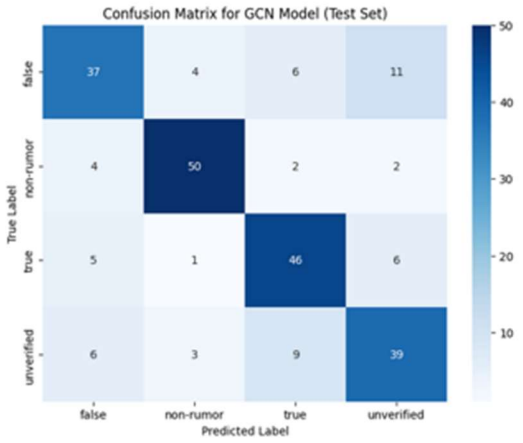
**Training Hyperparameters:**

*Table 20: GCN Training Hyperparameters*

| Hyperparameter | Value | Description |
|---|---|---|
| num_epochs | 100 | Number of training iterations |
| learning_rate | 0.001 | Step size for weight updates |
| weight_decay | 5e-4 | Regularization to prevent overfitting |
| dropout_rate | 0.5 | Randomly disables neurons during training |
| batch_size | 256 | Number of samples per batch (if using mini-batching) |
| optimizer | Adam | Optimization algorithm used for training |

After training and validating our Graph Convolutional Network (GCN) using the hyperparameters listed above, achieved the maximum Accuracy of 80% :

With the model fully trained, it is evaluated using the test set, and its performance is summarised in the Confusion Matrix and Classification Report as follows:




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| false | 0.71 | 0.64 | 0.67 | 58 |
| non-rumor | 0.86 | 0.86 | 0.86 | 58 |
| true | 0.73 | 0.79 | 0.76 | 58 |
| unverified | 0.67 | 0.68 | 0.68 | 57 |
| accuracy | | | 0.74 | 231 |
| macro avg | 0.74 | 0.74 | 0.74 | 231 |
| weighted avg | 0.74 | 0.74 | 0.74 | 231 |

*Figure 25: RNN Performance*

The Overall Accuracy of our GCN model was 74%, with 71% precision and 64% recall for false information/rumour and 86% precision and recall for non-rumour following 73%. Precision and 79% recall for true

This model underperformed for unverified labels, showing a precision of 67%, recall of 68% and F1-score of 68%

### 5.7. Hybrid Graph Convolutional Network-Transformer (GCN-Transformer) Performance:

**Model Architecture Hyperparameters:**

*Table 21: GCN-Transformer Model Architecture*

| Parameter | Value | Description |
|---|---|---|
| in_channels | data.x.size(1) | Number of input features per node |
| hidden_channels | 128 | Number of hidden layer features |
| out_channels | torch.unique(data.y[data.y != -1]).size(0) | Number of unique output classes (excluding -1 for unlabelled nodes) |
| num_heads | 4 | Number of attention heads in Transformer layers |
| num_transformer_layers | 2 | Number of Transformer layers |
| ff_hidden_dim | 128 | Hidden dimension in a feedforward network |
| dropout_rate | 0.1 | Dropout rate for regularisation |
| GCN Layers | 2 | Number of graph convolution layers |
| Activation Function | ReLU | Non-linearity applied after each layer |

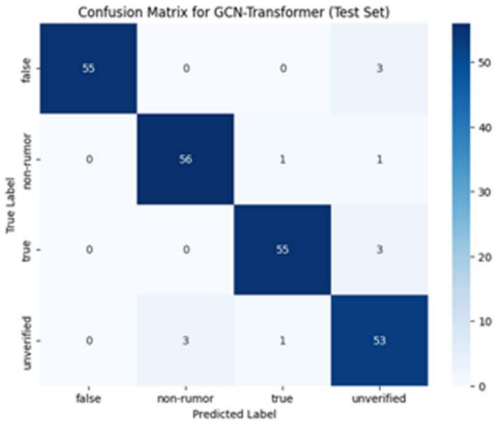
**Training Hyperparameters:**

*Table 22: GCN-Transformer Hyperparameters*

| Hyperparameter | Value | Description |
|---|---|---|
| num_epochs | 100 | Total number of training iterations |
| batch_size | 256 | Number of samples per batch in NeighborLoader |
| learning_rate | 0.001 | Step size for weight updates |
| weight_decay | 5e-4 | Regularization to prevent overfitting |
| optimizer | Adam | Optimization algorithm for training |
| loss_function | Cross Entropy Loss | Loss function for classification |

After training and validating our Graph Convolutional Network with a Transformer (GCN-Transformer) using the hyperparameters listed above, we achieved a 95.63% accuracy:

Once training was completed, we tested the model to see how well it performed. The results are shown in the Confusion Matrix and Classification Report below:




Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| false | 1.00 | 0.95 | 0.97 | 58 |
| non-rumor | 0.95 | 0.97 | 0.96 | 58 |
| true | 0.96 | 0.95 | 0.96 | 58 |
| unverified | 0.88 | 0.93 | 0.91 | 57 |
| accuracy | | | 0.95 | 231 |
| macro avg | 0.95 | 0.95 | 0.95 | 231 |
| weighted avg | 0.95 | 0.95 | 0.95 | 231 |

*Figure 26: GCN-Transformer Performance*

The GCN-Transformer achieved an accuracy of nearly 95%. Reaching, for false labels, a very good result of 100% precision, 95% Recall, and 97% f1-score.

Unverified has shown a lower performance than other labels, with 88% Precision, 93% Recall, and 91% f1-score.

## 5.8. Performance Comparison:

**Overall Analysis of the proposed model's performance:**

*Table 23: Overall Performance Comparison*

| Model | Test Accuracy | Precision (Avg.) | Recall (Avg.) | F1-Score (Avg.) |
|---|---|---|---|---|
| **SVM** | 82.68% | 0.83 | 0.83 | 0.83 |
| **XGBoost** | 80.09% | 0.80 | 0.80 | 0.80 |
| **Best FFNN** | 75% | 0.75 | 0.75 | 0.75 |
| **Best RNN** | 75% | 0.75 | 0.75 | 0.75 |
| **GCN** | 77% | 0.77 | 0.76 | 0.76 |
| **GCN-Transformer** | **94.81%** | **0.95** | **0.95** | **0.95** |

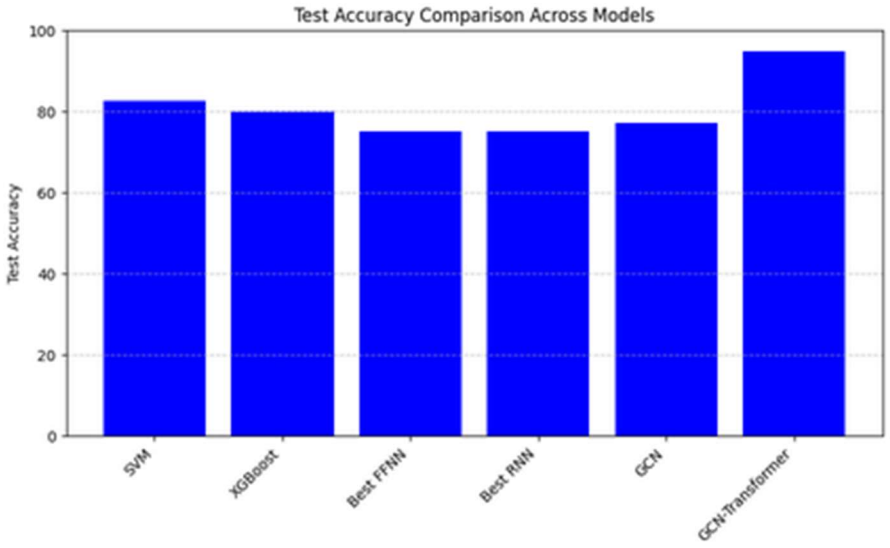



*Figure 27: Accuracy Across Models*

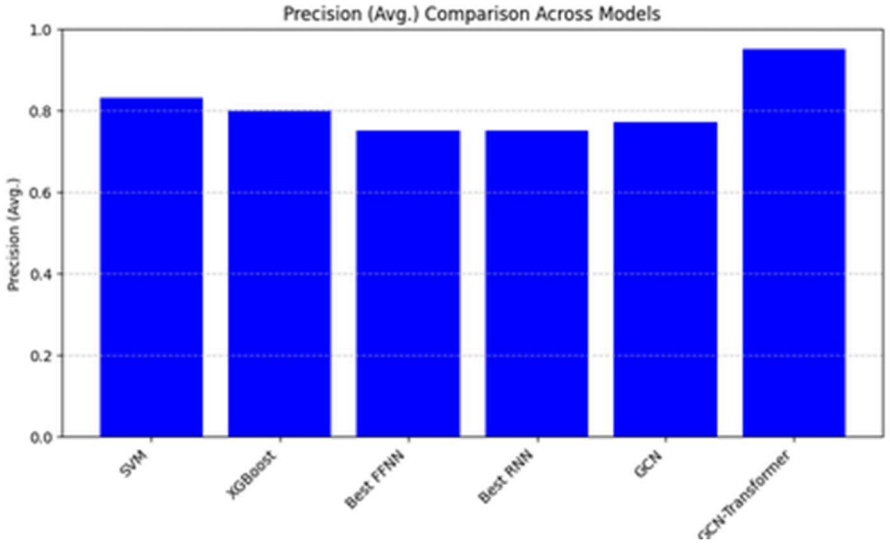



*Figure 28: Average Precision across Models*

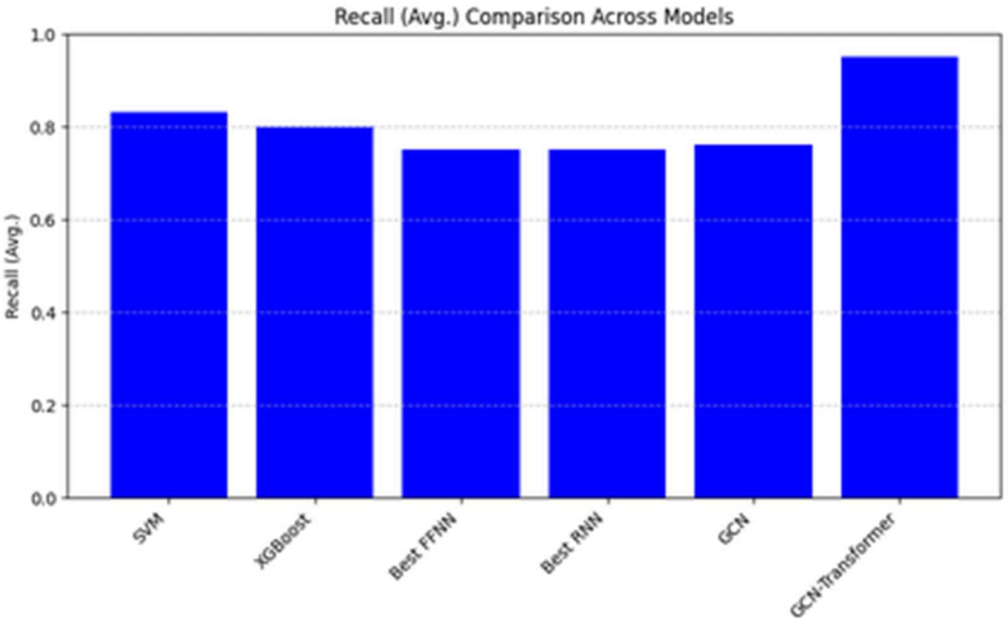



*Figure 29: Average Recall Across the Models*

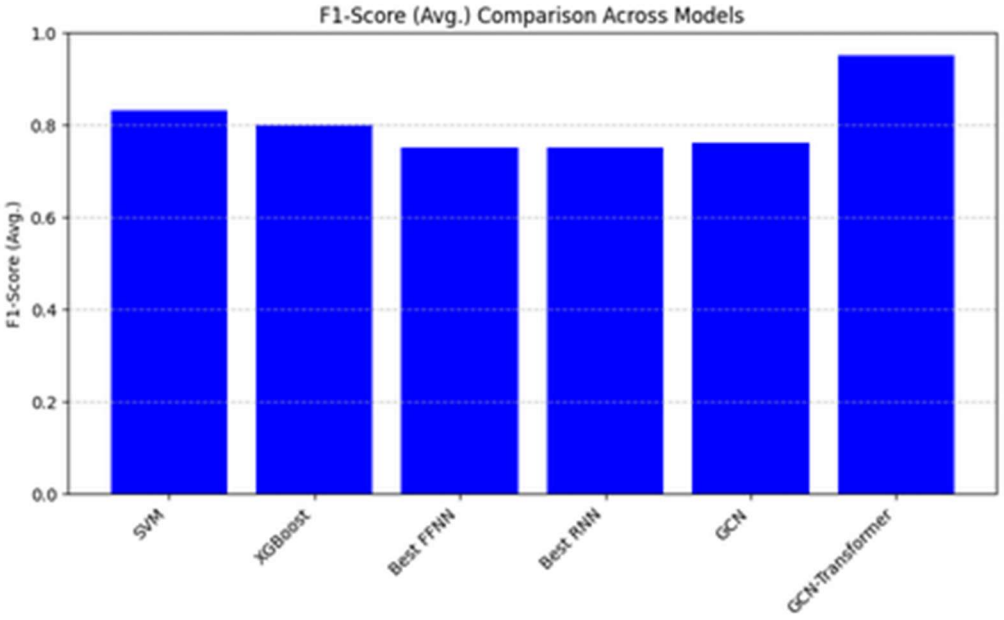



*Figure 30: Average F1-Score Across the Models*

## 5.9. False Label Performance:

*Table 24: False Label Performance comparison*

| Model | False Precision | False Recall | False F1 |
|---|---|---|---|
| Best SVM | 0.80 | 0.76 | 0.78 |
| Best XGBoost | 0.78 | 0.66 | 0.71 |
| Best FFNN | 0.67 | 0.66 | 0.66 |
| Best RNN | 0.69 | 0.69 | 0.69 |
| GCN | 0.79 | 0.64 | 0.70 |
| GCN-Transformer | 1.00 | 0.97 | 0.98 |

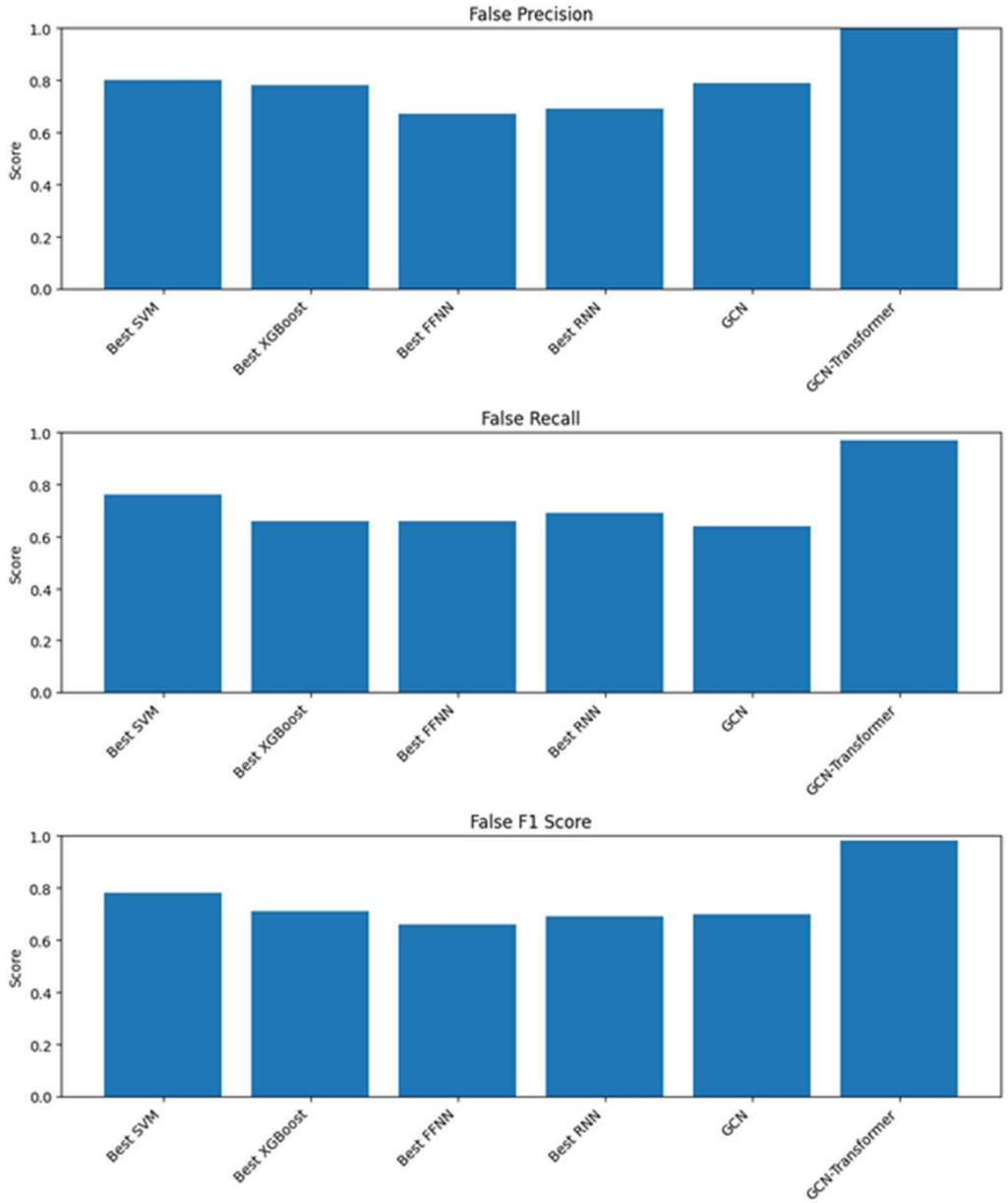



*Figure 31: False Label Performance Comparison*

## 5.10. Non-Rumour Label Performance:

*Table 25: Non-Rumour Label Performance Comparison*

| Model | Non-Rumour Precision | Non-Rumour Recall | Non-Rumour F1 |
|---|---|---|---|
| Best SVM | 0.83 | 0.83 | 0.83 |
| Best XGBoost | 0.78 | 0.90 | 0.83 |
| Best FFNN | 0.82 | 0.72 | 0.77 |
| Best RNN | 0.85 | 0.71 | 0.77 |
| GCN | 0.82 | 0.84 | 0.83 |
| **GCN-Transformer** | **0.96** | **0.95** | **0.89** |

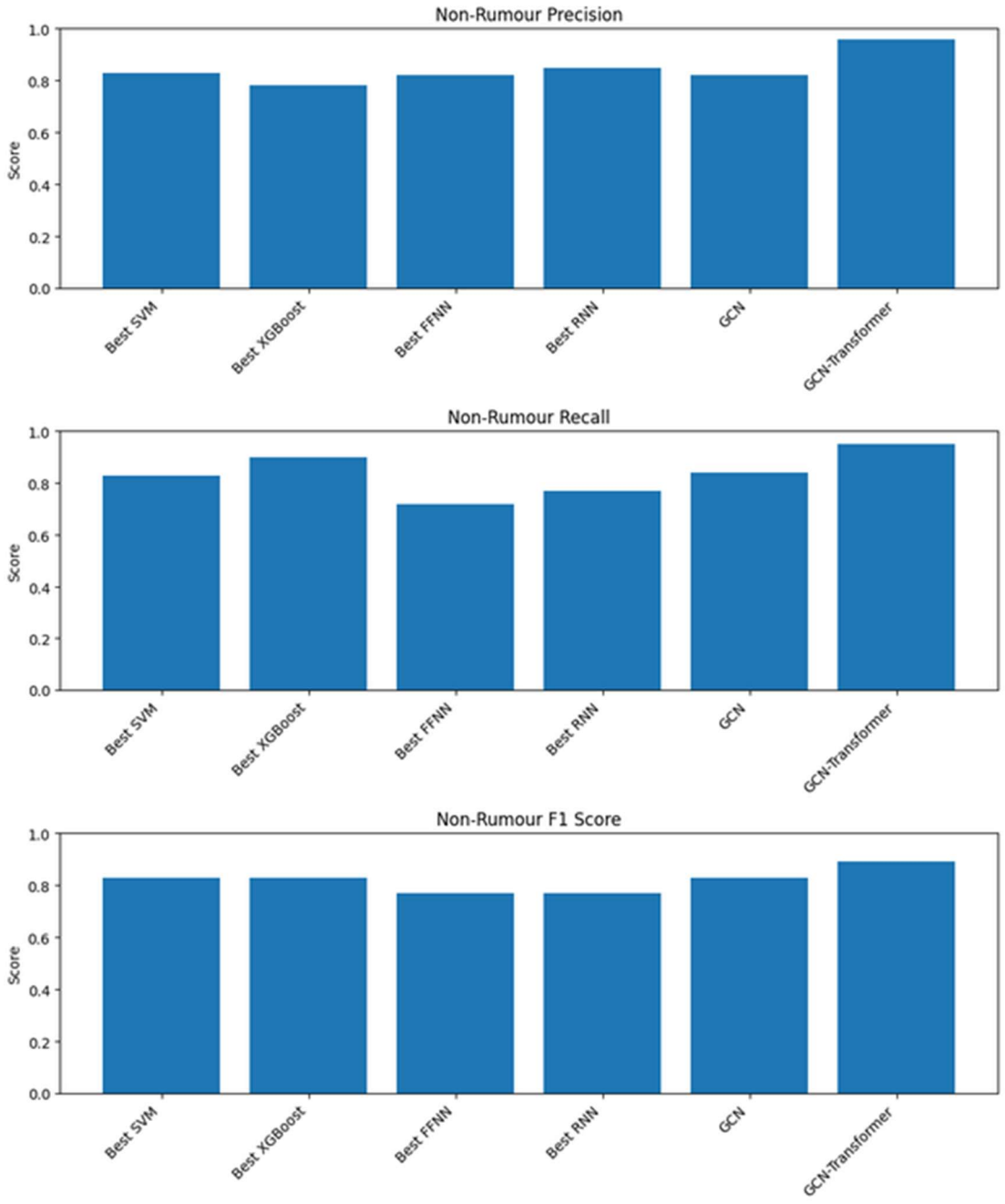



*Figure 32: Non-Rumour Label Performance Comparison*

## 5.11. True Label Performance:

*Table 26: True Label Performance Comparison*

| Model | True Precision | True Recall | True F1 |
|---|---|---|---|
| Best SVM | 0.89 | 0.93 | 0.91 |
| Best XGBoost | 0.86 | 0.88 | 0.87 |
| Best FFNN | 0.75 | 0.90 | 0.82 |
| Best RNN | 0.88 | 0.79 | 0.84 |
| GCN | 0.76 | 0.91 | 0.83 |
| **GCN-Transformer** | **0.98** | **0.93** | **0.96** |

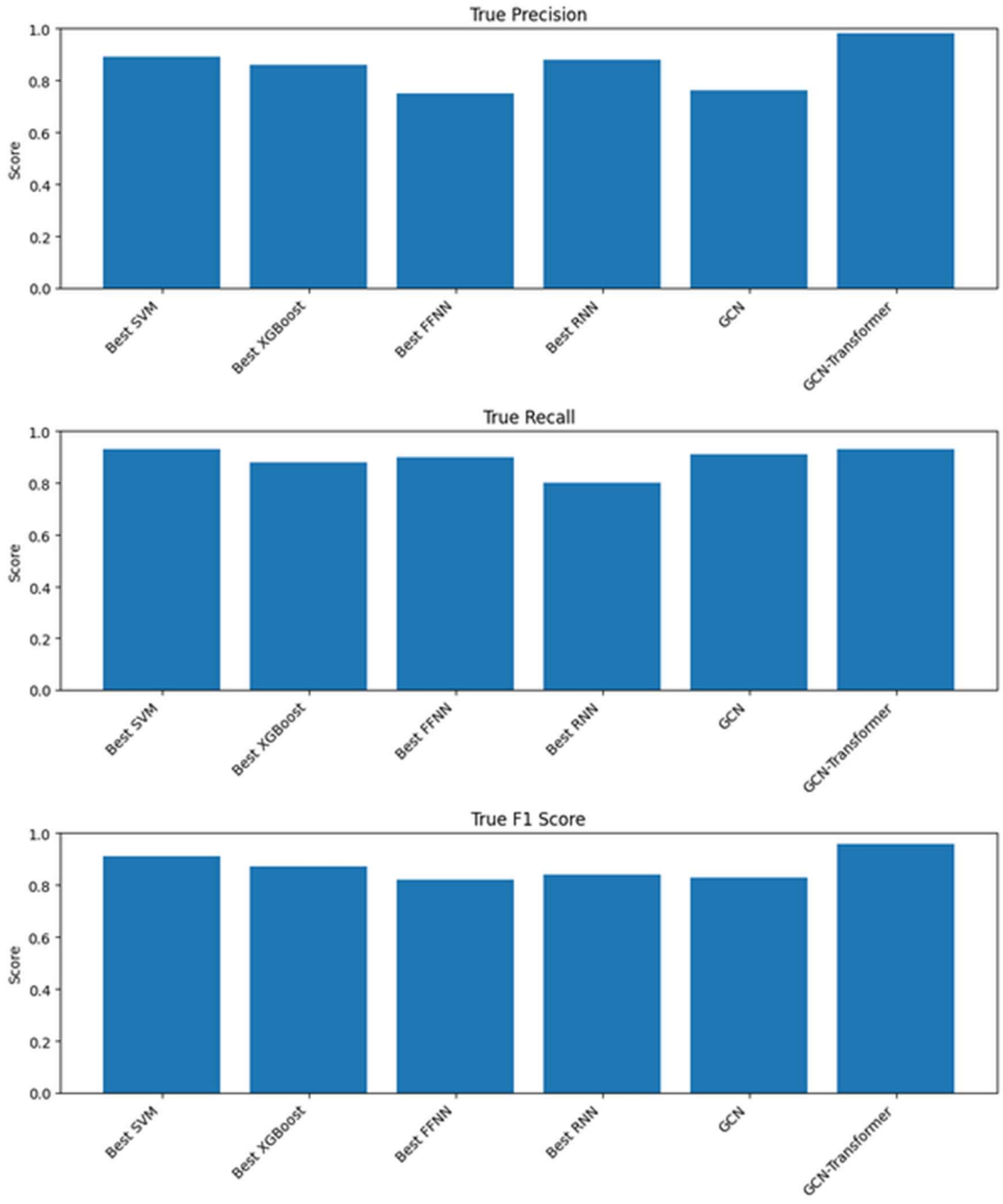



*Figure 33: True Label Performance Comparison*

## 5.12. Unverified Label Performance:

*Table 27: Unverified Label Performance Comparison*

| Model | Unverified Precision | Unverified Recall | Unverified F1 |
|---|---|---|---|
| Best SVM | 0.79 | 0.79 | 0.79 |
| Best XGBoost | 0.79 | 0.77 | 0.78 |
| Best FFNN | 0.78 | 0.74 | 0.76 |
| Best RNN | 0.63 | 0.81 | 0.71 |
| GCN | 0.70 | 0.67 | 0.68 |
| **GCN-Transformer** | **0.94** | **0.89** | **0.92** |

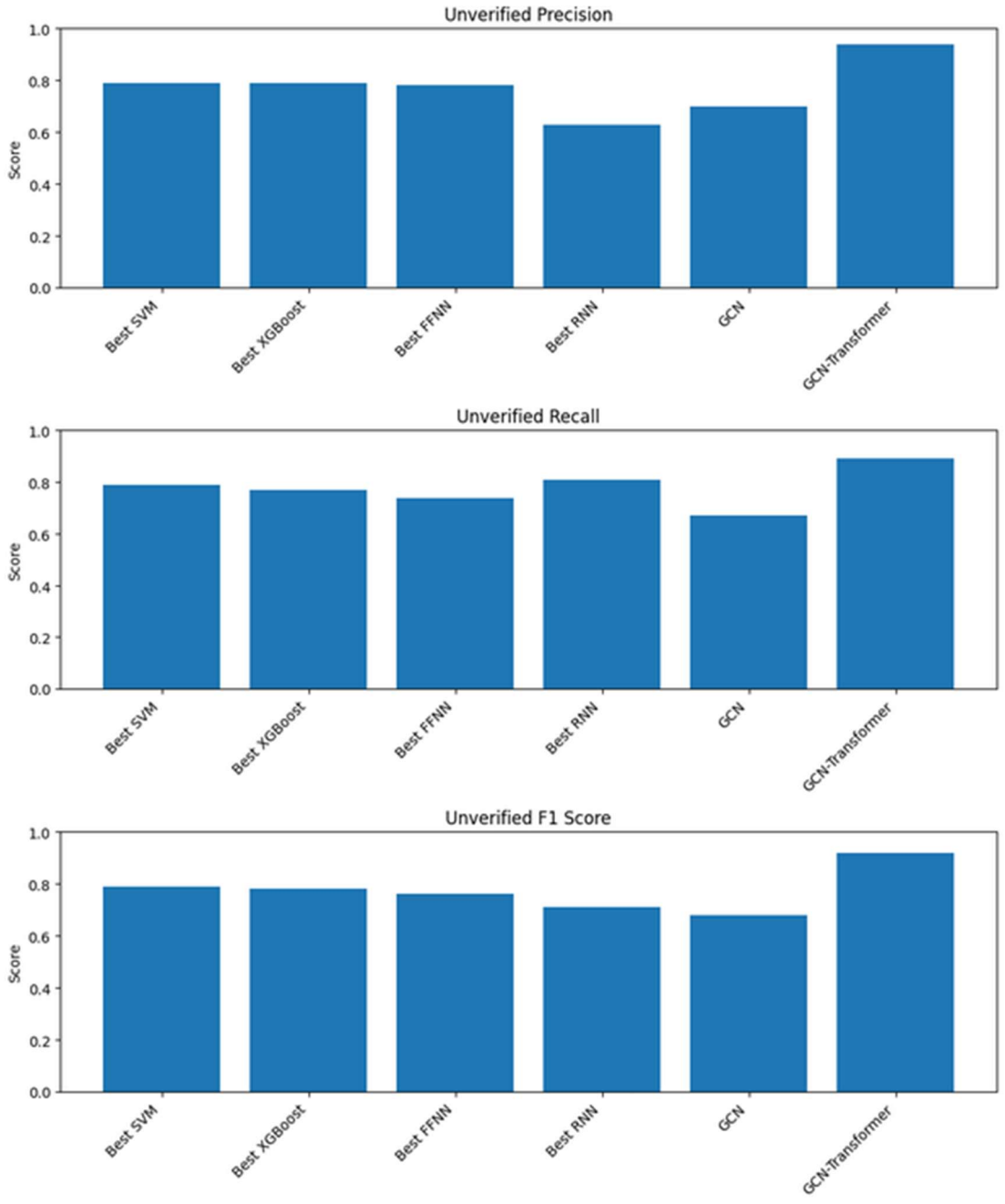



*Figure 34: Unverified Label Performance Comparison*

## 5.13. Best Performance:

**GCN-Transformer:**

In this experiment, the GCN-Transformer outperformed other models showing an incredible 95% accuracy when testing while maintaining a perfect rate of recall, especially for the tweets labeled as rumors.

GCN-Transformer sums the strength of Graph Convolutional Networks and Transformers to enhance the predictive capabilities of graph-structured data. GCNs excel in capturing local neighborhood information by aggregating features from connected nodes, while Transformers perform a global attention mechanism that enables the model to identify important relationships beyond the close connection. By integrating Transformer layers, the model can assess the importance of different nodes, even if they are not directly connected. This allows it to recognize meaningful patterns and relationships across the entire network, rather than just within close neighborhoods. As a result, the model gains a deeper and more complete understanding of the graph's structure. This combined approach enhances performance in tasks like identifying node characteristics or predicting connections, as it considers both close relationships and broader patterns across the graph, leading to more precise and reliable results.(Hoang, Pham and Ta, 2024).

**Understanding Fake News and Rumor Spreading:**

In the context of Fake News and Rumour spreading, information spreads in different ways. Some users share content within a small group (e.g., friends or family), while others aim for a much larger audience, making the misinformation go viral. Detecting harmful information requires an understanding of local and global spreading patterns:

**Local Spread (GCN's Role):** Imagine a small group of friends discussing news. If many people within the group share the same false story, it indicates that misinformation is circulating within that community. The GCN component of the model analyzes these close connections, understanding how false information spreads within small clusters.

**Global Influence (Transformer's Role):** Some misinformation extends beyond small groups, spreading across different communities through influential users, trending topics, or viral sharing. The Transformer component captures these broader connections, identifying how misinformation jumps from one group to another and spreads on a larger scale.

By combining both aspects, the model can capture how it spreads in a much more accurate way. It's not only looking at who is spreading but also how far it can reach. This way it can predict patterns of misinformation spread from small groups to something more professional.

## 5.14. Limitations of the Standard GCN Model:

Despite the relationship advantage of a Graph Neural Network, our two-layer GCN model did not outperform traditional machine learning models such as SVM and XGBoost. This aligns with the findings from (1912.09893v3), which emphasize that graph-based models do not always surpass feature-based models, especially when node attributes alone provide most of the predictive power. The study suggests that the effectiveness of GNNs depends on the extent to which the graph structure contributes meaningful information.

**Mixing Different Types of Nodes (Homogeneous Graph**): Forcing the Graph to be Homogenous came with a high price, the model treated tweet and user nodes as the same data type, using padding methods to make it fit into a uniform structure. However, this approach weakened the

unique characteristics of users, making their information less useful for distinguishing credible sources from misinformation spreaders.

**Over-Smoothing in Deeper Layers:** As the model processed more information, it spread data too evenly across the network. This caused tweets and users to become too similar, making it difficult to tell the difference between real and false information. The loss of unique details made it harder to accurately classify misinformation.

**Static Edge Weights:** The model treated all connections between tweets and users as equally important, which made it harder to identify the key sources spreading false information. Even though connection strengths were calculated based on similarity, they remained fixed throughout training. This meant the model could not adapt as misinformation spread, limiting its ability to track how false content moved through the network.

**How the Transformer Layer benefits our model:**

To improve the performance of the model, a Transformer layer was added to enhance how the graph was represented, addressing the previous issues. Here's how the Transformer layer helped fix those problems:

GCN-Transformer preserves the distinction between tweets and users, allowing the model to retain critical contextual information that GCN didn't. Even though tweets and users are technically the same node type, the Transformer layer learns to focus on key features that separate them.

Instead of spreading information evenly, the Transformer layer applies an attention mechanism that assigns different importance levels to different relationships, ensuring that key misinformation spreaders and credible sources do not lose their unique influence.

GCN-Transformer learns edge importance dynamically, adjusting weights based on context and evolving misinformation patterns. This allows the model to separate influential sources of information from casual ones, even with a homogenous graph structure.

The GCN-Transformer model significantly improved misinformation detection by combining the local aggregation power of GCNs with the global attention capabilities of Transformers. While GCN alone struggled with homogeneous graph representation, over-smoothing, and static edge weights, the Transformer layer helped overcome these limitations by preserving contextual information, selectively attending to important relationships, and dynamically adjusting edge weights.

This hybrid approach outperformed traditional machine learning models and provided a more accurate, scalable solution for identifying and tracking misinformation spread.

### 5.15. Label Prediction Insights:

**False Label (Misinformation) Classification:**

The GCN-Transformer model demonstrated the highest performance in detecting misinformation, achieving a perfect precision of 1.00 and a high recall of 0.97. In contrast, the GCN model struggled, particularly with false recall (0.64), meaning 36% of misinformation tweets were missed. This indicates issues related to over-smoothing and the limitations of using static edge weights in graph-based learning. Furthermore, traditional models like SVM and XGBoost underperformed due to their reliance solely on text and metadata, without incorporating network structures, which are crucial for understanding how misinformation propagates.

Achieving near-perfect recall in misinformation detection is critical for several reasons. Firstly, it prevents misinformation from going undetected, ensuring that false content is identified and does not continue to spread unchecked. Secondly, it strengthens trust in the model, as a system that frequently misses misinformation may allow false content to circulate further. While high recall can sometimes lead to false positives, it guarantees that the model prioritizes minimizing the spread of misinformation. Lastly, it enhances moderation efforts on social media platforms, ensuring that harmful or misleading content is flagged appropriately, thereby reducing its influence on users.

**Non-Rumour Label Classification:**

The GCN-Transformer model demonstrated the highest performance, maintaining high precision (0.96) and recall (0.95), making it the most effective at detecting misinformation. In comparison, SVM and XGBoost performed well, achieving recall scores between 83% and 90%, indicating that traditional machine learning models can effectively identify non-rumors. However, the GCN model showed slightly lower recall (0.84), suggesting some misclassifications of real content as misinformation, highlighting its limitations in distinguishing between truthful and false content.

**True-Label Classification:**

The GCN-Transformer model achieved the highest performance, with 98% precision and 93% recall, making it the most reliable in detecting misinformation. Traditional machine learning models, such as SVM and XGBoost, also performed well, maintaining precision and recall above 86%, demonstrating their effectiveness in classification tasks. However, the GCN model lagged slightly, with 76% precision and 91% recall, indicating that it misclassified more real content as misinformation compared to the other models.

**Unverified Label Classification:**

This category proved to be the most challenging for all models. The GCN-Transformer achieved the highest performance, with 0.94 precision and 0.89 recall, but still showed room for improvement. In contrast, the GCN model struggled the most, with 0.70 precision and 0.67 recall, highlighting difficulties in distinguishing uncertain content from other categories.

Across all models, the Unverified label consistently exhibited lower recall, indicating challenges in accurately differentiating it from other classes. This could be attributed to overlapping features or ambiguous patterns within the dataset. Additionally, Unverified content may share similarities with other labels, making classification more difficult and leading to more misclassifications. The Challenge with True and Non-Rumour Labels: A Risk of Incorrectly Flagging?

The GCN-Transformer model, while achieving almost a perfect recall for False (misinformation) labels, showed slightly lower recall for True (93%) and Non-Rumor (95%) content. Although these figures are strong, even minor inaccuracies in these categories can lead to significant consequences:

- **False Positive Risks**: If real, verified content is misclassified as misinformation, it could lead to the unjust suppression of accurate information.
- **Censorship Issues**: A moderation system that incorrectly flags legitimate news, expert opinions, or fact-based discussions might inadvertently silence important conversations.
- **Trust & Transparency**: Excessive misinformation detection could damage public trust, leading users to doubt the fairness of content moderation

# 6. Discussion

## 6.1. Insights

The results demonstrate that the GCN-Transformer model outperformed all other models in identifying both malicious and truthful tweets, including non-rumors. This success is largely due to its ability to capture both local relationships through GCN and global dependencies using the Transformer, leveraging their combined strengths. The model achieved a near-perfect recall of 97% for false labels, ensuring that misinformation is effectively detected and minimizing the risk of its spread. Additionally, it maintained strong and balanced evaluation metrics across all labels, making it a more reliable classifier compared to the other approaches tested in this study.

## 6.2. Key Observations

Traditional machine learning models (SVM, XGBoost) performed well for real content but struggled with misinformation detection:

- Their reliance on textual and metadata-based features limited their ability to distinguish between fake and real news.
- Lower recall for false labels (66%-76%) suggests that misinformation could still slip through.

Deep learning models (FFNN, RNN) had the weakest performance:

- Their inability to capture relational information between tweets and users led to a low recall for false labels (60%).
- This suggests that models based solely on textual patterns are not sufficient for misinformation detection.

GCN improved classification over traditional Deep Learning models but still struggled with misinformation detection:

- Over-smoothing in deeper layers caused false labels to blend with real ones, leading to misclassification issues.
- Static edge weights prevented the model from adapting to changing misinformation patterns.

GCN-Transformer effectively overcame GCN's limitations:

- Dynamic edge weighting improved adaptability, allowing the model to track how misinformation evolves in real-time.
- Selective attention mechanisms prevented over-smoothing, ensuring that misinformation spreaders remained distinguishable.
- Improved performance across all labels, achieving the best balance between precision and recall.

## 6.3. Limitations & Implications

Despite its strong performance, the GCN-Transformer model faces challenges that must be addressed for broader adoption.

**Dataset Scarcity & Expensive APIs**

Access to high-quality misinformation datasets is increasingly restricted due to API limitations and rising costs. Many studies rely on outdated datasets that may not reflect current misinformation trends, making real-time evaluation difficult. Additionally, labeling misinformation requires extensive human verification, which is resource-intensive and introduces potential biases.

**Computational Costs & Hardware Constraints**

The GCN-Transformer model is significantly more computationally demanding than traditional ML models, making real-time detection difficult. Transformer-based architectures require high memory bandwidth, making them impractical for large-scale misinformation networks without specialized hardware. Furthermore, deep learning frameworks are primarily optimized for NVIDIA's CUDA, with limited ROCm (AMD) support in key libraries like PyTorch Geometric (PyG) and Deep Graph Library (DGL). This restricts accessibility for researchers without NVIDIA hardware, increasing costs and reducing inclusivity in large-scale graph learning.

**Implications for Misinformation Detection**

This study highlights that integrating user interactions and propagation patterns into misinformation detection provides a significant advantage over traditional models relying solely on text and metadata. Graph-based approaches like GCN-Transformer analyze the spread of misinformation rather than treating individual tweets in isolation, allowing for more accurate classification.

One of the key advancements in this approach is overcoming the limitations of standard Graph Convolutional Networks (GCNs). Traditional GCNs suffer from over-smoothing, where deep layers distribute information too uniformly, making it harder to distinguish between misinformation and real content. Additionally, static edge weights treat all connections equally, failing to adapt to the evolving nature of misinformation. By incorporating Transformer layers, the model dynamically adjusts attention weights, identifying key misinformation spreaders and tracking their influence over time.

These insights have real-world applications for social media platforms like Twitter, Facebook, and Reddit, where misinformation spreads rapidly. The ability to analyze both small-scale interactions (close networks) and large-scale trends (viral misinformation campaigns) makes this approach valuable for fact-checking organizations, content moderation, and early detection systems.

## 6.4. Gaps & Future Work

Future improvements to the model could focus on incorporating heterogeneous graph neural networks (HGNNs) to better differentiate between users and tweets. The current approach treats all nodes as the same type, which may limit its ability to distinguish different roles in misinformation spread. HGNNs assign distinct properties to different node types, improving the model's understanding of how misinformation moves between users and tweets. Additionally, hyperparameter tuning could enhance performance, as the current model was trained with fixed parameters. Exploring newer GNN models may also strengthen misinformation detection, with promising approaches including Graph Attention Networks (GATs), which use adaptive attention weights similar to Transformers but optimized for graph structures, and Graph Isomorphism Networks (GINs), which capture structural differences to differentiate real and fake news patterns. Other specialized models like Heterogeneous GNNs (HAN, R-GCN) could be particularly useful for modeling user-tweet interactions.

Another area for future work is dynamic graph learning, which would allow the model to track misinformation as it evolves instead of using a static graph where relationships between users and tweets remain unchanged. Additionally, integrating multi-modal analysis could expand the model beyond text-based detection by incorporating images, videos, and external metadata. Since misinformation is often spread through various media formats, multi-modal deep learning models that combine graph-based learning with image and video analysis could create a more comprehensive and effective misinformation detection system.

# 7. Conclusion

This research focused on improving fake news detection by combining Graph Convolutional Networks (GCN) with Transformer layers. While GCN captured structural relationships in the data, the Transformer component helped model long-range dependencies. The results showed that this hybrid approach performed better than traditional machine learning models like SVM and XGBoost, as well as deep learning models such as FFNN and RNN. Even standalone GCN models fell short in comparison.

Incorporating text, metadata, and propagation patterns, the model was able to classify misinformation more effectively. The combination of graph-based learning and attention mechanisms allowed for a more accurate understanding of how false information spreads, making it a powerful tool for detecting fake news.

While the GCN-Transformer achieved the best performance, certain challenges remain, particularly in adapting to evolving misinformation patterns. Future research can address these limitations by incorporating heterogeneous graph learning, dynamic graph updates, and multi-modal analysis, as outlined in the Future Work section

# 8. Bibliography


Adriani, R. (2019) 'The Evolution of Fake News and the Abuse of Emerging Technologies'. Available at: https://www.nytimes.com/2017/10/18/world/europe/italy-fake-news.html.

Aïmeur, E., Amri, S. and Brassard, G. (2023) 'Fake news, disinformation and misinformation in social media: a review', *Social Network Analysis and Mining*, 13(1). Available at: https://doi.org/10.1007/s13278-023-01028-5.

Al-alshaqi, M., Rawat, D.B. and Liu, C. (2024) 'Ensemble Techniques for Robust Fake News Detection: Integrating Transformers, Natural Language Processing, and Machine Learning', *Sensors*, 24(18). Available at: https://doi.org/10.3390/s24186062.

Alarfaj, F.K. and Khan, J.A. (2023) 'Deep Dive into Fake News Detection: Feature-Centric Classification with Ensemble and Deep Learning Methods', *Algorithms*, 16(11). Available at: https://doi.org/10.3390/a16110507.

Allcott, H. and Gentzkow, M. (2017) 'Social media and fake news in the 2016 election', *Journal of Economic Perspectives*. American Economic Association, pp. 211–236. Available at: https://doi.org/10.1257/jep.31.2.211.

Baldo, F. *et al.* (2023) 'Adaptive Fast XGBoost for Multiclass Classification', *Journal of Information and Data Management*, 14(1). Available at: https://doi.org/10.5753/jidm.2023.3150.

Bashir, A. (2022) 'SVM Python - Easy Implementation Of SVM Algorithm 2023'. Available at: https://hands-on.cloud/svm-python-tutorial/.

Biswal, A. (2023) 'Recurrent Neural Network Tutorial'. Available at: https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn.

Devlin, J. *et al.* (2018) 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding'. Available at: http://arxiv.org/abs/1810.04805.

Dhawan, M. *et al.* (2022) 'GAME-ON: Graph Attention Network based Multimodal Fusion for Fake News Detection'. Available at: http://arxiv.org/abs/2202.12478.

Dwivedi, V.P. *et al.* (2021) 'Graph Neural Networks with Learnable Structural and Positional Representations'. Available at: http://arxiv.org/abs/2110.07875.

Esuli, A. and Sebastiani, F. (2009) *Training Data Cleaning for Text Classification*.

G. R. Nair *et al.* (2023) *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE 2023) : Antwerp, Belgium 17-19 April 2023*. IEEE.

Gao, Y. (2023) 'Federated XGBoost with bagging aggregation'. Available at: https://flower.ai/blog/2023-11-29-federated-xgboost-with-bagging-aggregation/.

GeeksforGeeks (2024) 'Feedforward neural network'. Available at: https://www.geeksforgeeks.org/feedforward-neural-network/.

H2O.ai (no date) 'What is a Confusion Matrix how are they Used?' Available at: https://h2o.ai/wiki/confusion-matrix/.

Han, Y., Karunasekera, S. and Leckie, C. (2020) 'Graph Neural Networks with Continual Learning for Fake News Detection from Social Media'. Available at: http://arxiv.org/abs/2007.03316.

Harary, F. (1962) *The Determinant of the Adjacency Matrix of a Graph*, *SIAM Review*. Available at: http://links.jstor.org/sici?sici=0036-1445%28196207%294%3A3%3C202%3ATDOTAM%3E2.0.CO%3B2-C.

Hoang, T.L., Pham, T.D. and Ta, V.C. (2024) 'Improving Graph Convolutional Networks with Transformer Layer in social-based items recommendation'. Available at: https://doi.org/10.1109/KSE53942.2021.9648823.

Holcomb, J., Gottfried, J. and Mitchell, A. (2013) 'News Use Across Social Media Platforms'. Available at: https://www.pewresearch.org/journalism/2013/11/14/news-use-across-social-media-platforms/.

Khanam, Z. *et al.* (2021) 'Fake News Detection Using Machine Learning Approaches', *IOP Conference Series: Materials Science and Engineering*, 1099(1), p. 012040. Available at: https://doi.org/10.1088/1757-899x/1099/1/012040.

Kipf, T. and Welling, M. (2016) 'How powerful are Graph Convolutional Networks?' Available at: https://tkipf.github.io/graph-convolutional-networks/.

Kipf, T.N. and Welling, M. (2016) 'Semi-Supervised Classification with Graph Convolutional Networks'. Available at: http://arxiv.org/abs/1609.02907.

Kipf, T.N. and Welling, M. (2017) *SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS*.

Kumar, S. *et al.* (2020) 'Fake news detection using deep learning models: A novel approach', *Transactions on Emerging Telecommunications Technologies*, 31(2). Available at: https://doi.org/10.1002/ett.3767.

Lee, D.H. *et al.* (2019) 'Fake news detection using deep learning', *Journal of Information Processing Systems*, 15(5), pp. 1119–1130. Available at: https://doi.org/10.3745/JIPS.04.0142.

Li, X. *et al.* (2021) 'Exploring Text-transformers in AAAI 2021 Shared Task: COVID-19 Fake News Detection in English'. Available at: http://arxiv.org/abs/2101.02359.

Liu, P., Qiu, X. and Huang, X. (2016) 'Recurrent Neural Network for Text Classification with Multi-Task Learning'. Available at: http://arxiv.org/abs/1605.05101.

M, H. and M.N, S. (2015) 'A Review on Evaluation Metrics for Data Classification Evaluations', *International Journal of Data Mining & Knowledge Management Process*, 5(2), pp. 01–11. Available at: https://doi.org/10.5121/ijdkp.2015.5201.

Maekawa, S. *et al.* (2019) *General Generator for Attributed Graphs with Community Structure*. Available at: https://linqs.soe.ucsc.edu.

McBride, M. (2023) 'GraphicMaths - Adjacency matrices'. Available at: https://graphicmaths.com/computer-science/graph-theory/adjacency-matrices/.

Murat H. Sazli (2006) 'A Brief Review of Feed-Forward Neural Networks'.

Pardamean, A. and Pardede, H.F. (2021) 'Tuned bidirectional encoder representations from transformers for fake news detection', *Indonesian Journal of Electrical Engineering and Computer Science*, 22(3), pp. 1667–1671. Available at: https://doi.org/10.11591/ijeecs.v22.i3.pp1667-1671.

Parikh, S.B. and Atrey, P.K. (2018) 'Media-Rich Fake News Detection: A Survey', in *Proceedings - IEEE 1st Conference on Multimedia Information Processing and Retrieval, MIPR 2018*. Institute of

Electrical and Electronics Engineers Inc., pp. 436–441. Available at: https://doi.org/10.1109/MIPR.2018.00093.

Pennycook, G. and Rand, D.G. (2021) 'The Psychology of Fake News', *Trends in Cognitive Sciences*. Elsevier Ltd, pp. 388–402. Available at: https://doi.org/10.1016/j.tics.2021.02.007.

Phan, H.T., Nguyen, N.T. and Hwang, D. (2023) 'Fake news detection: A survey of graph neural network methods', *Applied Soft Computing*. Elsevier Ltd. Available at: https://doi.org/10.1016/j.asoc.2023.110235.

scikit-learn (2018) 'sklearn.metrics.classification_report — scikit-learn 0.20.3 documentation'. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html.

Segura-Bedmar, I. and Alonso-Bartolome, S. (2022) 'Multimodal Fake News Detection', *Information (Switzerland)*, 13(6). Available at: https://doi.org/10.3390/info13060284.

Sharma, U., Saran, S. and Patil, S.M. (2021) *Fake News Detection using Machine Learning Algorithms*. Available at: www.ijert.org.

Shu, K. *et al.* (2019) 'Defend: Explainable fake news detection', in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, pp. 395–405. Available at: https://doi.org/10.1145/3292500.3330935.

Slovikovskaya, V. (2019) 'Transfer Learning from Transformers to Fake News Challenge Stance Detection (FNC-1) Task'. Available at: http://arxiv.org/abs/1910.14353.

Smutek, T. *et al.* (2024) *A Graph-Based Recommendation System Leveraging Cosine Similarity for Enhanced Marketing Decisionspl; A Graph-Based Recommendation System Leveraging Cosine Similarity for Enhanced Marketing Decisions 84*, *European Research Studies Journal*.

Thota, A. *et al.* (2018) *Number 3 Article 10 2018 Part of the Artificial Intelligence and Robotics Commons Recommended Citation Recommended Citation Thota, Aswini; Tilak, Priyanka; Ahluwalia, Simrat; and Lohia*, *SMU Data Science Review*. Available at: https://scholar.smu.edu/datasciencereviewAvailableat:https://scholar.smu.edu/datasciencereview/vol1/iss3/10http://digitalrepository.smu.edu.

Truică, C.O., Apostol, E.S. and Karras, P. (2024) 'DANES: Deep Neural Network Ensemble Architecture for Social and Textual Context-aware Fake News Detection', *Knowledge-Based Systems*, 294. Available at: https://doi.org/10.1016/j.knosys.2024.111715.

Veličkovi´veličkovi´c, P. *et al.* (2018) *GRAPH ATTENTION NETWORKS*.

Wang, Y. *et al.* (2018) 'EANN: Event adversarial neural networks for multi-modal fake news detection', in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, pp. 849–857. Available at: https://doi.org/10.1145/3219819.3219903.

Wang, Y. *et al.* (2021) 'COVID-19 Fake News Detection Using Bidirectional Encoder Representations from Transformers Based Models'. Available at: http://arxiv.org/abs/2109.14816.

with Scikit-Learn Keras and TensorFlow, 2nd Edition (2019) 'Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition'. Available at: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/.

Wu, Y. *et al.* (2021) *Multimodal Fusion with Co-Attention Networks for Fake News Detection*.

Yang, L. *et al.* (2019) *Masked Graph Convolutional Network*.

Yang, P. *et al.* (2023) 'Multi-modal transformer for fake news detection', *Mathematical Biosciences and Engineering*, 20(8), pp. 14699–14717. Available at: https://doi.org/10.3934/mbe.2023657.

Yu, H. and Kim, S. (2012) *SVM Tutorial: Classification, Regression, and Ranking*.

Yuan, C. *et al.* (2019) 'Jointly embedding the local and global relations of heterogeneous graph for rumor detection'. Available at: http://arxiv.org/abs/1909.04465.

Yun, S., Jeong, M., Kim, R., Kang, J. and Kim, Hyunwoo J. (2019) 'Graph Transformer Networks'. Available at: http://arxiv.org/abs/1911.06455.

Zaman, S. *et al.* (2022) *Parallelizing Graph Neural Networks via Matrix Compaction for Edge-Conditioned Networks*.

Zhou, X. and Zafarani, R. (2018) 'A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities'. Available at: https://doi.org/10.1145/3395046.