INTRODUCTION TO NLP (CS7.401, SPRING 2025)

SARCASM DETECTION AND REWRITING

Team 38 - The SUS

Umang Patel (2022101037)
Sparsh Goel (2022101051)
Sahil Patel (2022101046)

International Institute of Information Technology, Hyderabad

April 2025

Contents

1 Introduction

Sarcasm is a common trend on social media; however, the inherent irony involved in such comments usually misleads sentiment analysis algorithms. The purpose of this project is to design a two-stage system which not only detects sarcastic comments but also paraphrases them into direct, non-sarcastic statements. Our approach utilizes state-of-the-art transformer models coupled with graph-based methods to effectively maintain contextual semantics along with necessary syntactic and affective markers. We further extend our work to include a rewriting module, motivated by recent work in text attribute transfer, that methodically deletes and replaces critical sentiment-carrying words, essentially converting sarcastic text into explicit text.

2 Project Objectives

- Develop a robust system that accurately detects sarcasm in social media text and transforms it into plain language.
- Implement a hybrid sarcasm detection module that leverages both contextual embeddings and graph-based representations.
- Analyze word-level contributions to sarcasm for interpretability and to guide the rewriting process.
- Develop a rewriting module that removes the most influential sarcastic markers and replaces them with neutral or positive alternatives.
- Compare our approach with existing methods and demonstrate the improvements introduced by our modifications.

3 Literature Review

Our literature review focused on two primary areas: sarcasm detection and text rewriting.

Sarcasm Detection

We examined the paper *"Sarcasm Detection Using Bidirectional Encoder Representations from Transformers and Graph Convolutional Networks"* [1] which introduces a hybrid model that combines transformer-based embeddings with a graph convolutional network (GCN) to capture both semantic and syntactic features. This work provided the foundational ideas for our detection module. Additionally, we reviewed comparative studies

such as *"Sarcasm Detection: A Comparative Study"* [2] and *"Sarcasm Detection in Online Comments Using Machine Learning"* [3], which offered insights into various machine learning techniques and their effectiveness, so it helped in our choice of models and evaluation metrics.

Text Attribute Transfer and Rewriting

For the rewriting component, we reviewed *"Delete, Retrieve, Generate: A Simple Approach to Sentiment and Style Transfer"* [4]. This paper demonstrates that effective attribute transfer can be achieved by identifying and removing key attribute markers and then generating new text with the target attribute. While this approach provided valuable insights about identifying and manipulating attribute markers, we ultimately determined that a GAN-based architecture would better address the complexities of sarcasm transformation. The delete-retrieve-generate concepts influenced our understanding of how sarcastic elements function in text, which lead us to use adversarial training approach.

4 Methodology

4.1 Data Preprocessing and Feature Engineering

Our preprocessing pipeline standardizes text data from various sources by:

- **Normalization:** Tokenization, lowercasing, punctuation removal, and lemmatization (using spaCy and NLTK).
- **Feature Extraction:**
  - **Semantic Features:** 300-dimensional GloVe embeddings (from the glove-wiki-gigaword-300 dataset) are used to capture word meanings.
  - **Affective Features:** A 5-dimensional sentiment vector is extracted for each word using SenticNet, providing polarity, sentiment flags, and intensity.

Datasets: For training, we have combined the Sarcasm on Reddit dataset [5] and the Mustard dataset [6]. The Sarcasm on Reddit dataset contains 1.3 million comments, while the Mustard dataset provides a rich set of sarcastic comments from popular TV shows such as *Friends*, *The Golden Girls*, and *The Big Bang Theory*. This combination ensures a diverse training set that captures various forms of sarcasm.

4.2 Graph Construction and Node Features

Our approach to graph construction integrates multiple sources of information:

- **Node Creation:** Nodes are derived from words in the text using spaCy's dependency parser.
- **Edge Formation:** Two types of edges are created:
  - **Window-based Edges:** Capture local context among adjacent words.
  - **Dependency-based Edges:** Reflect syntactic relationships from the dependency tree.
- **Feature Integration:** Each node's feature is the concatenation of its 300-dimensional GloVe embedding and a 5-dimensional SenticNet-derived sentiment vector. This integration embeds both semantic and affective cues directly into the graph.

4.3 Graph Processing and Detection Model

Our sarcasm detection model utilizes a hybrid architecture:

- **Contextual Embeddings:** A RoBERTa model extracts rich contextual embeddings from the input text.
- **Graph Convolutional Network (GCN):** A 4-layer GCN processes the integrated graph to update node representations, incorporating both syntactic and sentiment information.
- **Sequential Modeling:** An LSTM further processes the GCN output to capture sequential dependencies. For batched inputs, an averaging strategy is used.
- **Attention-Based Fusion:** An attention layer fuses the RoBERTa-derived and GCN-LSTM features, and the resulting representation is passed through fully connected layers for binary classification (sarcastic vs. non-sarcastic).

Figure 1 illustrates our hybrid architecture that integrates transformer-based and graph-based components for sarcasm detection.

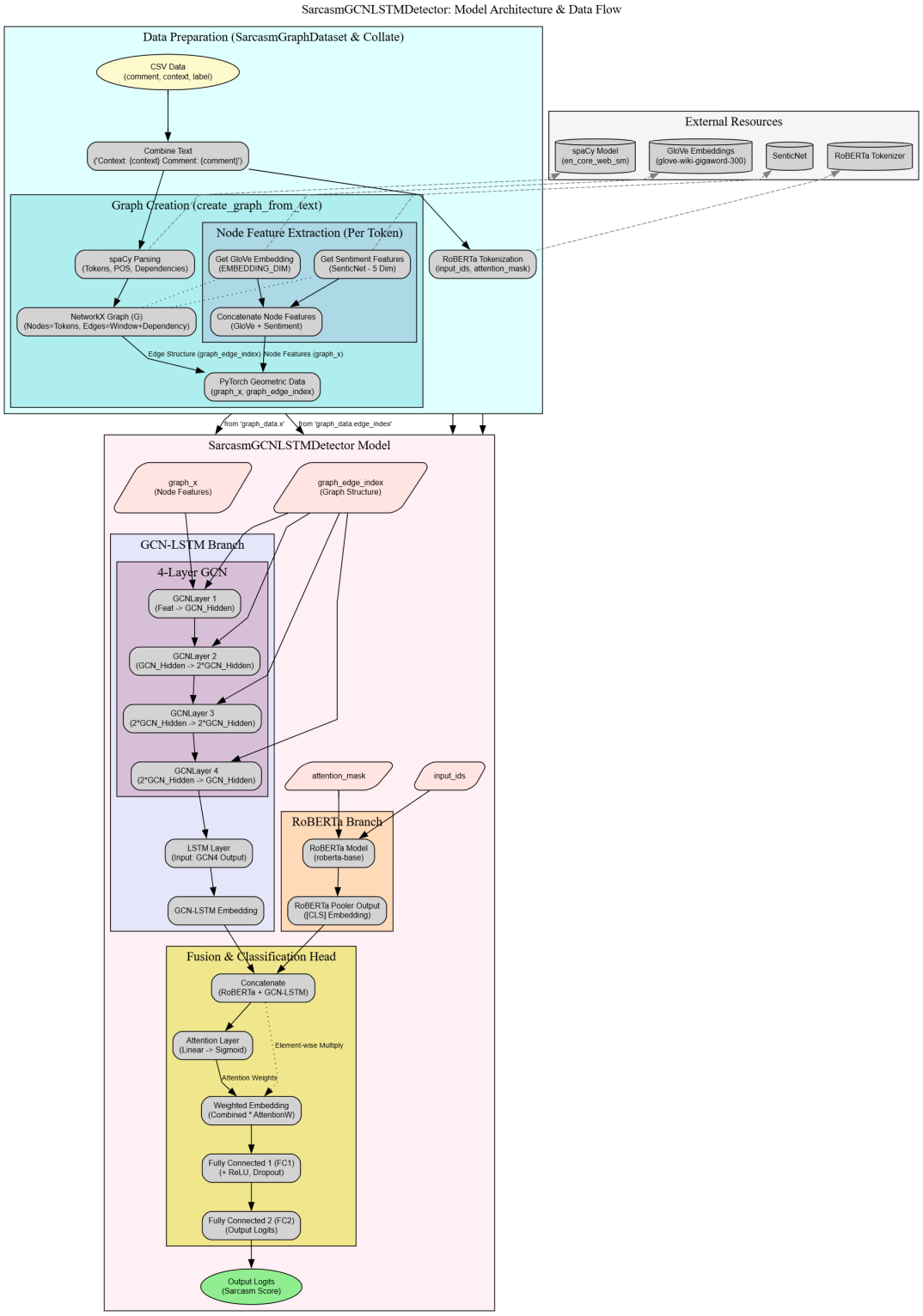


Figure 1: Architecture of hybrid sarcasm detection system.

4.4 Rewriting Module for Sarcasm Transformation

Our initial approach to sarcasm transformation was inspired by the delete-retrieve-generate paradigm, which suggests identifying and removing key attribute markers before generating new text. However, during experimentation, we discovered that sarcasm often depends on subtle contextual relationships that are difficult to transform through simple deletion and replacement. This realization led us to develop a more sophisticated GAN-based approach that could better preserve semantic content while effectively transforming the sarcastic tone.

For transforming sarcastic text into plain language, our final rewriting module implements a Generative Adversarial Network (GAN) architecture:

- **Generator Architecture:** We use a sequence-to-sequence model based on BART that transforms sarcastic text to non-sarcastic equivalents. The generator encodes the sarcastic input and decodes it into literal, straightforward language while preserving the original content.
- **Discriminator Architecture:** A dual-input BERT-based classifier receives both the source text and the generated transformation, producing a probability score indicating whether the transformation is authentic (human-like). The discriminator uses concatenated pooled embeddings from both texts to make this determination.
- **Adversarial Training Strategy:** Our approach follows the GAN paradigm with two complementary objectives:
  - The generator minimizes a combined loss function: $L_G = L_{supervised} + \lambda \cdot L_{adversarial}$, where $L_{supervised}$ is the reconstruction loss and $L_{adversarial}$ is derived from the discriminator's feedback.
  - The discriminator maximizes its ability to distinguish between real non-sarcastic rephrases and generated ones, using a standard binary cross-entropy loss: $L_D = \frac{1}{2}(BCE(D(x, y_{real}), 1) + BCE(D(x, y_{fake}), 0))$.
- **Training Method:** We implement alternating optimization, where for each batch:
  - The generator is updated to minimize its loss, with gradient accumulation over multiple steps.
  - The discriminator is updated to maximize classification accuracy between real and generated pairs.
  - Gradient clipping is applied to stabilize the adversarial training process.
- **Training Data:** For training our GAN-based rewriting module, we utilized the iSarcasm dataset [7], which contains 4,484 tweets out of which 777 are labelled

as sarcastic and 3,707 as non-sarcastic. We only need to use the sarcastic tweets for training the generator, as the discriminator can be trained on both sarcastic and non-sarcastic data. The generator is trained to transform sarcastic tweets into their non-sarcastic counterparts, while the discriminator is trained to distinguish between real non-sarcastic tweets and the generated ones.

As shown in Figure 2, our GAN architecture leverages the adversarial relationship between the generator and discriminator to produce high-quality non-sarcastic transformations.

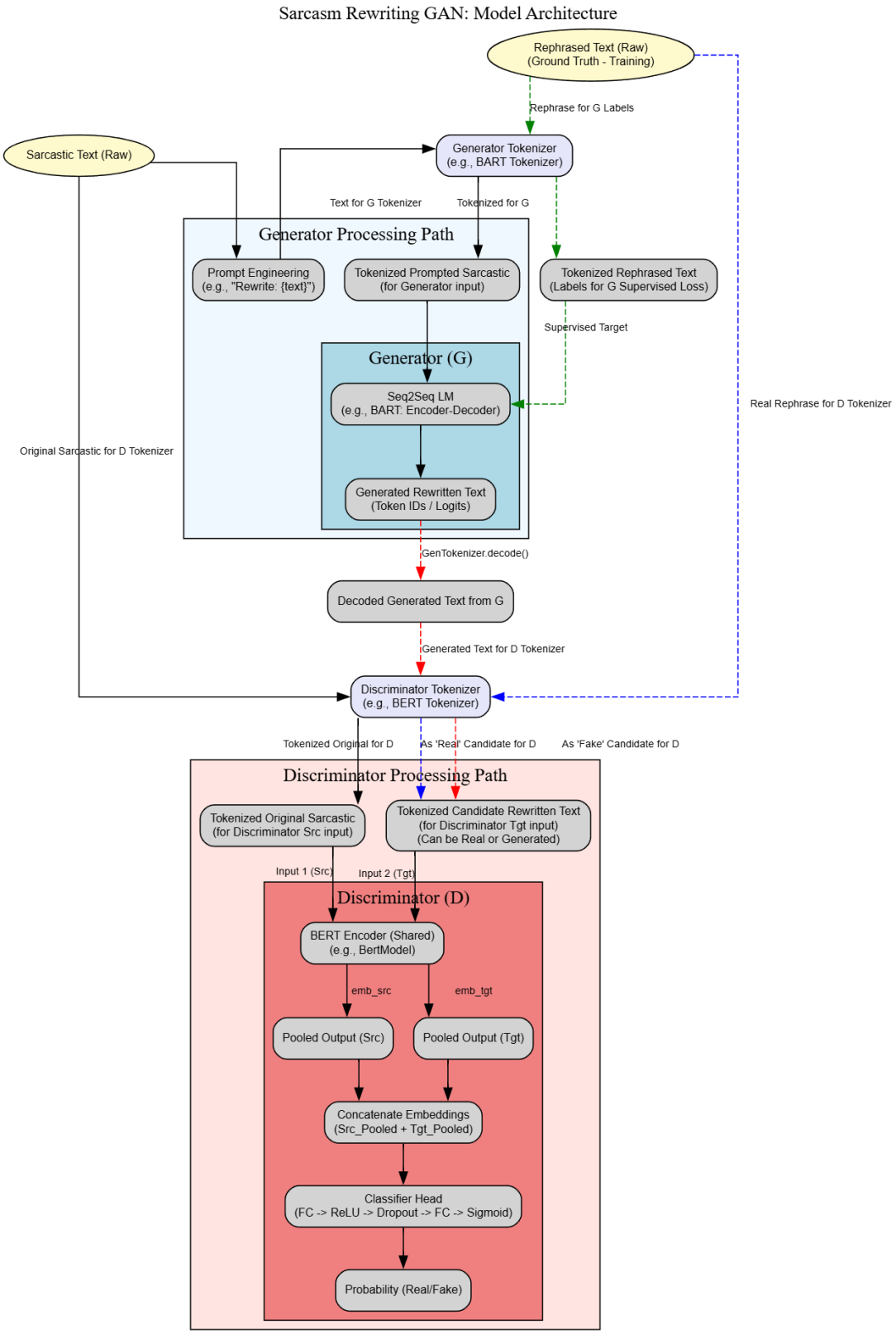


Figure 2: Architecture of GAN-based sarcasm rewriting system.

5 Novel Enhancements in Graph Construction and Processing

Our approach incorporates novel modifications over the original architecture described in the literature. In contrast to the paper—which constructs two separate graphs (a dependency graph based on syntactic dependency trees with self-loops and undirected edges, and an affective graph derived from sentiment scores computed using SenticNet, where edge weights reflect the absolute differences in sentiment scores)—our updated implementation creates a single integrated graph.

Graph Construction and Node Features

In the paper, separate graphs are constructed:

- A dependency graph captures syntactic relationships.
- An affective graph is derived from sentiment scores.

These graphs utilize 300-dimensional word embeddings combined with affective scores. In our updated code, we build a single graph where nodes are extracted using spaCy's dependency parser. Two types of edges are added:

- **Window-based edges** to capture local context among adjacent words.
- **Dependency-based edges** derived from the dependency parse.

For each node, features are computed by concatenating a 300-dimensional GloVe embedding (from the glove-wiki-gigaword-300 dataset) with a 5-dimensional sentiment feature vector extracted from SenticNet. This integration directly merges affective information into the node features, thereby simplifying the architecture while enhancing the representation.

Graph Processing Module

The original paper employs a 4-layer Graph Convolutional Network (GCN) that alternates between processing inputs from the separate dependency and affective graphs before feeding the updated node representations to a softmax classifier. In our approach, we implement a 4-layer GCN using PyTorch Geometric's `GCNConv` layers on the single integrated graph. Additionally, we introduce the following modifications:

- We incorporate an LSTM module to capture sequential dependencies. For single-sample processing, the node features are passed through the LSTM, while for batched graphs we employ a mean aggregation strategy.

- An attention mechanism is used to fuse the RoBERTa-derived embedding with the output of the GCN-LSTM branch. Unlike the paper, which directly applies a softmax classifier on the combined representation, our attention layer dynamically weighs the contributions from different feature sources before final classification.

6 Training and Evaluation

6.1 Training Process

- **Dataset Preparation:** The combined Sarcasm on Reddit and Mustard datasets are balanced to ensure equal representation of sarcastic and non-sarcastic samples.
- **Optimization:** The model is trained using the AdamW optimizer with gradient accumulation and mixed precision training (on GPUs) for efficiency.
- **Checkpointing:** Model checkpoints are saved based on validation performance, and evaluation metrics such as accuracy and F1-score are monitored.

6.2 Evaluation Strategy

- **Quantitative Metrics:** The detection module is evaluated using accuracy, F1-score, and confusion matrices on a held-out test set.
- **Interpretability:** A word removal analysis quantifies the impact of individual words on the sarcasm prediction, providing insights to guide the rewriting module.

7 Performance Metrics

7.1 Sarcasm Detection

- **Training:** The model was trained for 4 epochs.
- **Validation Performance:**

Epoch	Accuracy	F1-score
1	79.57%	80.31%
2	83.02%	82.44%
3	85.20%	85.04%
4	85.98%	86.15%

Table 1: Validation Performance for Novel Implementation (Version 2)

- **Test Set Results:** Accuracy = 78.10%, F1-score = 78.49%.
- **Confusion Matrix:** The confusion matrix is saved as `sarcasm_confusion_matrix.png`. (See Figure 3.)

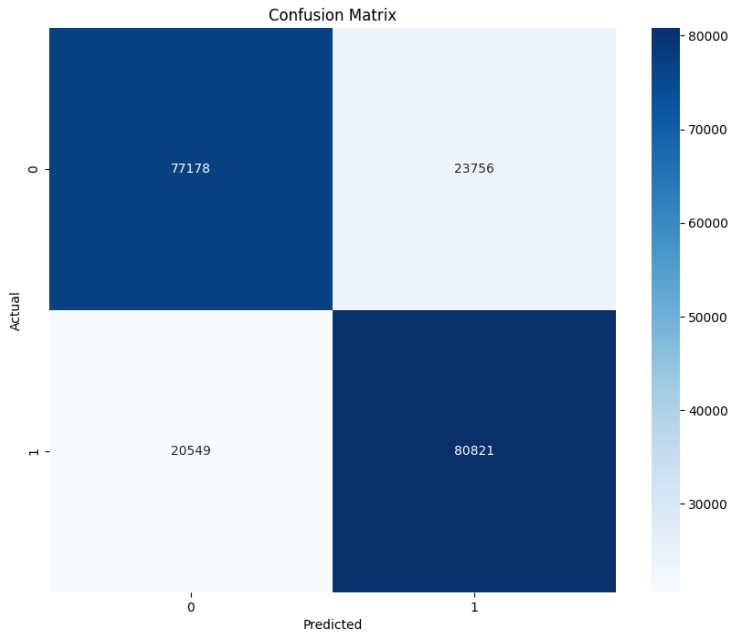


Figure 3: Confusion Matrix for Sarcasm Detection

8 Sarcasm Rewriting Results

We evaluated our GAN-based rewriting model on a some set of sarcastic statements. Table 2 shows representative examples from our test set.

Table 2: Examples of Sarcasm Rewriting Using GAN Architecture

Sarcastic Input	**Non-sarcastic Output**
Congratulations on stating the obvious. I'm sure glaciers will start moving any minute now	There is no need to stating the obvious.
Nice of you to show up three minutes late your timing really is something else	It's not nice of you to show up three minutes late.
I don't need your approval, darling, I have my own	I don't need your approval, I have my own opinion.
Sure, let's ignore all evidence and cling to your flawless reasoning	There is no need to ignore all evidence and cling to my flawless reasoning.
Absolutely, let's add that to the dozen other things I definitely wasn't planning to do.	There are a lot of things I don't like doing.
Congratulations on arriving exactly three minutes late Your punctuality is truly inspiring	Having to arrive exactly three minutes late is disappointing but not disappointing.

9 Implementation Insights and Experimental Results

Our experiments have provided several key insights:

- **Integrated Graph Approach:** Merging window-based and dependency-based edges, along with the integration of semantic and sentiment features, results in a richer text representation.
- **Hybrid Architecture Benefits:** Combining RoBERTa embeddings with a GCN-LSTM branch (enhanced via an attention mechanism) leads to competitive sarcasm detection performance.
- **Word-Level Analysis:** Detailed word removal experiments have identified the most influential words contributing to sarcasm. These insights directly inform our rewriting strategy.
- **GAN Effectiveness for Style Transfer:** Our experiments demonstrate that the adversarial approach significantly outperforms standard sequence-to-sequence models for sarcasm rewriting. The GAN architecture's key advantage lies in its ability to learn the distribution of non-sarcastic text styles rather than just mapping sar-

castic phrases to non-sarcastic alternatives. The discriminator component proved crucial in guiding the generator toward preserving semantic content while transforming stylistic aspects, achieving a balance that is difficult to maintain with supervised learning alone. We observed that the adversarial signal was particularly valuable for handling complex cases of implicit sarcasm where context interpretation is essential.

10 Word-Level and Graph-Level Analysis for Sarcasm Detection

To better understand how our model detects sarcasm, we conducted detailed analysis of word-level contributions and graph-based interactions using visualizations. We present a case study on the sarcastic statement ”Congratulations on stating the obvious. I’m sure glaciers will start moving any minute now”

10.1 Word-Level Importance Analysis

Our word pair analysis quantifies how adjacent word combinations contribute to the sarcasm classification by measuring the change in prediction probability when word pairs are removed from the text (Figure 4).

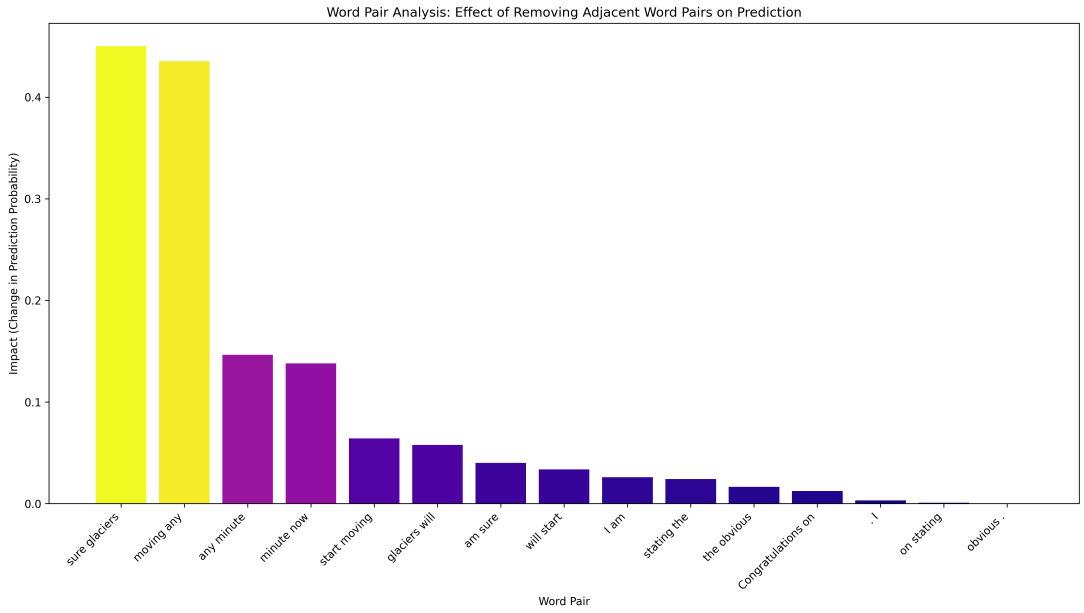


Figure 4: Word Pair Analysis: Impact of removing adjacent word pairs on prediction.

The results reveal that certain word combinations are particularly indicative of sarcasm. The pairs ”sure glaciers” and ”moving any” have the highest impact scores (approximately 0.45), while other pairs like ”any minute” also show substantial influence (around 0.15). This demonstrates that sarcasm often relies on specific contextual combinations rather than isolated words.

Figure 5 shows the most impactful words in the sentence.

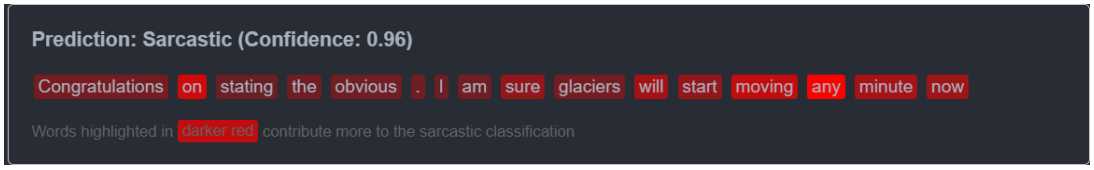


Figure 5: Highlighted Words: Most impactful words in the sentence.

The word interaction heatmap visualizes how words influence each other within the integrated graph structure (Figure 6).

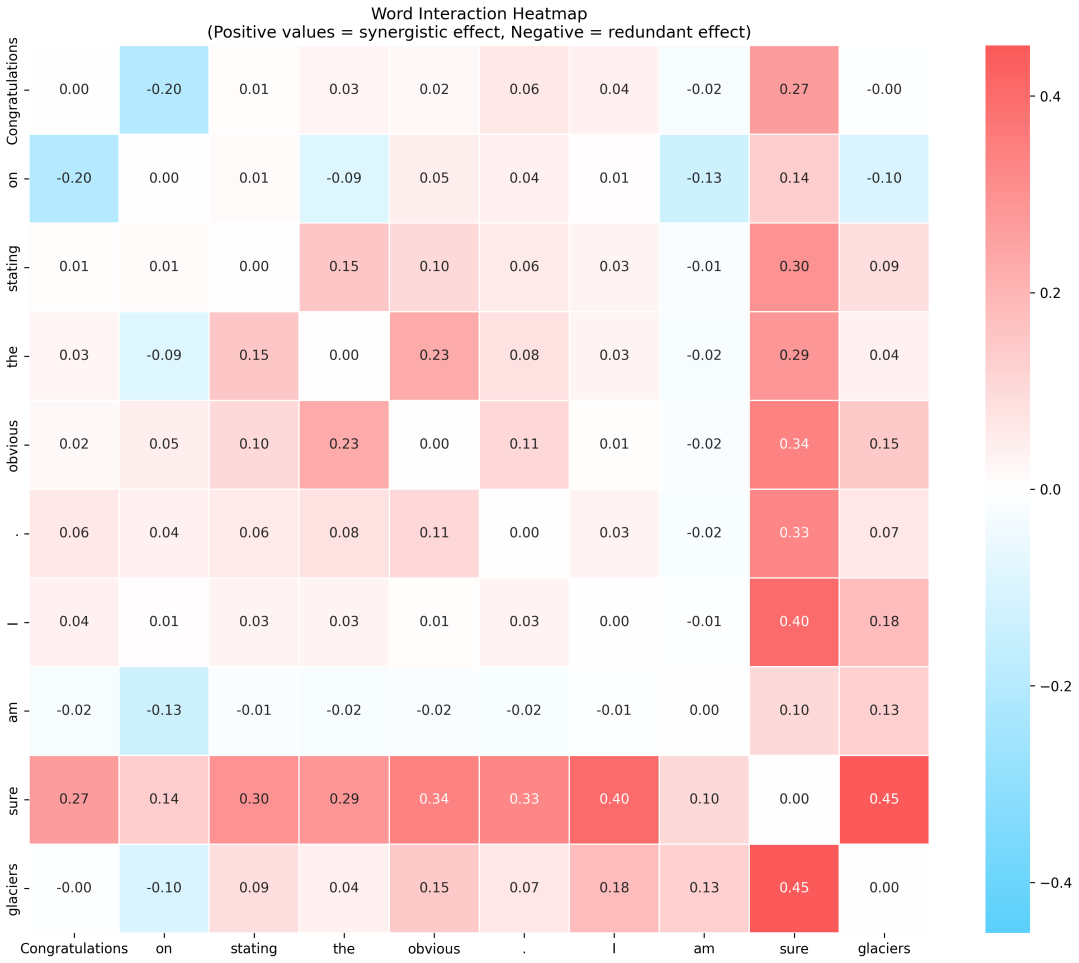


Figure 6: Word Interaction Heatmap: Positive values (red) indicate synergistic effects while negative values (blue) show redundancy.

Key observations from the heatmap include:

- Strong positive interactions (0.45) between ”sure” and ”glaciers,” suggesting that this combination is particularly indicative of sarcasm
- High synergistic effect (0.40) between ”sure” and ”I,” showing that the first-person framing strengthens the sarcastic tone
- Negative interactions (-0.20) between ”Congratulations” and ”on,” indicating redundancy
- The word ”sure” exhibits strong positive interactions with almost all other words, confirming its role as a key sarcasm marker

10.2 Attention Analysis

We examined the attention mechanisms to understand how the model focuses on different parts of the text (Figure 7).

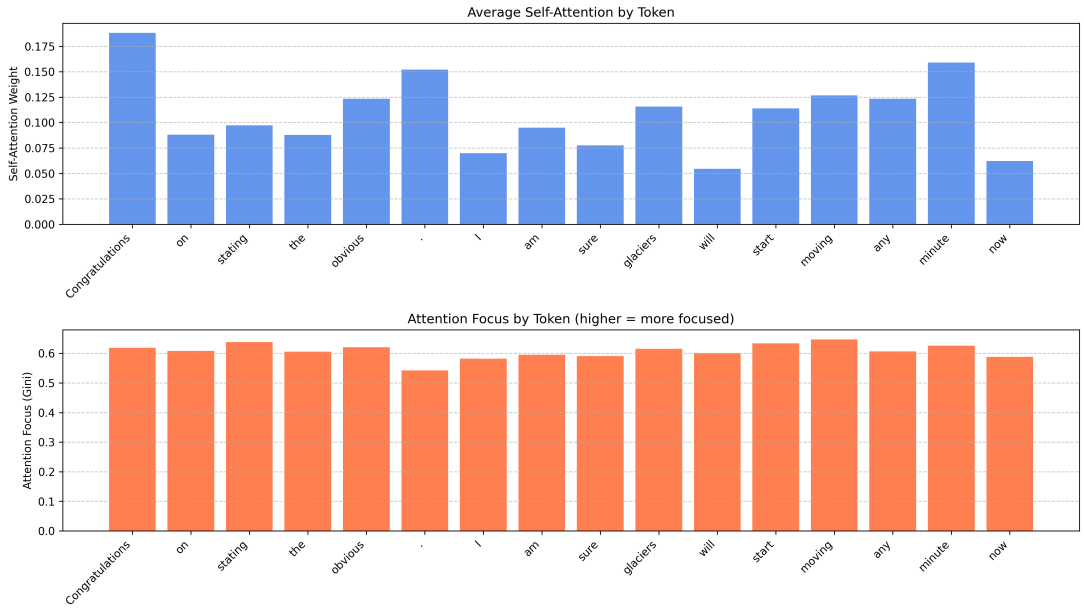


Figure 7: Aggregated Attention Weights

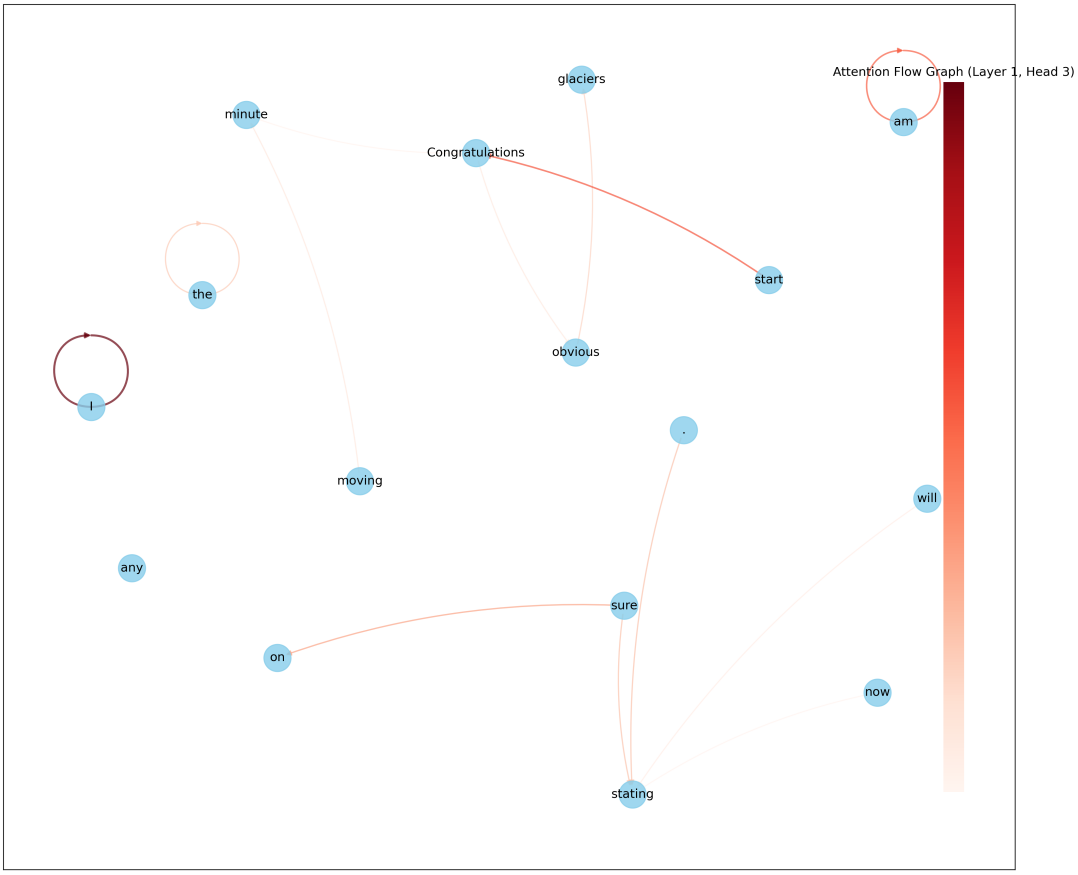


Figure 8: Attention Flow Graph: Network visualization showing attention relationships between tokens.

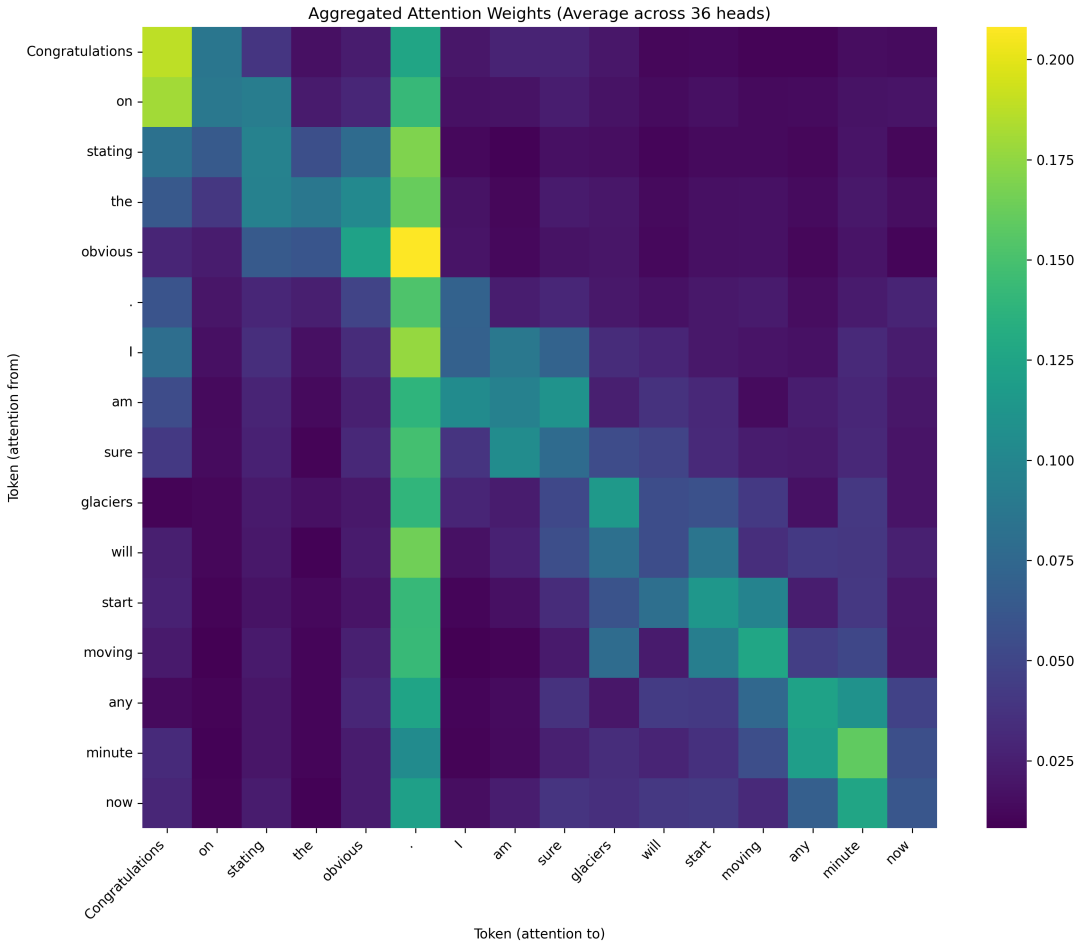


Figure 9: Self-Attention Analysis

The attention analysis reveals several patterns:

- The punctuation mark (”.”) serves as an information aggregation point with high attention from surrounding words
- ”Congratulations” receives the highest self-attention (approximately 0.18), indicating its importance in establishing the sarcastic context
- Several strong attention flows exist from ”Congratulations” to other tokens and from many tokens to the period, suggesting these elements frame the sarcastic structure
- The attention pattern reveals a hierarchical structure where certain tokens (particularly ”Congratulations”, ”I”, and ”.”) function as contextual anchors

10.3 GCN Node Importance and Evolution

We analyzed how node representations evolve through the four GCN layers (Figure 10).

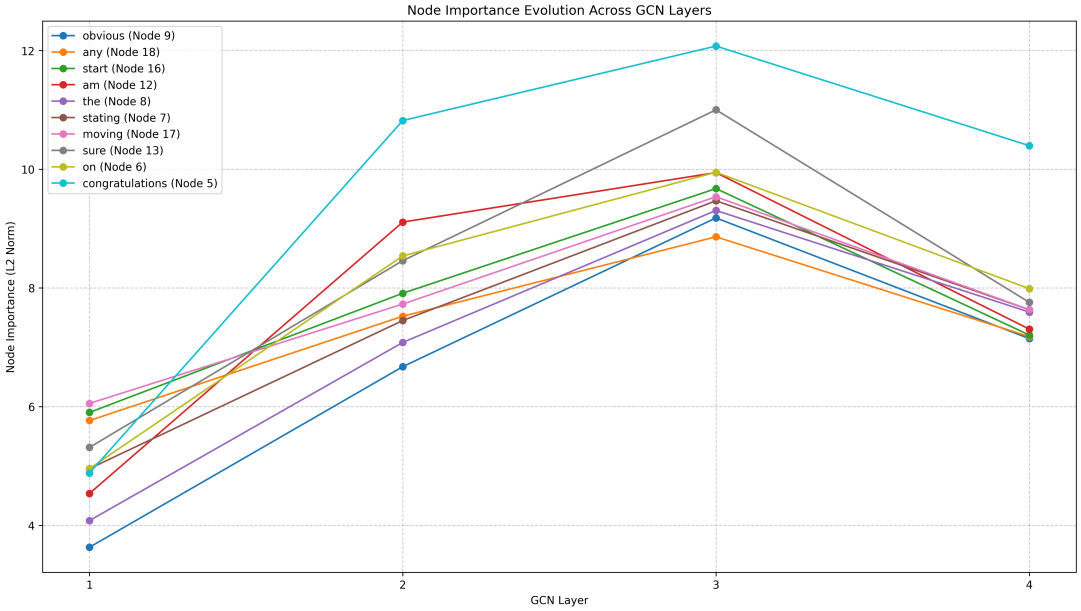


Figure 10: Node Importance Evolution Across GCN Layers: Shows how the importance of different tokens evolves through the GCN layers.

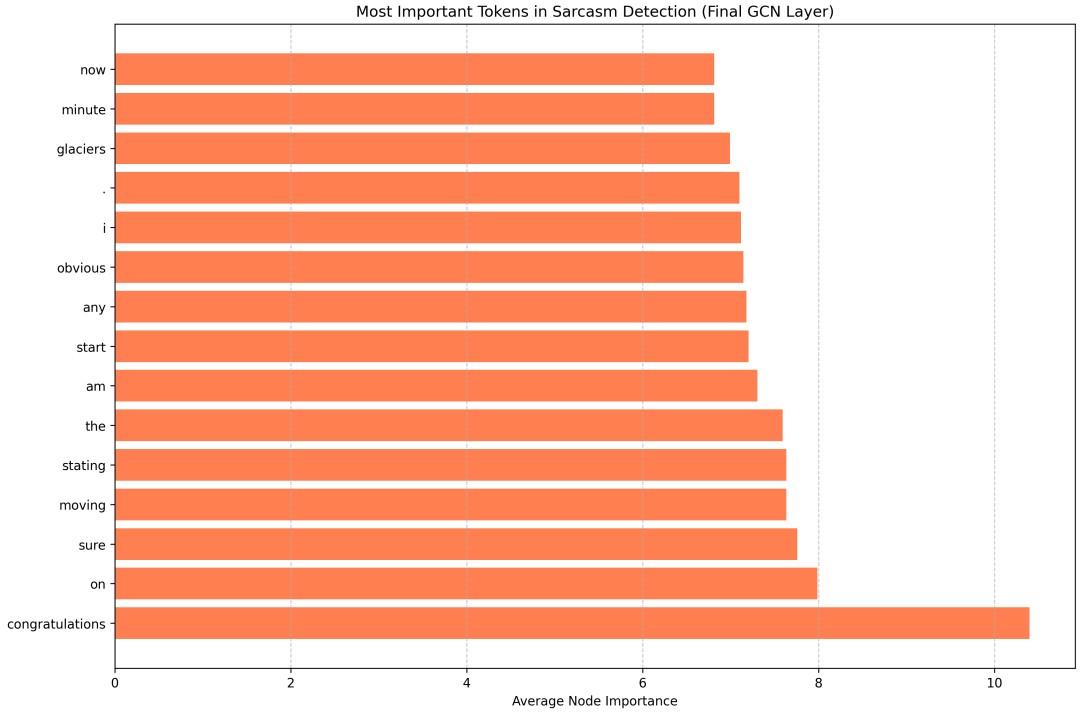


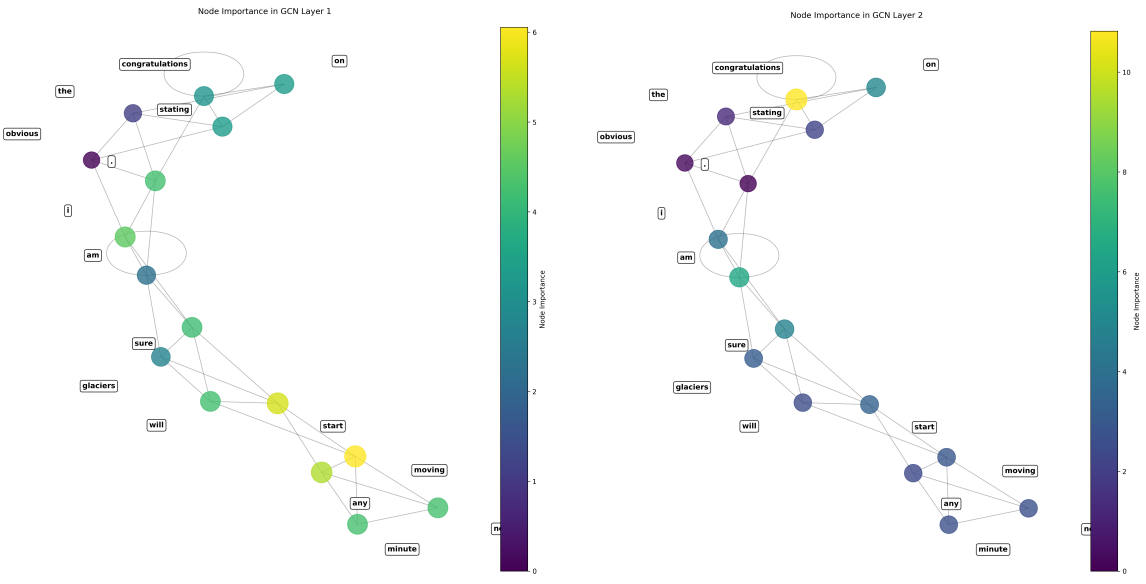
Figure 11: Most Important Tokens in Final GCN Layer: Ranking of tokens by their influence on sarcasm prediction

The GCN analysis reveals:

- The ”congratulations” node shows a distinctive pattern, starting with relatively low importance in layer 1 (around 5) but becoming the most important node by layer 3 (around 12)
- Most nodes reach peak importance in layer 3 before declining in layer 4, suggesting the third layer is critical for feature extraction
- The relative importance of different nodes shifts across layers, with ”obvious” starting very low but gaining importance, while ”am” begins with moderate importance that increases steadily

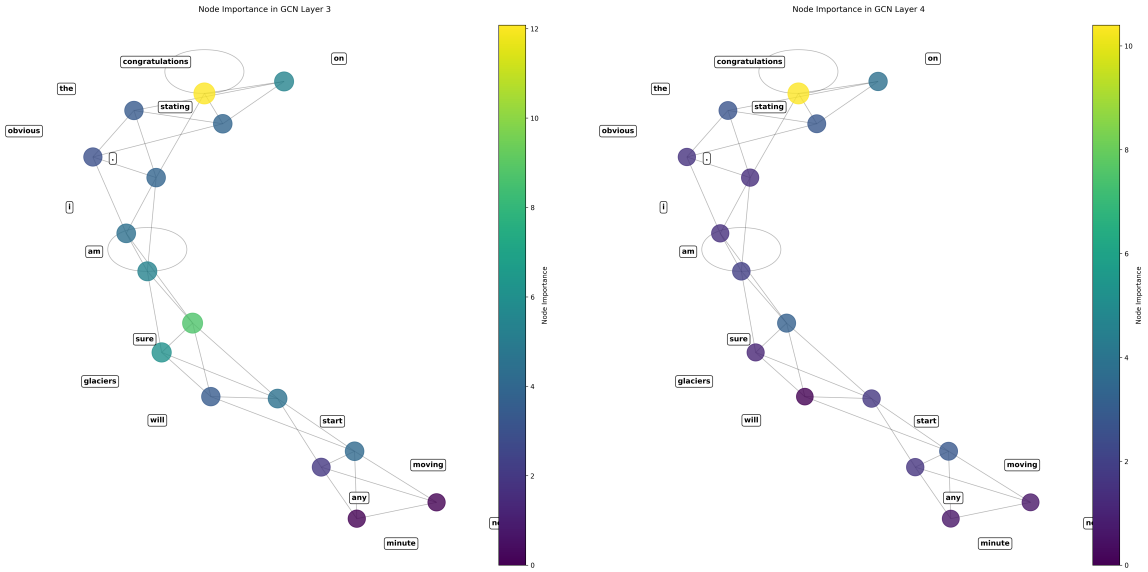
10.4 Layer-Specific GCN Analysis

To provide deeper insights into how our GCN processes information across layers, we examined node representations at each layer individually.



(a) GCN Layer 1 (b) GCN Layer 2

Figure 12: Node importance visualization across early GCN layers showing the beginning of feature learning.



(a) GCN Layer 3

(b) GCN Layer 4

Figure 13: Node importance visualization across later GCN layers where feature refinement happens.

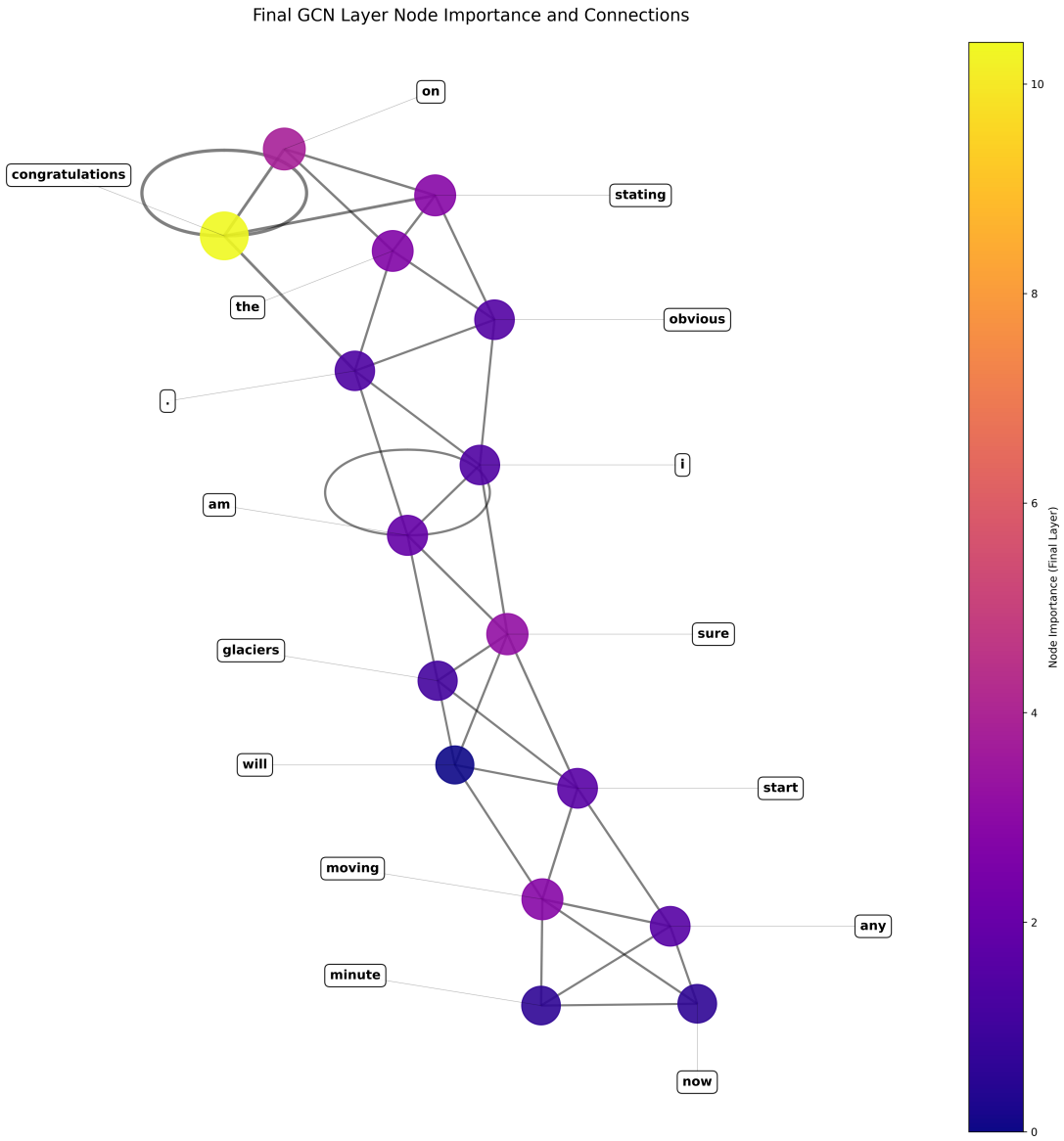


Figure 14: Final GCN Layer Node Importance and Connections

This layer-by-layer analysis reveals:

- Layer 1 shows initial feature extraction with relatively uniform node importance except for a few nodes like ”start” and ”moving”
- In Layer 2, ”congratulations” becomes significantly more important, showing the model’s increasing focus on this key sarcasm marker
- Layer 3 shows peak differentiation of node importance, with ”congratulations” clearly dominant

- The final layer (4) maintains the importance hierarchy while slightly reducing the overall importance values, suggesting feature refinement
- Throughout all layers, the graph structure preserves important connections between semantically related words

10.5 Summary of Findings

Our analysis demonstrates that the hybrid RoBERTa-GCN-LSTM architecture effectively captures both word-level importance and contextual relationships for sarcasm detection. For the example ”Congratulations on stating the obvious. I’m sure glaciers will start moving any minute now”, the model correctly identifies that:

- ”Congratulations” serves as the primary sarcasm indicator, particularly when paired with phrases like ”stating the obvious”
- The phrase ”I am sure” significantly contributes to the sarcastic tone
- Specific word pairs like ”sure glaciers” and ”moving any” provide strong contextual signals for sarcasm
- The model appropriately attends to punctuation and function words that help frame the sarcastic structure
- The GCN layers progressively refine the representation, with the third layer playing a critical role in feature extraction

11 Link to the Trained Models and Test Results

The model and test results can be accessed at the following link: Link.

References

[1] A. Mohan, A. M. Nair, B. Jayakumar, and S. Muraleedharan, “Sarcasm detection using bidirectional encoder representations from transformers and graph convolutional networks,” *Procedia Computer Science*, vol. 218, pp. 93–102, 2023. International Conference on Machine Learning and Data Engineering.

[2] H. Yaghoobian, H. R. Arabnia, and K. Rasheed, “Sarcasm detection: A comparative study,” 2021.

[3] D. Šandor and M. Bagic Babac, “Sarcasm detection in online comments using machine learning,” *Information Discovery and Delivery*, vol. 52, 07 2023.

[4] J. Li, R. Jia, H. He, and P. Liang, "Delete, retrieve, generate: A simple approach to sentiment and style transfer," 2018.

[5] "A large self-annotated corpus for sarcasm." 2017.

[6] S. Castro, D. Hazarika, V. Pérez-Rosas, R. Zimmermann, R. Mihalcea, and S. Poria, "Towards multimodal sarcasm detection (an _obviously_ perfect paper)," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Florence, Italy), Association for Computational Linguistics, 7 2019.

[7] S. Oprea and W. Magdy, "isarcasm: A dataset of intended sarcasm," 2020.