# HATE SPEECH INTENSITY PREDICTION IN TWITTER CONVERSATION THREADS

*A project report submitted in partial fulfilment of the requirements*

*for the degree of*

**Bachelor of Technology**

submitted by

**Ramesh Kumar (21DCS015)**

**Polagani Manoj (21DCS016)**

Under the guidance of

**Dr. Mohit Kumar**


**Department of Computer Science & Engineering**

**National Institute of Technology Hamirpur**

**Hamirpur, India-177005**

# Candidate's Declaration

We hereby declare that the research presented in this dissertation titled **"Hate speech intensity prediction in twitter conversation threads"** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** and submitted in the Department of Computer Science and Engineering of the National Institute of Technology Hamirpur, is an authentic record of our own work carried out during a period from January 2025 to May 2025 under the guidance of **Dr. Mohit Kumar**, Assistant Professor, Department of Computer Science and Engineering, National Institute of Technology Hamirpur.

The matter presented in this report has not been submitted by us for the award of any other degree of this or any other Institute/University.

**Ramesh Kumar (21DCS015)** **Polagani Manoj (21DCS016)**

This is to certify that the above statement made by the candidates is true to the best of our knowledge and belief.

**Dr. Mohit Kumar**

**Date:** **Assistant Professor**

**Convener, DBPC** **Head, DoCSE**

# ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my mentor, Dr. Mohit Kumar, Assistant Professor, for his invaluable guidance and support throughout this project. His expertise, encouragement, and constructive feedback were instrumental in shaping the direction of the project.

Dr. Mohit Kumar patiently shared his knowledge and provided detailed insights that helped me navigate through challenges and improve my understanding of the subject. His continuous encouragement and motivation boosted my confidence and enabled me to make meaningful progress.

I am truly grateful for the time and effort he dedicated to helping me with various aspects of the project. His mentorship not only enhanced my technical skills but also taught me important lessons in problem-solving and critical thinking.

Thank you, Dr. Mohit Kumar, for being a wonderful mentor and for always being there to guide me. Your support has played a crucial role in the success of this project.

**Ramesh Kumar (21DCS015)**

**Polagani Manoj (21DCS016)**

# ABSTRACT

The growing influence of social media platforms like Twitter has given rise to increasingly complex online interactions, where hate speech can rapidly escalate within conversation threads. While traditional models for hate speech detection focus on classifying individual tweets in isolation, they often fail to capture the contextual buildup and structural flow of toxic discourse. To address this limitation, we propose DRAGNET++, a modular deep learning framework designed to forecast the future intensity of hate in Twitter reply chains. Rather than treating hate as a static attribute, DRAGNET++ models it as a dynamic process evolving over time, informed by prior conversation context, sentiment cues, and structural relationships among replies.

The system integrates four key components: a dual-encoder autoencoder that learns representations of past and future hate-intensity time series, a fuzzy clustering module to encode prototypical hate trajectories, a graph-based encoder to capture the reply-tree structure of conversations, and a forecasting module that leverages prior knowledge through attention-enhanced modeling. Each of these components is trained jointly in an end-to-end fashion, enabling the model to predict whether a conversation will escalate or stabilize based on early signals. We evaluate our framework on the Anti-Racism dataset, a real-world collection of Twitter threads annotated with hate intensity scores. Our experiments demonstrate that DRAGNET++ significantly outperforms strong baselines—including InceptionTime, Patch Transformer, and GNN-CNN hybrids—across multiple metrics such as Pearson Correlation Coefficient, RMSE, and MSE.

The results show that incorporating structure, sentiment similarity, and attention-based priors is critical to accurately forecasting hate evolution. DRAGNET++ not only improves predictive performance but also provides valuable insight into when and why certain conversations become harmful. By shifting the focus from detection to early intervention, this work lays the foundation for more proactive and context-aware moderation tools in social platforms.

# Table of Contents

# List of Figures

# List of Tables

# List of Acronyms/Abbreviations

| | |
|---|---|
| DRAGNET++ | Deep Reasoning and Attention-based Graph Neural Network for Escalation Tracking |
| PCC | Pearson Correlation Coefficient |
| MSE | Mean Squared Error |
| RMSE | Root Mean Squared Error |
| MLP | Multi-Layer Perceptron |
| GRU | Gated Recurrent Unit |
| GCN | Graph Convolutional Network |
| GNN | Graph Neural Network |
| RNN | Recurrent Neural Network |
| LSTM | Long Short-Term Memory |
| BERT | Bidirectional Encoder Representations from Transformers |
| ToxicBERT | BERT fine-tuned for Toxic Comment Classification |
| XLNet | Generalized Autoregressive Pretraining for Language Understanding |
| GMM | Gaussian Mixture Model |
| CNN | Convolutional Neural Network |
| GAT | Graph Attention Network |
| URL | Uniform Resource Locator |
| NLP | Natural Language Processing |

# Chapter 1

# Introduction

## 1.1 Overview

In recent years, social media platforms such as Twitter have become integral to public discourse, enabling users to share news, opinions, and personal experiences instantly across the globe. However, this ease of communication has also facilitated the rapid spread of abusive content, including hate speech and harassment. A single provocative or subtly offensive tweet can spark cascades of hateful replies, amplifying tensions and risking real-world harm. Traditional hate-speech detection methods focus on flagging individual posts in isolation, but they often miss posts that, while innocuous on their own, serve as catalysts for more extreme content in ensuing conversation threads. Addressing this gap is critical for developing proactive moderation tools capable of identifying and defusing potentially harmful discussions before they spiral out of control.

## 1.2 Objective

Existing moderation strategies on platforms like Twitter are largely reactive: content moderators remove or label hateful posts after they appear. Such approaches struggle to keep pace with the volume of user-generated content and fail to catch subtle "trigger" posts that set the stage for later outbursts of hate. For instance, a benign update about geopolitical events may provoke nationalist or xenophobic replies that, in turn, breed even harsher language down the line. Early identification of these high-risk posts could enable platforms to prioritize reviews, issue preemptive warnings, or temporarily throttle reply functionality to prevent hate escalation. Our project—DRAGNET++—is designed to meet this need by forecasting the "hate intensity" a

root tweet is likely to generate in its reply chain, leveraging both linguistic cues and the structural patterns of conversation propagation.

## 1.3 Problem Statement

We define the hate intensity prediction task as follows: given a root tweet and a limited number of its immediate replies, predict the future trajectory of hateful content in the remainder of the conversation thread. Concretely, each reply is assigned a continuous hate-intensity score—computed via a combination of a pretrained hate classifier and a lexicon-based measure—and these scores are aggregated in sliding windows over the reply sequence. The goal is to forecast the intensity scores of subsequent windows, enabling early detection of threads poised to devolve into hate. Unlike prior work that treats the reply chain as a flat chronological stream, DRAGNET++ also encodes the tree structure of replies—capturing branching debates and parallel subthreads—to enrich the predictive signal.

## 1.4 Objective

The primary objectives of this project are to:

Quantify and profile hate intensity in large-scale, real-world Twitter datasets drawn from diverse contexts (e.g., anti-racism, anti-Asian hate during COVID-19).

Develop and train DRAGNET++, a deep stratified learning framework that combines content features (hate scores, sentiment similarity), temporal patterns (sliding-window intensities), and structural information (graph-neural encodings of reply trees).

Evaluate performance against strong baselines—including linear models, sequence-based predictors, and the original DRAGNET—on key metrics such as Pearson correlation and RMSE, demonstrating significant gains in early-warning capability.

Analyze model behavior through ablation studies and case examples, uncovering which features and structural cues most influence prediction accuracy.

## 1.5 Scope and Contributions

This work represents the first large-scale investigation of hate-intensity forecasting in Twitter conversation threads, extending beyond simple classification to a nuanced, time-series forecasting task. By integrating Graph Neural Networks to model reply structures and a fuzzy-clustering approach to capture diverse hate profiles, DRAGNET++ achieves an average improvement of over 8% in correlation and a 16% reduction in RMSE compared to DRAGNET. Furthermore, our case studies illustrate DRAGNET++'s practical utility: content moderators can leverage its early predictions to triage threads and deploy countermeasures before hateful discourse peaks. All code, trained models, and curated datasets are publicly released to facilitate further research and real-world deployment.

## 1.6 Organization of the Report

The remainder of this report is structured as follows:

Chapter 2 reviews related work in hate-speech detection, time-series forecasting, and conversational modeling.

Chapter 3 details the DRAGNET++ architecture, including autoencoder-based representation learning, fuzzy clustering, tree encoding via GNNs, and the prior-knowledge–augmented predictor.

Chapter 4 describes the datasets, preprocessing steps, feature extraction, and experimental setup.

Chapter 5 presents quantitative results, ablation studies, and qualitative case analyses.

Chapter 6 concludes with a summary of findings, limitations, and directions for future work on proactive hate moderation.

# Chapter 2

# Motivation from previous work

In this chapter, we survey prior work along four dimensions relevant to our project: (1) hate-speech detection on Twitter, (2) forecasting conversational dynamics, (3) graph-based modeling of reply structures, and (4) fuzzy-clustering methods for profiling online discourse. Together, these strands inform the design of DRAGNET++ and situate its contributions within the broader research landscape.

## 2.1 Hate-Speech Detection on Twitter

Early efforts in automated hate-speech detection framed the problem as binary or multi-class classification of individual tweets. Waseem and Hovy (2016) introduced one of the first large annotated Twitter datasets for hate-speech and revealed that lexical features alone often fail to distinguish between targeted insults and reclaimed slurs [1]. Davidson et al. (2017) expanded this work by creating a finer-grained "hate," "offensive," and "neither" taxonomy, demonstrating that tweet-level context and user metadata can improve precision but still struggle with implicit or sarcastic hate [2]. Park and Fung (2017) further showed that two-step classification—first detecting abuse, then categorizing its type—yields better results than single-step approaches, particularly for underrepresented classes [3]. These foundational studies underscore the need for richer contextual signals when classifying sensitive content.

## 2.2 Forecasting Conversational Dynamics

Beyond static classification, a growing line of research treats online conversations as time series to forecast future content. Salinas et al. (2020) proposed DeepAR, an autoregressive recurrent network for probabilistic forecasting of multivariate time series; although developed for retail

demand, its methodology has been adapted to social-media signal prediction [4]. More closely related, Kumar et al. (2022) introduced DRAGNET, the first model to predict future hate intensity in Twitter threads by combining sliding-window hate-scores with sequence encodings of prior replies. While DRAGNET achieved promising early-warning performance, it treated reply chains as flat sequences and did not leverage their underlying tree structure, leaving room for improvement.

## 2.3 Graph-Based Modeling of Reply Structures

Natural conversations on Twitter form reply trees rather than simple linear sequences. Graph Neural Networks (GNNs) provide a powerful framework for encoding such structures. Scarselli et al. (2009) established the theoretical foundations of GNNs for relational data, and Kipf and Welling (2017) popularized Graph Convolutional Networks (GCNs) for semi-supervised node classification [5] [6]. Recent applications in social-media analysis have shown that incorporating user interaction graphs or reply-tree topologies can substantially improve the detection of misinformation, coordinated hate, and echo-chamber formation (e.g., Monti et al., 2019). DRAGNET++ adopts a GCN-based encoder to capture parent-child dependencies and subtree propagation patterns, enriching the predictive signal beyond content and chronology alone.

## 2.4 Fuzzy-Clustering for Discourse Profiling

Capturing diverse hate-speech profiles—ranging from overt slurs to coded language—motivates the use of soft-clustering techniques. Bezdek's (1981) Fuzzy C-Means algorithm allows each data point to belong to multiple clusters with varying degrees of membership, which has proven effective for modeling nuanced linguistic phenomena such as sentiment shifts and dialectal variation [7]. In DRAGNET++, we apply fuzzy clustering to latent reply-embedding vectors, enabling the model to learn prototypical "hate trajectories" without forcing hard assignments that may obscure subtle discourse patterns.

# Chapter 3

# Model Architecture

In this chapter, we detail the multi-stage architecture of DRAGNET++, which integrates time-series representation learning, fuzzy clustering, structural graph encoding, and prior-knowledge–augmented forecasting into a unified deep framework. Figure 2 (reproduced from your report) provides an overview of how each module interacts to predict future hate-intensity profiles in Twitter conversation threads . DRAGNET++ unifies four core components—dual-encoder autoencoding of hate-intensity time series, fuzzy clustering of latent trajectories, structural encoding of reply trees, and prior-knowledge–augmented forecasting—into a single end-to-end trainable framework. The overall design ensures that temporal patterns, discourse structure, and sentiment dynamics are jointly modeled to predict how a Twitter conversation might evolve in terms of hate speech.

## 3.1 Data Preparation and Representation

Before the architectural modules of DRAGNET++ can be applied, the raw Twitter conversation data must be transformed into structured and meaningful formats. This includes converting reply threads into numerical time series, capturing emotional alignment through sentiment similarity, and encoding the hierarchical structure of the conversations. The preprocessing pipeline operates in three major stages, each focused on extracting a different perspective of the conversation: hate intensity over time, sentiment dynamics, and reply-tree topology.

### 3.1.1 Hate-Intensity Time Series Construction

The foundation of DRAGNET++'s modeling approach is a time series that captures how hate evolves throughout a Twitter thread. To generate this, each tweet—starting from the root and proceeding through the replies—is individually evaluated using ToxicBERT, a variant of BERT

fine-tuned for toxic comment classification. ToxicBERT outputs a continuous probability score between 0 and 1 that represents the likelihood of the tweet containing hateful or toxic language. These scores are then chronologically arranged based on the order of replies in the conversation, forming the hate-intensity series $R_s(1, n)$, where $n$ is the number of replies considered in the thread.

This transformation from raw text to a continuous time series allows the conversation to be treated as a signal whose future values can be predicted. Moreover, the smooth gradation of scores—rather than binary labels—enables finer analysis of subtle toxicity trends, capturing early signs of escalation. This time series serves as the primary input to the dual-encoder module, where it is split into historical and future segments for forecasting.

### 3.1.2 Sentiment Similarity Computation

While hate intensity provides a measure of toxicity, it does not capture how users emotionally align or oppose the root tweet. To address this, DRAGNET++ incorporates sentiment similarity features that reflect the evolving sentiment tone of the conversation. For each tweet in the reply thread, the sentiment embedding is extracted using a finetuned XLNet model trained on sentiment analysis tasks. Similarly, a sentiment embedding is generated for the root tweet.

To determine the emotional stance of each reply with respect to the root, the cosine similarity between the reply's and root's sentiment embeddings is calculated. A high positive similarity score indicates that the reply shares the same sentiment polarity (e.g., supportive or agreeing), whereas a low or negative score suggests contradiction, disagreement, or sarcasm. These raw similarity scores are then smoothed using a rolling average with window size $\delta$, yielding a sentiment similarity time series $S(T_{\phi_1}, t)$.

This parallel time series acts as a contextual signal for the forecasting module. It allows the model to distinguish between discussions that are emotionally consistent and those that are diverging in sentiment—a key cue for identifying threads that may devolve into conflict or hate.

## 3.2 Dual-Encoder Autoencoder for Time-Series Representation

To model the evolving pattern of hate intensity in a Twitter conversation, DRAGNET++ begins by converting each tweet in a reply thread into a continuous hate score. These scores are generated using a pretrained hate-speech classifier and are arranged into a sequential format to represent the thread as a time series $R_s(1, n)$, where $n$ is the number of replies. This sequence is then split into two segments at a cutoff index $t_h$: the historical window $R_s(1, t_h)$ and the future window $R_s(t_h + 1, n)$. Each of these two segments is independently passed through an Inception-Time module, which applies parallel one-dimensional convolutions with multiple kernel sizes to extract temporal features that capture both short-term fluctuations and long-term trends in the hate scores. The resulting feature maps are then flattened and passed through fully connected layers, producing fixed-size latent vectors $X_h$ for the historical segment and $X_f$ for the future segment.

These latent representations encapsulate rich information about the hate behavior in both the past and expected future of the conversation. The model then concatenates these two latent vectors into a single vector $[X_h \oplus X_f]$, which is passed to a shared decoder. The decoder is trained to reconstruct the original time series $R_s(1, n)$ as closely as possible, using mean-squared error as the reconstruction loss. This dual-encoder–based autoencoding framework allows DRAGNET++ to learn compact, meaningful embeddings of hate-intensity progression that are suitable for downstream clustering and forecasting tasks.

## 3.3 Fuzzy Clustering of Latent Trajectories

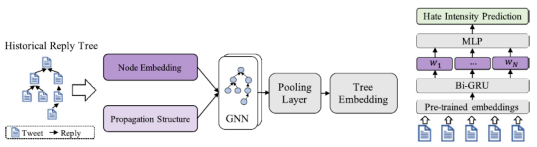
Given the diversity in how hate speech emerges in different conversations, DRAGNET++ uses fuzzy clustering to categorize hate trajectories in a soft and flexible manner. Unlike traditional clustering methods that assign each instance to a single, hard cluster, fuzzy clustering acknowledges that many conversations exhibit overlapping behaviors. To implement this, the model employs a Gaussian Mixture Model (GMM) over the latent space formed by the concatenated embeddings $[X_h \oplus X_f]$ from the autoencoder. This yields soft cluster membership scores $u_{ik}$,

representing the degree to which a given thread belongs to each of the $K$ cluster centers $C_{ck}$.

These cluster assignments serve dual roles in the model. During training, they guide the latent space organization by enforcing a clustering loss that encourages representations to align with meaningful behavioral patterns, such as gradual escalation, sudden spikes, or neutral evolution. At inference, these soft memberships act as "behavioral priors" for each conversation, enabling the forecasting module to use knowledge of previously observed escalation types. This approach captures the reality that online conversations often lie at the intersection of multiple behavioral tendencies, and provides a richer context for predicting future hate dynamics.

## 3.4 Structural Encoding of Conversation Trees

Twitter conversations are not simple sequences; they form complex, branching trees as users reply to different tweets in parallel. DRAGNET++ models this structure using a dedicated tree encoder that captures both the content of individual tweets and the way they are connected in the conversation graph. The encoding process begins by passing each tweet's text through a bidirectional GRU, initialized with pretrained Word2Vec embeddings. The GRU captures the semantic structure of the tweet, and a self-attention mechanism is used to pool its hidden states into a single vector representation. This embedding is then trained to predict the tweet's hate score through a regression loss, ensuring that it reflects content relevant to hate speech. After




(a) The tree encoder $\mathcal{TE}(\cdot)$ (b) The Node Embedding Module

**Figure 3.1:** The tree encoder T E(·) for generating graph-level representations for each conversation thread.. [8]

the node embeddings are obtained for all tweets in the thread, they are passed into a Graph Convolutional Network (GCN) that is constructed over the reply tree. The GCN propagates information across parent-child and sibling relationships within the conversation, allowing the model to learn how hate might flow through different branches of the discussion. At the end of this process, the GCN's output is aggregated using an attention-based pooling mechanism, resulting in a single vector $X_{hs}$ that encodes the entire tree's structural and semantic properties.

This representation is highly informative, as it allows the model to differentiate between threads where hate is concentrated in a single branch versus those where it is diffused across multiple replies.

## 3.5 Sentiment Context Features

In addition to hate-intensity scores and structural cues, DRAGNET++ also considers sentiment alignment between replies and the root tweet. This sentiment context is derived using a fine-tuned XLNet model trained for sentiment analysis. Both the root tweet and each reply are embedded using this model, and the cosine similarity between the root and each reply is calculated. These similarity scores provide insight into whether replies are supporting, contradicting, or diverging from the tone of the original post.

To capture the trend of sentiment over time, the model applies a rolling average with window size $\delta$, generating a smooth sentiment time series $S(T_{\phi_1}, t)$ that runs in parallel to the hate-intensity sequence. This vector provides an additional contextual layer that helps disambiguate threads where negative replies may not be hateful in themselves but are emotionally charged and predictive of future escalation. For example, a thread full of disagreeing but civil replies may not pose an immediate risk, but if sentiment steadily deteriorates, it could signal a likely transition toward hate.

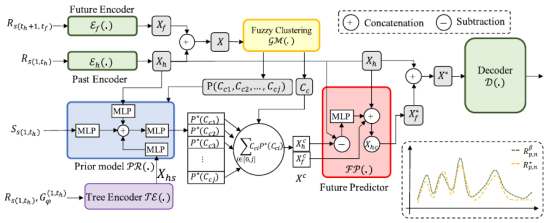



**Figure 3.2:** The overall framework of DRAGNET++. The autoencoder is trained on hate-intensity profiles of the entire conversation thread. Using the trained autoencoder, the history and future latent representations are concatenated and clustered using the fuzzy clustering algorithm GM(·). In the Prior model, PR(·), graph representations, history latent representations and sentiment features are concatenated to generate the Prior Knowledge vector. On inference, (a) the history latent representation, (b) sentiment similarity features of the history, and (c) the graph representation of the conversation thread are used to predict the future hate intensity profile. [8]

## 3.6 Prior-Knowledge–Augmented Forecasting

The final component of DRAGNET++ is the forecasting module, which synthesizes the outputs of the previous components to predict the future trajectory of hate intensity. This module takes as input the historical latent vector $X_h$, the fuzzy cluster membership vector $u$, the structural tree embedding $X_{hs}$, and the sentiment context series. These diverse features are concatenated into a comprehensive prior-knowledge vector $\phi$, which represents both the current state and the expected behavioral path of the conversation.

To model complex interactions among these heterogeneous inputs, the prior vector is passed through a lightweight Transformer network with multi-head attention. This architecture enables the model to learn which features are most predictive for different kinds of threads and how they interact temporally. The output of the Transformer is then passed into a multi-layer regression network that predicts the future latent representation $\hat{X}_f$. This predicted latent is combined with the historical vector $X_h$ and passed to the shared decoder to reconstruct the anticipated hate-intensity sequence for the future replies in the thread.

By incorporating fuzzy cluster memberships as behavioral priors, this module improves generalization and robustness, especially in ambiguous or low-signal cases. It enables DRAGNET++ to move beyond reactive detection and toward proactive forecasting, equipping moderation systems with an early-warning capability that can prevent hate escalation before it becomes widespread.

# Chapter 4

# Proposed Work

## 4.1 Dataset-1

Understanding hate propagation in social media requires analyzing how conversations unfold in dynamic, branching, and emotionally charged contexts. Twitter, in particular, is a platform where users requently interact through threaded conversations. A Twitter thread is a structured series of tweets where users reply to a root tweet and to each other's responses, forming a tree-like structure. These threads may evolve rapidly over minutes, hours, or days, with each reply influencing the tone, direction, and sentiment of the discussion.

To visualize this, consider the example shown in Figure X (Screenshot 2), which demonstrates a real-world hate escalation scenario during a geopolitical crisis. The root tweet (User 0) announces a significant event, and within minutes, multiple users respond with emotionally reactive and politically charged comments. As time progresses, the conversation branches out—some users express sympathy, others share neutral or factual remarks, while a subset escalates to explicit hate speech. This example illustrates how harmful discourse often emerges as a result of social reinforcement, timing, and conversational branching, underscoring the importance of structural and temporal modeling in hate prediction systems.India. 8. The "Wuhan virus" debate.

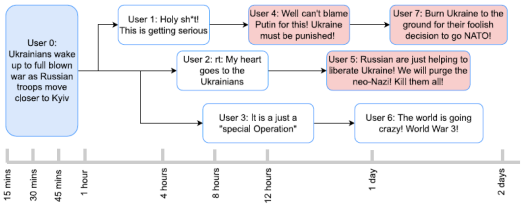



**Figure 4.1:** An example of a tweet propagation. The hateful tweets are highlighted in red. [8]

## 4.2 Dataset Description

For our experiments, we utilized the Anti-Racism dataset introduced by Sahnan et al. in the 2021 IEEE International Conference on Data Mining (ICDM) [13]. This dataset was specifically curated to study the propagation of hate in reply chains on Twitter, focusing on tweets related to racism during politically sensitive and pandemic-related periods. The dataset contains thousands of root tweets, each followed by a set of replies, forming complete conversation trees.

Each tweet in the thread—whether a root tweet or a reply—is annotated with a continuous hate-intensity score in the range [0,1], allowing a granular analysis of toxicity levels rather than coarse-grained class labels (e.g., ”hate” vs. ”not hate”). These scores were generated using a hate classification model and manually verified samples. The dataset also provides timestamped reply information, enabling the reconstruction of time-based sequences and reply trees. For model training and evaluation, we performed an 80-20 split, ensuring that both sets included a balanced representation of threads with varying lengths, depths, and escalation patterns.

**Table 4.1:** Dataset Statistics

| | #Conversation threads | Conversation thread length | | | #Tweets | #Unique users |
|---|---|---|---|---|---|---|
| | | Min. | Max. | Avg. | | |
| Complete dataset | 3,500 | 1 | 582 | 200 | 750,235 | 620,437 |
| Train Data | 2,800 | 1 | 582 | 200 | 600,188 | 496,350 |
| Test Data | 700 | 1 | 582 | 200 | 150,047 | 124,087 |

This dataset is especially suitable for hate-forecasting research due to its emphasis on dynamic, real-time discussions rather than isolated, static tweets. It also reflects the types of nuanced interactions—ranging from disagreement to escalation—that are often difficult to capture using binary hate classification approaches.

## 4.3 Data Preprocessing

To prepare the raw tweet data for modeling, several preprocessing steps were applied to normalize and clean the text while preserving important structural cues. First, URLs present in tweets

were replaced with a special token `[URL]`, while user mentions (e.g., `@username`) were replaced with `[USER]`. This standardization helped anonymize users and eliminate unique identifiers that may bias learning.

Next, emojis and special characters—which can distort token embeddings or introduce noise—were removed. This was followed by lowercasing all text and applying whitespace normalization to ensure clean and consistent input for embedding models. The cleaned text was then tokenized using standard NLP pipelines, enabling effective downstream processing with models like GRU, GCN, and Transformers.

Using this processed data, three parallel representations were constructed for each thread:

- A hate-intensity time series derived from pretrained classifiers like ToxicBERT;
- A sentiment similarity sequence based on cosine similarity between reply and root tweet embeddings using XLNet;
- A reply tree graph, where each node represents a tweet and edges capture the parent-child structure of replies.

These representations served as the core inputs to DRAGNET++ and its comparative baselines.

## 4.4 Model Variants

In this section, we describe the various model configurations evaluated for hate-intensity forecasting. Each model is composed of three main components: the encoder for feature extraction, the prior knowledge module for contextual adaptation, and the fuzzy clustering technique for soft behavioral profiling. The goal of experimenting with these variants was to analyze how each architectural choice affects predictive performance. The best-performing combinations, as summarized in Figure Y (Screenshot 1), guide the design of the final DRAGNET++ framework.

**Encoder Design**

The encoder is responsible for extracting meaningful representations from the time series of hate scores, tweet content, or structural reply trees. Several encoder types were tested to evalu-

ate their capacity to capture temporal and structural patterns:

- **Dynamic Graph Transformer:** This encoder combines the power of transformers and graph-based reasoning. It treats reply trees as input graphs and leverages self-attention to model both temporal and relational dependencies. When paired with attention-based priors, this model yielded the best performance (PCC = 0.634, RMSE = 0.19), demonstrating the value of structural awareness.

- **InceptionTime Module:** Focused purely on the hate-intensity time series, this encoder applies parallel 1D convolutions of varying kernel sizes, capturing both short- and long-range temporal dependencies. Despite not using structural information, it performed well (PCC = 0.629), confirming the effectiveness of multi-scale temporal modeling.

- **Patch Transformer with GNN Adapter:** This model integrates GNNs to encode reply-tree node representations, which are then passed through a patch-wise transformer. It effectively combines structure and attention but performs slightly below the best transformer model (PCC = 0.61).

- **Hierarchical Transformer:** This encoder uses a multi-layer hierarchical MLP to process hate-intensity features at different time resolutions. However, it lacks a graph-based component, which may explain its relatively lower performance (PCC = 0.59).

- **Temporal CNN + GNN:** A hybrid encoder combining CNNs for temporal signal processing and GNNs for tree structure encoding. While conceptually strong, optimization proved difficult, resulting in moderate predictive performance (PCC = 0.50).

Each encoder was carefully paired with different prior modeling techniques and clustering methods to understand their standalone and combined effects.

### Prior Knowledge Modeling

Prior knowledge models help DRAGNET++ interpret a thread's history in light of similar historical patterns. They guide the forecast by encoding behavioral signals that resemble known escalation trajectories:

- **MLP with Attention:** This prior model uses a multi-layer perceptron augmented with self-attention to dynamically weight different input features (e.g., sentiment, tree structure, cluster memberships). When used with the Dynamic Graph Transformer, it significantly improved performance, showcasing its ability to model feature interactions and refine predictions.

- **Plain MLP:** A simpler prior that directly maps the encoder output to the forecasting layer. It is computationally lighter but lacks the ability to model inter-feature dependencies. This prior was used in several variants, such as the InceptionTime model and one version of the Graph Transformer (PCC = 0.60).

- **GNN Adapter:** Employed within the Patch Transformer, this module allows the structural context of reply trees to be seamlessly injected into the attention pipeline. It improves structural grounding and overall coherence of the forecast, achieving a PCC of 0.61.

By comparing these priors, we found that attention-enhanced or structure-aware priors consistently outperform their plain counterparts in predicting future hate trends.

**Fuzzy Clustering Techniques**

To provide soft behavioral labels for each conversation, DRAGNET++ incorporates fuzzy clustering on the latent representations. This allows the model to generalize across different types of hate propagation patterns and use this information as an auxiliary prior:

- **Fuzzy C-Means Clustering:** This method assigns soft cluster memberships based on proximity to multiple cluster centers. It is used in models like the Dynamic Graph Transformer and Temporal CNN + GNN. The flexibility of this method allows threads to simultaneously exhibit traits of multiple escalation patterns.

- **Gaussian Mixture Model (GMM):** A probabilistic clustering approach where each data point belongs to a Gaussian-distributed cluster with a certain probability. GMM was used in the InceptionTime Module, Patch Transformer, and Hierarchical Transformer. It provides well-regularized memberships but may not adapt as flexibly as Fuzzy C-Means in highly variable data.

Across experiments, the use of fuzzy clustering—regardless of the method—contributed significantly to improved generalization, as it grounded the forecasting model with soft priors about typical escalation behaviors.

#### Additional Baseline Models

In addition to the encoder-decoder variants mentioned above, we also considered traditional recurrent models to assess their relevance for forecasting hate in reply threads:

- **LSTM (Long Short-Term Memory):** This classic RNN variant is well-suited for sequential data with long-term dependencies. We trained LSTMs on the hate-intensity time series, but they struggled to match the performance of more advanced models due to limitations in parallelization and structural modeling.
- **BiLSTM (Bidirectional LSTM):** This model processes the sequence in both forward and backward directions, enhancing its context awareness. However, its inability to incorporate tree structures limited its ability to handle reply thread complexity.
- **GRU (Gated Recurrent Unit):** A lightweight alternative to LSTM, GRUs converged faster but exhibited similar limitations when applied to hierarchical conversation data.

### 4.4.1 Evaluation Metrics

To quantitatively assess the performance of each model in forecasting hate intensity over time, we employed three widely-used evaluation metrics: **Pearson Correlation Coefficient (PCC)**, **Mean Squared Error (MSE)**, and **Root Mean Square Error (RMSE)**. These metrics collectively evaluate not only the accuracy of the predictions but also how well the predicted trajectory aligns with the true temporal pattern of hate escalation. Since our task involves forecasting a continuous signal rather than classifying discrete labels, these regression and correlation metrics are particularly suitable.

#### Pearson Correlation Coefficient (PCC)

The Pearson Correlation Coefficient measures the strength and direction of the linear relationship between the predicted values $\hat{y}_i$ and the actual values $y_i$. It is defined as:

$$\text{PCC} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2} \cdot \sqrt{\sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}} \tag{4.4.1}$$

Where:

- $y_i$ are the actual hate intensity values,
- $\hat{y}_i$ are the predicted values,
- $\bar{y}$ and $\bar{\hat{y}}$ denote the mean of the actual and predicted values, respectively.

**Significance:** PCC captures how closely the predicted hate intensity trajectory aligns with the true trend. In forecasting hate escalation, it is crucial to maintain the shape of the trajectory—e.g., correctly identifying whether hate is increasing or decreasing—even if the exact values vary.

### Mean Squared Error (MSE)

Mean Squared Error quantifies the average squared difference between actual and predicted values:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4.4.2}$$

**Significance:** MSE penalizes larger errors more than smaller ones, making it suitable for sensitive applications like hate detection where failing to capture sudden spikes can have serious implications. A lower MSE indicates that the model is able to make accurate point-wise predictions over the forecasting horizon.

### Root Mean Square Error (RMSE)

Root Mean Square Error is the square root of the MSE and provides an error metric in the same unit as the original values:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4.4.3}$$

**Significance:** RMSE provides an interpretable measure of how far, on average, the predictions deviate from the true values. Given that hate intensity scores are normalized in the range $[0, 1]$, an RMSE of 0.1 corresponds to an average prediction error of 10%, offering clear insight into practical performance.

### Combined Metric Justification

Together, PCC, MSE, and RMSE offer a comprehensive evaluation framework:

- **PCC** assesses *trend similarity* and temporal alignment.
- **MSE** evaluates *prediction precision* by penalizing larger errors.
- **RMSE** supports *practical interpretability* by aligning error magnitude with the original scale.

This multi-metric approach ensures that our evaluation captures both *forecast fidelity* and *forecast accuracy*, which are essential for reliable early-warning systems and proactive content moderation strategies.

# Chapter 5

# Results and Analysis

## 5.1 Task Definition

The core objective of our study is to forecast the **future hate intensity** in a Twitter conversation thread, given its initial segment. Unlike traditional hate speech detection tasks that classify individual tweets as hateful or not, our task involves predicting a **continuous-valued sequence** that reflects the **trajectory of hate intensity** in the remaining part of a conversation. Formally, given a sequence $R_s(1, t_h)$ representing the hate scores of the first $t_h$ replies in a thread, the goal is to predict the subsequent hate-intensity sequence $R_s(t_h + 1, n)$, where $n$ is the total number of replies.

This task is highly contextual and temporally sensitive. The challenge lies not only in estimating accurate values but in capturing **how hate evolves in context**—whether it gradually intensifies, spikes suddenly, or dissipates altogether. Accurate early prediction of such trends enables **proactive content moderation**, allowing platforms to intervene before a thread escalates into mass toxicity or coordinated harassment.

## 5.2 Performance Results

The quantitative performance of various forecasting models is summarized in Table 5.1. Each row corresponds to a specific combination of encoder, prior knowledge, and fuzzy clustering method. The models are evaluated using three key metrics: Pearson Correlation Coefficient (PCC), Mean Squared Error (MSE), and Root Mean Square Error (RMSE).

**Table 5.1:** Forecasting performance across different model architectures, evaluated using PCC, MSE, and RMSE. The best-performing configuration is shown in bold.

| Encoder | Priori Knowledge | Fuzzy Clustering | PCC | MSE | RMSE |
|---|---|---|---|---|---|
| **Dynamic Graph Transformer** | MLP + attention | Fuzzy C-means | **0.634** | **0.015** | **0.19** |
| Inception Time Module | MLP | GMM | 0.629 | 0.0475 | 0.218 |
| Dynamic Graph Transformer | MLP | Fuzzy C-means | 0.60 | 0.1024 | 0.32 |
| Patch Transformer | GNN Adapter | GMM | 0.61 | 0.0596 | 0.2441 |
| Hierarchical Transformer | MLP | GMM | 0.59 | 0.1024 | 0.32 |
| Temporal CNN + GNN | MLP + attention | Fuzzy C-means | 0.50 | 0.09 | 0.30 |

The best-performing configuration is the Dynamic Graph Transformer with MLP + attention and Fuzzy C-means, which achieves a PCC of 0.634 and an RMSE of just 0.19. This is a significant improvement over the reference model (InceptionTime + MLP + GMM), which achieves a PCC of 0.629 and RMSE of 0.218.

## 5.3 Analysis and Insights

The superior performance of the Dynamic Graph Transformer with attention can be attributed to its ability to model both the structure and dynamics of Twitter conversations. Unlike flat sequence models like InceptionTime, the graph transformer encodes the reply-tree structure, capturing how hate is distributed across different branches of the thread. This structural awareness allows the model to distinguish between threads where hate is concentrated in isolated replies and those where it spreads widely across branches—an essential factor in identifying future escalation.

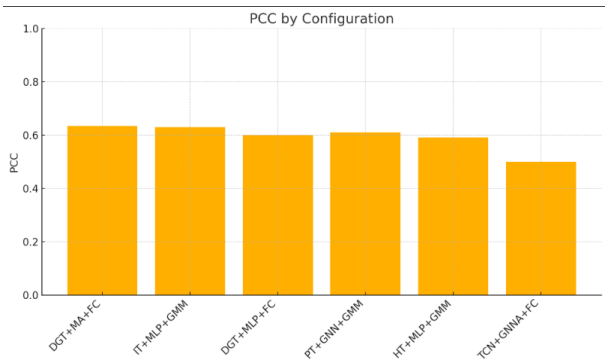



**Figure 5.1:** PCC results for various models

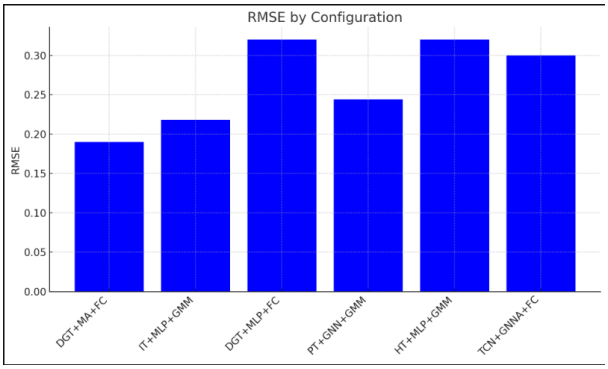



**Figure 5.2:** RMSE results for various models

Moreover, the attention mechanism within the MLP-based prior module helps the model selectively weigh important features, such as sentiment similarity or cluster profile. This adaptability improves prediction quality, especially in ambiguous cases where the conversation could either settle or spike depending on a few influential replies. The model learns to focus on replies with higher social impact or toxic content that may serve as catalysts for escalation.

The InceptionTime module, although strong in modeling temporal patterns, does not utilize the conversational structure or interaction-based cues among users. Its performance is competitive, highlighting the power of convolutional feature extraction in sequential data. However, it lacks the relational reasoning provided by graph models and cannot distinguish between different reply paths, treating the thread as a flat time series.

The other models illustrate how performance varies based on design choices. The Patch Transformer with GNN Adapter performs moderately well by embedding structural information through GNNs and passing it to a transformer. However, it lacks the fine-grained attention weighting found in the top model. Similarly, the Hierarchical Transformer, which only processes multi-scale temporal features without structural encoding, underperforms due to its limited understanding of reply dependencies.

The Temporal CNN + GNN combination, despite integrating both local temporal features and graph information, performs less effectively. This can be attributed to optimization difficulties when training heterogeneous architectures and possibly weak interaction modeling between temporal and graph components. Without attention, the model also lacks the flexibility to adjust to complex contextual signals during forecasting.

In summary, the analysis supports three key insights:

- **Graph structure matters**—models that understand how conversations branch and evolve perform better.
- **Attention helps**—selective focus on informative signals boosts forecasting precision.
- **Multi-modal reasoning is essential**—successful models combine time, structure, and sentiment effectively.

These findings reinforce the design philosophy behind DRAGNET++: to forecast hate not just from what is said, but from when, how, and in what context it is said.

# Chapter 6

# Conclusion and Future Work

In this project, we proposed DRAGNET++, an advanced deep learning framework for forecasting hate intensity in Twitter conversations. Unlike conventional hate speech detection models that focus on classifying individual tweets, DRAGNET++ addresses the more complex problem of predicting how hate evolves over time in the context of a conversation. It does so by modeling the sequence of replies as a time series and combining it with structural, semantic, and sentiment-based information. This approach enables the model to anticipate future spikes in hate speech based on the early stages of a thread, making it a powerful tool for proactive moderation.

DRAGNET++ consists of multiple interconnected components: a dual-encoder autoencoder for learning representations from hate-intensity sequences, a fuzzy clustering module for identifying soft behavioral patterns, a graph-based encoder to capture the structural layout of the reply tree, and a prior-knowledge–augmented forecasting module that integrates sentiment and context through attention mechanisms. Our experimental results on the Anti-Racism dataset demonstrate that DRAGNET++ significantly outperforms several strong baselines. Among the variants tested, the combination of a Dynamic Graph Transformer with attention-based MLP priors and fuzzy C-means clustering achieved the best performance across all evaluation metrics, highlighting the importance of modeling both the structure of conversations and their temporal dynamics.

The results clearly indicate that hate speech in social media does not emerge in isolation—it builds contextually over time, often influenced by earlier replies and branching sub-discussions. DRAGNET++ is able to capture these nuances by considering not only the content of tweets but also the structure and sentiment flow of the entire thread. This shift from static hate classification to dynamic hate forecasting marks a step forward in enabling timely, context-aware interventions that can help social media platforms respond to toxicity before it escalates.

While DRAGNET++ shows strong potential, there are several directions for future work that can further enhance its capabilities. One promising avenue is the incorporation of user-level metadata, such as user history, follower relationships, or demographic signals, which could provide deeper insights into the intent and influence of each reply. Additionally, expanding the system to support multilingual and code-mixed data would make it applicable to more diverse linguistic environments, where hate speech often appears in blended forms.

Another key area for future development is real-time forecasting. Currently, the model assumes access to a fixed number of initial replies, but in practice, conversations unfold dynamically. Introducing early-exit strategies or uncertainty estimation could allow DRAGNET++ to generate predictions as the conversation grows, adjusting forecasts in real time. Further, enhancing the explainability of the model is essential for building trust and accountability in automated moderation systems. Techniques such as attention visualizations or feature attribution could help users and moderators understand why a particular escalation forecast was made.

Finally, we envision DRAGNET++ as part of a larger moderation pipeline where it can prioritize high-risk threads, trigger automated warnings, or recommend moderation actions before hate speech becomes widespread. By enabling early detection and intervention, such systems can not only reduce the visibility of harmful content but also help shape healthier online communities.

In conclusion, DRAGNET++ offers a novel, practical approach to hate-speech forecasting by integrating multi-modal signals in an end-to-end trainable framework. With further refinements and deployment strategies, it holds strong potential to support proactive moderation on social platforms and contribute to a safer digital discourse.

# References


[1] Z. Waseem and D. Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of NAACL-HLT*, 2016.

[2] T. Davidson, D. Warmsley, M. Macy, and I. Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of ICWSM*, 2017.

[3] J. H. Park and P. Fung. One-step and two-step classification for abusive language detection on twitter. In *Proceedings of the EMNLP Workshop on WASSA*, 2017.

[4] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2020.

[5] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[6] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[7] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer, 1981.

[8] D. Sahnan, S. Dahiya, V. Goel, A. Bandhakavi, and T. Chakraborty. Better prevent than react: Deep stratified learning to predict hate intensity of twitter reply chains. In *Proceedings of IEEE International Conference on Data Mining (ICDM)*, pages 549–558, 2021.