

Article

A Graph Neural Network-Based Context-Aware Framework for Sentiment Analysis Classification in Chinese Microblogs

Zhesheng Jin and Yunhua Zhang *

Department of Computer Science & Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China; 202230603070@mails.zstu.edu.cn

* Correspondence: zhang-yunhua@zstu.edu.cn; Tel.: +86-13606800508

Abstract: Sentiment analysis in Chinese microblogs is challenged by complex syntactic structures and fine-grained sentiment shifts. To address these challenges, a Contextually Enriched Graph Neural Network (CE-GNN) is proposed, integrating self-supervised learning, context-aware sentiment embeddings, and Graph Neural Networks (GNNs) to enhance sentiment classification. First, CE-GNN is pre-trained on a large corpus of unlabeled text through self-supervised learning, where Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) are leveraged to obtain contextualized embeddings. These embeddings are then refined through a context-aware sentiment embedding layer, which is dynamically adjusted based on the surrounding text to improve sentiment sensitivity. Next, syntactic dependencies are captured by Graph Neural Networks (GNNs), where words are represented as nodes and syntactic relationships are denoted as edges. Through this graph-based structure, complex sentence structures, particularly in Chinese, can be interpreted more effectively. Finally, the model is fine-tuned on a labeled dataset, achieving state-of-the-art performance in sentiment classification. Experimental results demonstrate that CE-GNN achieves superior accuracy, with a Macro F-measure of 80.21% and a Micro F-measure of 82.93%. Ablation studies further confirm that each module contributes significantly to the overall performance.

Keywords: sentiment analysis; Chinese microblogs; graph neural networks; context-aware embeddings; syntactic structure modeling



Received: 25 February 2025

Revised: 12 March 2025

Accepted: 15 March 2025

Published: 18 March 2025

Citation: Jin, Z.; Zhang, Y. A Graph Neural Network-Based Context-Aware Framework for Sentiment Analysis Classification in Chinese Microblogs. *Mathematics* **2025**, *13*, 997. <https://doi.org/10.3390/math13060997>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

MSC: 68T07

1. Introduction

The rapid expansion of social media platforms, particularly Weibo, has led to the generation of an immense volume of user-generated content [1,2]. This content provides a wealth of data reflecting public sentiment and opinions, making Sentiment Analysis (SA) on these platforms a critical task for various applications, including marketing, political strategy, and public relations. However, effectively capturing the complex and nuanced sentiment expressed in natural language remains a significant challenge [3,4].

In recent years, with the increasing availability of sensor data from mobile devices and other hardware, multimodal approaches to sentiment analysis have emerged. These approaches integrate data from various sensors, such as accelerometers, GPS, and even physiological sensors like heart rate monitors, to enrich textual sentiment data with contextual information [5]. This multimodal fusion allows for deeper insights into user sentiment by correlating textual data with behavioral patterns or environmental context captured

through sensors, thereby improving the robustness of sentiment predictions. In IoT environments, where devices such as smart assistants and wearables generate continuous data, our model's ability to capture fine-grained sentiment shifts could enable adaptive, personalized services that respond to user emotions in real time. For example, integrating CE-GNN with sensor inputs in smart devices may help improve user experience by adjusting device settings based on detected sentiment, offering a pathway for sentiment-based personalization across various applications [6,7].

Traditional sentiment analysis approaches have primarily relied on machine learning techniques that require extensive manual feature engineering [5,8]. Although these methods have been successful in certain contexts, they are often constrained by their inability to generalize across different domains and handle the intricate syntactic structures inherent in natural language. The advent of deep learning has revolutionized SA, allowing models to learn features directly from raw text data. This shift has resulted in substantial improvements in both the accuracy and depth of sentiment classification. Nevertheless, several critical challenges persist in the field [3,9,10].

One of the major challenges is the integration of syntactic information into sentiment analysis models. Deep learning architectures like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have proven effective in capturing local and sequential patterns within text [11,12]. However, these models typically treat text as a flat sequence of tokens, which leads to the loss of important hierarchical syntactic structures that are crucial for understanding nuanced sentiment [13]. For instance, modeling the dependencies between words in a sentence is essential for accurately interpreting sentiment, particularly in languages like Chinese, where word order and grammatical constructs significantly impact meaning [14].

In Chinese, the lack of morphological variations and heavy reliance on word order and contextual cues pose unique challenges for sentiment analysis. For example, dependency parsing plays a crucial role in understanding syntactic relationships, as the emotional meaning of a sentence often depends on subtle dependencies between words. Effectively capturing these relationships is essential for achieving accurate sentiment classification.

In recent years, Transformer-based models such as BERT have set new benchmarks in sentiment analysis by leveraging self-attention mechanisms to capture long-range dependencies and contextual relationships within text [15–17]. These models have shown remarkable success in handling context-dependent sentiment across entire sentences. However, they often struggle with fine-grained sentiment shifts within sentences, especially in cases where complex syntactic structures are involved [18]. Transformers, while powerful, typically operate at the token level and may overlook the deeper syntactic and semantic relationships necessary for precise sentiment analysis. Graph Neural Networks (GNNs) are uniquely suited to address these limitations by representing words as nodes and their syntactic relationships as edges in a graph. This structure allows GNNs to naturally model hierarchical and dependency-based relationships within sentences, capturing both local and global syntactic information. Unlike RNNs and CNNs, which struggle to capture long-range dependencies effectively, GNNs propagate information across the graph structure, making them well-equipped for tasks requiring nuanced syntactic understanding. Additionally, GNNs can flexibly incorporate external linguistic knowledge, such as dependency parsers, to further enhance the representation of syntactic relationships. This ability to integrate syntactic information while maintaining computational efficiency makes GNNs an ideal choice for sentiment analysis tasks involving complex languages like Chinese.

Another significant challenge in sentiment analysis is the reliance on large amounts of labeled data to train deep learning models [19]. The process of manually annotating data is both time-consuming and costly, which limits the scalability of sentiment analysis

applications. In response to this challenge, self-supervised learning has emerged as a promising alternative. By pre-training models on vast amounts of unlabeled data, self-supervised learning allows models to learn general linguistic patterns that can be fine-tuned for specific tasks. However, existing self-supervised methods often lack tasks specifically designed to capture the subtle nuances of sentiment expression, particularly in languages with complex grammar and syntax [20].

Despite these advancements, key issues remain unresolved, including the effective integration of syntactic information, the handling of fine-grained sentiment shifts, and the development of models that can generalize across different contexts without relying heavily on labeled data. These challenges underscore the need for innovative approaches in sentiment analysis that can address these limitations and enhance the robustness of sentiment models, especially for complex languages like Chinese, where word order and grammar can significantly alter meaning [21,22]. Additionally, incorporating sensor data could further aid in capturing the emotional context of user interactions, providing a richer and more comprehensive understanding of sentiment.

In this work, we address the aforementioned challenges by proposing a novel approach that integrates Graph Neural Networks (GNNs), sentiment embeddings with context awareness, and self-supervised learning for sentiment classification in Chinese microblogs. Our key contributions are as follows:

- We design specific self-supervised learning tasks to pre-train the model on large-scale unlabeled data, allowing it to learn fine-grained sentiment features. These tasks are tailored to the intricacies of sentiment expression, significantly enhancing the model's generalization ability.
- We introduce a sentiment embedding layer that maps words into a multi-dimensional space, capturing both semantic and sentiment characteristics. This embedding is dynamically adjusted through a context-awareness mechanism, enabling the model to accurately reflect sentiment shifts across different contexts.
- We develop a GNN-based model that captures the syntactic dependencies within microblog texts. By representing words and phrases as nodes and dependency relations as edges, our GNN effectively models complex syntactic structures, which are essential for accurate sentiment classification. While our primary focus is on text-based sentiment analysis, we highlight potential extensions where sensor data could enrich the model's understanding of user context, offering a path for future research in multimodal sentiment analysis.

Our experimental results demonstrate that the proposed model achieves relatively advanced performance in Chinese microblog sentiment classification, outperforming existing methods in both accuracy and robustness, particularly in handling complex syntactic structures and contextual variations. While our primary focus is text-based sentiment analysis in social media, we envision potential extensions to IoT applications, where CE-GNN could integrate sensor data to enhance real-time sentiment understanding, enabling personalized services across smart devices and environments.

2. Related Work

2.1. Traditional Sentiment Analysis Approaches

Traditional sentiment analysis approaches have primarily relied on manually engineered features coupled with classical machine learning algorithms. These methods often emphasize linguistic features such as n-grams, part-of-speech tags, and syntactic patterns [23,24]. While these approaches have laid the foundation for sentiment analysis, their reliance on domain-specific features limits their generalizability across different con-

texts and languages. Furthermore, these methods typically treat text as a flat sequence of tokens, disregarding the hierarchical and syntactic structures crucial for capturing nuanced sentiment expressions. Additionally, traditional approaches often neglect the potential contributions of non-textual data, such as sensor data, which could offer valuable contextual information about users' physical states or environments during sentiment expression.

2.2. Deep Learning for Sentiment Analysis

The advent of deep learning brought a transformative change to sentiment analysis, with models capable of automatically learning hierarchical representations from raw text data. Convolutional Neural Networks (CNNs) were among the first deep learning models applied to sentiment analysis, recognized for their ability to extract local features, particularly those that are sentiment-rich, from text [25]. Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, further advanced the field by capturing sequential dependencies, thus providing a deeper understanding of sentiment flow across sentences and documents [26,27]. These models have also begun to incorporate multimodal inputs, such as data from sensors like accelerometers and heart rate monitors, to better understand the interplay between textual sentiment and physical context, offering a more comprehensive sentiment analysis framework. However, despite their success, these models tend to overlook syntactic relationships between words, treating text as a simple sequence without considering its underlying grammatical structure, which is crucial for accurate sentiment interpretation. The graph neural network makes up for this shortcoming.

Graph Neural Networks (GNNs) have been widely applied in Natural Language Processing (NLP) due to their ability to model relational and structural information. Unlike traditional sequence-based models, GNNs capture global dependencies between words, entities, or sentences. Several studies have demonstrated the effectiveness of GNNs in NLP tasks, including text classification, relation extraction, and knowledge graph reasoning. For instance, Text-GNN [28] introduces a graph-based representation for document classification, utilizing word co-occurrence graphs. Similarly, SG-Net [29] enhances named entity recognition by leveraging syntactic structures through GNNs. Moreover, knowledge graph-based models, such as KBGNN [30], have been utilized to enhance question answering systems.

2.3. Incorporating Syntactic Information

To overcome the limitations of treating text as a flat sequence, researchers have explored the integration of syntactic information into sentiment analysis models. Techniques such as dependency parsing and tree-based models have been employed to capture the grammatical relationships between words, enhancing the model's ability to interpret complex sentences [31,32]. These approaches allow models to focus on syntactic structures that contribute significantly to sentiment, thereby improving interpretability and accuracy.

2.4. Transformer Models and Contextual Sentiment Analysis

Transformer-based models, such as BERT, have established new standards in sentiment analysis by leveraging self-attention mechanisms to capture long-range dependencies and contextual relationships within text [33]. These models excel at understanding context-dependent sentiment across entire sentences, making them particularly effective in dealing with nuanced language. The use of sensors can further enhance Transformer models by providing external context, such as location data or biometric feedback, which helps to interpret sentiment within specific situational contexts. However, challenges remain in capturing fine-grained sentiment shifts within sentences, especially in languages with complex syntactic structures like Chinese [34]. Furthermore, the reliance on large labeled

datasets for fine-tuning these models highlights the need for more efficient and scalable training strategies, particularly in resource-constrained environments.

Despite their success, Transformers suffer from several limitations when applied to graph-structured data. First, Quadratic Complexity: The self-attention mechanism in Transformers has an $O(L^2)$ complexity, where L is the sequence length. This makes it computationally expensive for long sequences or large graphs [35]. Second, Lack of Explicit Structural Modeling: Transformers do not inherently model graph structures, requiring additional positional encoding methods, such as Graphormer, to incorporate relational information. Third, Weak Locality Modeling: Unlike CNNs or GNNs, vanilla Transformers struggle to efficiently model local dependencies without extensive modifications [36]. These limitations motivate the development of hybrid models, such as T-GCN and Graph WaveNet, which combine graph-based learning with sequential modeling.

2.5. Self-Supervised Learning in Sentiment Analysis

To address the dependency on large labeled datasets, self-supervised learning has emerged as a promising approach. This paradigm allows models to be pre-trained on vast amounts of unlabeled data, learning general language representations that can be fine-tuned for specific tasks such as sentiment analysis [37]. Self-supervised tasks, such as masked language modeling, have shown significant potential in enhancing model performance across various NLP tasks. In addition to text-based tasks, the integration of sensor data into self-supervised learning presents an opportunity to train models on multimodal inputs, enabling them to learn richer representations that account for both linguistic and physical contexts. However, these tasks are often broadly designed and may not fully capture the intricate requirements of sentiment classification, particularly when dealing with complex and subtle sentiment expressions. This underscores the need for more specialized self-supervised tasks that are tailored to the specific demands of sentiment analysis.

In summary, while sentiment analysis has made significant progress with the adoption of deep learning methods such as CNNs, RNNs, and Transformers, key challenges persist. Current approaches often struggle to effectively incorporate syntactic information, which is essential for accurately capturing sentiment nuances, particularly in complex languages like Chinese. Furthermore, fine-grained sentiment shifts within sentences and heavy dependence on large labeled datasets for training hinder the scalability and effectiveness of these models. Integrating sensor data presents a promising avenue for providing additional contextual information to enhance model performance. These challenges highlight the need for novel approaches that seamlessly integrate syntactic structures, manage contextual dependencies more effectively, and leverage both self-supervised learning and multimodal data to reduce reliance on labeled data. To address these gaps, our research introduces a model that combines Graph Neural Networks (GNNs) with sentiment embeddings and self-supervised learning strategies, aiming to improve sentiment analysis performance in Chinese microblogs. Compared to GATs, Graph Attention Networks (GATs) utilize attention mechanisms to dynamically weigh the importance of neighboring nodes. While effective, standard GAT models have certain drawbacks: first, Limited Global Context: GATs primarily focus on local neighborhood structures, making them less effective for long-range dependencies in graphs. Second, Computational Bottlenecks: The original GAT scales poorly to large graphs due to the quadratic complexity of attention computation. Third, Improved variants have emerged: GATv2 addresses some of these issues by allowing dynamic attention coefficients, while Graphormer integrates Transformers to enhance global information propagation. Compared to GATs, our proposed method improves global structure modeling while maintaining computational efficiency. By incorporating

graph-based message passing and sequential dependencies, our model achieves superior performance in IoT-related classification tasks.

3. Methodology

In this section, we introduce our proposed approach for sentiment analysis in Chinese microblogs. Our methodology is specifically designed to tackle the unique challenges of this task by integrating self-supervised learning, context-aware sentiment embeddings, and Graph Neural Networks (GNNs). This unified framework enables our model to effectively capture syntactic structures, handle fine-grained sentiment shifts, and mitigate dependence on large labeled datasets. The overall architecture of our model is depicted in Figure 1.

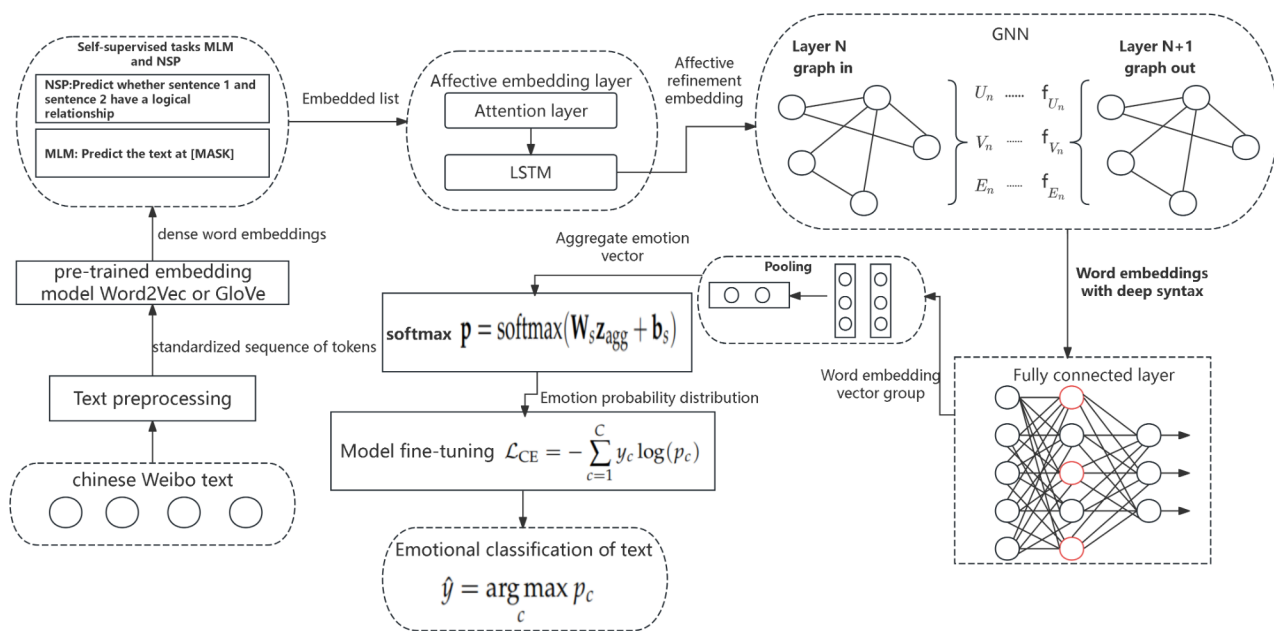


Figure 1. The overview of CE-GNN.

3.1. Overview of the Model Architecture

The proposed model architecture for sentiment analysis in Chinese microblogs comprises four key components: self-supervised pre-training, context-aware sentiment embeddings, syntactic structure modeling with Graph Neural Networks (GNNs), and fine-tuning for sentiment classification. The model first undergoes self-supervised pre-training on a large corpus of unlabeled text, learning contextual word embeddings that encode both semantic and syntactic information through tasks such as Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). These embeddings are then refined by a context-aware sentiment embedding layer, which dynamically adjusts them based on surrounding text to ensure sensitivity to sentiment shifts across different contexts. Subsequently, the refined embeddings are processed using GNNs to capture syntactic dependencies between words, constructing a graph where nodes represent words and edges define their syntactic relationships. This graph-based approach enhances the model's ability to interpret complex sentence structures, particularly in Chinese, where grammatical nuances play a crucial role in sentiment expression. Finally, the model undergoes fine-tuning on a labeled dataset to optimize its performance in sentiment classification tasks, resulting in a robust framework capable of accurately predicting sentiment in Chinese microblogs.

3.2. Self-Supervised Pre-Training

The initial stage of our sentiment analysis model involves self-supervised pre-training on a large corpus of unlabeled Chinese microblog texts. This stage is crucial for learning robust language representations that capture both semantic and syntactic information, providing a foundational basis for the subsequent model components. The flow of our model is depicted in Figure 2.

The input data for this stage consists of raw, unlabeled Chinese microblog texts. These texts are preprocessed through a series of steps: tokenization, lowercasing, and removal of punctuation. The preprocessing produces a standardized sequence of tokens $X = \{x_1, x_2, \dots, x_n\}$, where each x_i represents a token in the sequence.

After preprocessing, the tokenized sequences are converted into dense word embeddings using a pre-trained embedding model such as Word2Vec or GloVe. The embeddings are represented as $E = \{e_1, e_2, \dots, e_n\}$, where each e_n in R^d is a d -dimensional vector capturing the semantic meaning of token x_i . These embeddings serve as the input to the self-supervised learning tasks.

To further enrich these embeddings, we employ self-supervised tasks, namely Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). These tasks are designed to capture both word-level and sentence-level contextual dependencies, providing a robust foundation for downstream tasks. The details of these tasks are as follows:

MLM is designed to help the model learn deep contextual relationships between words by predicting masked tokens based on their surrounding context. Specifically, for a given input sequence $X = \{x_1, x_2, \dots, x_n\}$, a proportion p of tokens is randomly selected and replaced with a special token [MASK], resulting in a modified sequence X_{mask} . For example: Input: "I [MASK] this movie." Target: "I like this movie."

During training, the model is tasked with predicting the original token x_i at each masked position i in M , where M is the set of masked indices. The MLM task is formulated as a cross-entropy loss over the masked tokens:

$$\mathcal{L}_{MLM} = -\frac{1}{|M|} \sum_{i \in M} \log(P(x_i | X_{mask}))$$

where $P(x_i | X_{mask})$ is the predicted probability of the original token x_i given the masked sequence X_{mask} .

The masking process ensures that the model learns bidirectional context. Additionally, to prevent overfitting to the [MASK] token, 80% of the selected tokens are replaced with [MASK], 10% are replaced with random tokens, and the remaining 10% are left unchanged.

NSP is designed to capture sentence-level coherence and contextual relationships between sentences. For this task, the model is provided with pairs of sentences (s_1, s_2) , where s_2 may or may not logically follow s_1 . The task involves classifying whether s_2 is a logical continuation of s_1 . For example: Positive pair: s_1 : "I went to the market". s_2 : "It was very crowded". Negative pair: s_1 : "I went to the market". s_2 : "The weather was hot".

The NSP task is modeled as a binary classification problem, where the objective function is:

$$\mathcal{L}_{NSP} = -[y \log P(\text{follow}) + (1 - y) \log(1 - P(\text{follow}))]$$

where $y = 1$ if s_2 follows s_1 , and $y = 0$ otherwise. The model computes the probability $P(\text{follow})$ by passing the concatenated sentence pair $[CLS]s_1[SEP]s_2[SEP]$ through a classifier head attached to the pooled [CLS] embedding. To construct the dataset for NSP, 50% of the sentence pairs are positive (logical continuation), and 50% are negative (randomly sampled). This balance ensures that the model learns to differentiate coherent sentence pairs from incoherent ones effectively.

The final objective for the self-supervised pre-training combines the losses from MLM and NSP tasks:

$$\mathcal{L}_{pretrain} = L_{MLM} + L_{NSP}$$

By optimizing this combined loss, the model learns rich representations that capture both token-level and sentence-level contextual dependencies. These embeddings, denoted as $\mathbf{E}^* = \{e_1^*, e_2^*, \dots, e_n^*\}$, form the basis for subsequent stages in CE-GNN, including context-aware sentiment embeddings and syntactic structure modeling.

The output of the self-supervised pre-training phase is a set of contextually enriched word embeddings $\mathbf{E}^* = \{e_1^*, e_2^*, \dots, e_n^*\}$ that incorporate both semantic meaning and syntactic structure. These embeddings, which have been enhanced through the self-supervised learning tasks, are now capable of representing more complex language features and relationships. They form the input for the subsequent stage of context-aware sentiment embeddings, where they will be further refined to capture sentiment-specific information within the text.

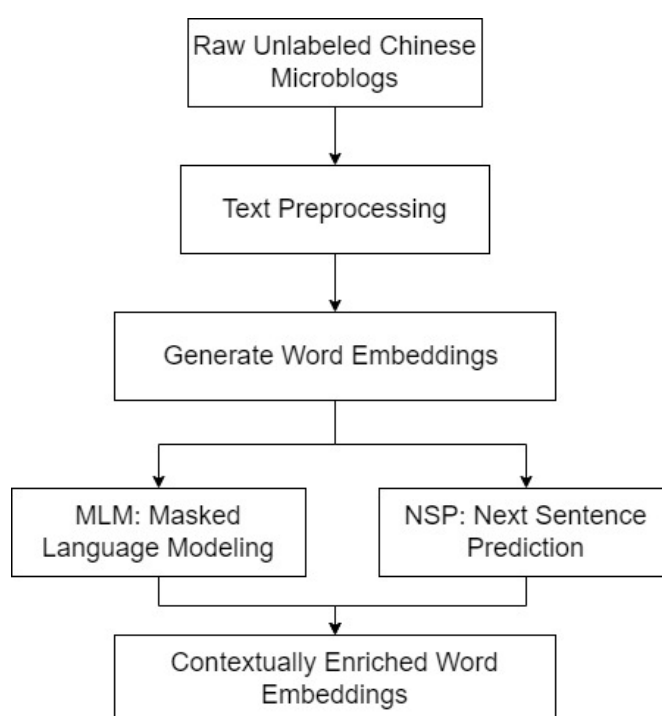


Figure 2. Self-Supervised Pre-Training process in CE-GNN. This figure outlines the steps of text preprocessing, embedding generation, and the application of self-supervised tasks such as Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) to enrich word embeddings.

3.3. Context-Aware Sentiment Embeddings

Following the self-supervised pre-training phase, where the model learns general language representations, the next step is to refine these representations to capture sentiment-specific information. This is achieved through the context-aware sentiment embedding layer, which adjusts the pre-trained embeddings $\mathbf{E}^* = \{e_1^*, e_2^*, \dots, e_n^*\}$ to better reflect the sentiment expressed in different contexts within the text.

The context-aware sentiment embedding layer enhances the embeddings by dynamically incorporating contextual information, which is critical for accurate sentiment analysis. This mechanism allows the model to account for sentiment shifts within sentences, enabling it to focus on contextually relevant features. Given the output from the pre-training phase $\mathbf{E}^*$, this layer leverages mechanisms such as attention and Long Short-Term Memory (LSTM) networks to refine these embeddings.

An attention mechanism is employed to compute a context vector $\mathbf{c}_i$ for each word embedding $\mathbf{e}_i^*$, enabling the model to identify and focus on sentimentally or semantically significant words within the sentence. The attention score e_{ij} between the i -th and j -th words is computed as:

$$e_{ij} = \mathbf{v}^\top \tanh(\mathbf{W}_1 \mathbf{e}_i^* + \mathbf{W}_2 \mathbf{e}_j^*)$$

where $\mathbf{v}$, $\mathbf{W}_1$, and $\mathbf{W}_2$ are learned parameters. The attention weights α_{ij} are then obtained by applying a softmax function:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^n \exp(e_{ik})}$$

The context vector $\mathbf{c}_i$ for each word is computed as a weighted sum of all word embeddings in the sentence:

$$\mathbf{c}_i = \sum_{j=1}^n \alpha_{ij} \mathbf{e}_j^*$$

This context vector $\mathbf{c}_i$ encapsulates the relational dependencies between words and reflects the contextual importance of each word within the sentence. By combining $\mathbf{c}_i$ with the original embedding $\mathbf{e}_i^*$, the refined embedding $\mathbf{e}_i^{\text{refined}}$ is obtained, capturing both the word's intrinsic meaning and its contextual significance:

To further capture sequential dependencies and enhance the model's ability to handle word order and temporal sentiment variations, these embeddings are processed through an LSTM network. The LSTM dynamically updates its cell state $\mathbf{c}_t$ and hidden state $\mathbf{h}_t$ at each time step t at each time step t , as follows:

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{e}_t^* + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{e}_t^* + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{e}_t^* + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c \mathbf{e}_t^* + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t)$$

Here, $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ represent the input, forget, and output gates, respectively, with σ denoting the sigmoid activation function. $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o$, and $\mathbf{W}_c$ are weight matrices for the input embeddings, while $\mathbf{U}_i$, $\mathbf{U}_f$, $\mathbf{U}_o$, and $\mathbf{U}_c$ are weight matrices for the hidden states from the previous time step. The operator $\odot$ denotes element-wise multiplication.

The LSTM outputs a sequence of hidden states $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_n\}$, where each $\mathbf{h}_t$ encodes contextualized sentiment information up to time step t . This dynamic refinement process ensures that the embeddings are both context-sensitive and sentiment-aware, enabling the model to effectively handle sentiment shifts and intricate contextual relationships.

The refined embeddings $\mathbf{H}$ are passed to the next stage of the model, where they are processed by a Graph Neural Network (GNN) to further capture syntactic dependencies. This two-step mechanism of context refinement and syntactic modeling allows CE-GNN to achieve superior performance in sentiment analysis tasks.

3.4. Syntactic Structure Modeling with GNNs

In this phase, the contextually and sentiment-aware embeddings $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_n\}$ are further refined using Graph Neural Networks (GNNs) to capture the syntactic dependencies within the text. This step is crucial for enhancing the model's ability to understand the complex grammatical structures that influence sentiment expression.

The first step in this phase involves constructing a syntactic dependency graph from the input text. Each word embedding $\mathbf{h}_i$ from the previous layer serves as a node in the graph, denoted as v_i . The syntactic dependencies between words, derived from a dependency parser, are represented as directed edges e_{ij} between nodes v_i and v_j . Formally, the graph $G = (V, E)$ is defined as:

$$V = \{v_1, v_2, \dots, v_n\}, \quad E = \{e_{ij} \mid \text{word } i \text{ has a syntactic relation with word } j\}$$

The adjacency matrix $\mathbf{A}$ represents the connections between nodes, where $\mathbf{A}_{ij} = 1$ if there is a syntactic dependency between v_i and v_j , and $\mathbf{A}_{ij} = 0$ otherwise. The graph structure $\mathbf{G} = (\mathbf{A}, \mathbf{H})$ thus encodes both the syntactic relations and the sentiment-aware embeddings.

Once the graph is constructed, a Graph Neural Network (GNN) is applied to propagate information across the nodes, allowing the model to learn enriched, context-sensitive word representations that integrate both syntactic and semantic information. The propagation rule for a GNN layer can be defined as:

$$\mathbf{H}^{(l+1)} = \sigma(\mathbf{A}\mathbf{H}^{(l)}\mathbf{W}^{(l)} + \mathbf{B}^{(l)})$$

Here, $\mathbf{H}^{(l)}$ represents the node features (i.e., word embeddings) at layer l , $\mathbf{W}^{(l)}$ is the weight matrix learned at layer l , and $\mathbf{B}^{(l)}$ is a bias term. The function σ denotes a non-linear activation function such as ReLU. The adjacency matrix $\mathbf{A}$ is used to aggregate information from neighboring nodes, enabling each word's embedding to incorporate information from syntactically related words.

The GNN processes the graph over multiple layers, iteratively refining the node embeddings to capture more complex dependencies. After L layers of GNN, the final node embeddings $\mathbf{H}^{(L)} = \{\mathbf{h}_1^{(L)}, \mathbf{h}_2^{(L)}, \dots, \mathbf{h}_n^{(L)}\}$ are obtained, where each $\mathbf{h}_i^{(L)}$ is a rich representation of the word that integrates both its local context and the syntactic structure of the entire sentence.

The core operation within the GNN is graph convolution, which updates each node's embedding by aggregating the embeddings of its neighbors. For node v_i , the update rule is:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \frac{1}{\sqrt{|\mathcal{N}(i)| \cdot |\mathcal{N}(j)|}} \mathbf{W}^{(l)} \mathbf{h}_j^{(l)}\right)$$

where $\mathcal{N}(i)$ denotes the set of neighbors of node i , and the factor $\frac{1}{\sqrt{|\mathcal{N}(i)| \cdot |\mathcal{N}(j)|}}$ is a normalization term that accounts for the degree of the nodes. This operation allows the GNN to capture not only direct syntactic dependencies but also the influence of a word's broader syntactic context. As shown in Figure 3.

The final output of the GNN is a set of syntactically enriched embeddings $\mathbf{H}^{(L)}$ for each word in the sentence. These embeddings now incorporate deep syntactic information, enabling the model to more accurately interpret the sentiment conveyed by the text. These enriched embeddings are passed to the next stage for sentiment classification, where the model will utilize the combined syntactic and semantic information to make final sentiment predictions.

By incorporating GNNs in this manner, the model attains a more advanced understanding of text, particularly in managing the intricate syntactic structures of Chinese, where word order and dependency relationships play a crucial role in shaping sentiment. This approach capitalizes on our innovation in integrating context-aware embeddings with graph-based syntactic modeling, yielding a robust framework for sentiment analysis.

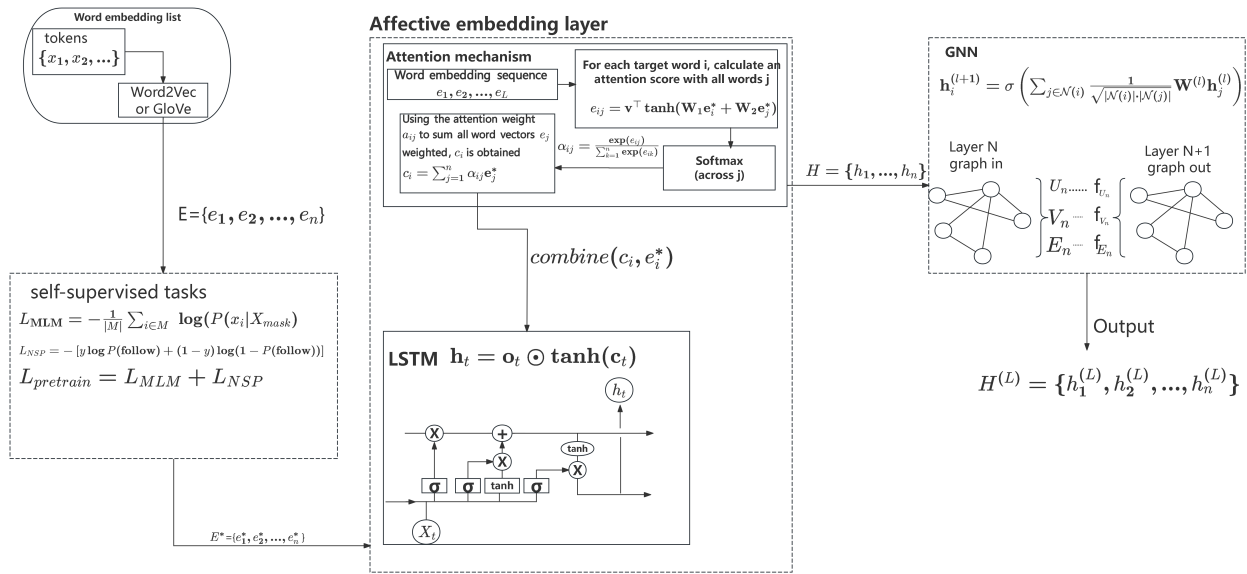


Figure 3. The evolution of word embedding.

3.5. Fine-Tuning and Sentiment Classification

The final stage of our sentiment analysis model involves fine-tuning the network on a labeled dataset to optimize it for sentiment classification tasks. This stage leverages the syntactically enriched embeddings $\mathbf{H}^{(L)} = \{\mathbf{h}_1^{(L)}, \mathbf{h}_2^{(L)}, \dots, \mathbf{h}_n^{(L)}\}$ produced by the Graph Neural Network (GNN) in the previous phase. The goal is to refine these embeddings further and map them to specific sentiment categories, ensuring that the model accurately captures the sentiment nuances expressed in the text.

In the fine-tuning process, the GNN-processed embeddings $\mathbf{H}^{(L)}$ are passed through a fully connected (dense) layer, which serves to transform the high-dimensional embeddings into a space suitable for classification. The transformation can be expressed as:

$$\mathbf{z}_i = \mathbf{W}_{fc} \mathbf{h}_i^{(L)} + \mathbf{b}_{fc}$$

where $\mathbf{W}_{fc}$ is the weight matrix of the fully connected layer, $\mathbf{b}_{fc}$ is the bias vector, and $\mathbf{z}_i$ is the output vector for the i -th word. The fully connected layer allows the model to combine the syntactic and semantic information encoded in $\mathbf{h}_i^{(L)}$ and align it with the specific sentiment classes.

After the fully connected layer, the output vectors $\mathbf{z}_i$ are aggregated (e.g., using an average or max pooling operation) to produce a single vector $\mathbf{z}_{agg}$ that represents the overall sentiment of the entire sentence or document:

$$\mathbf{z}_{agg} = \text{Pooling}(\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n)$$

This aggregated vector $\mathbf{z}_{agg}$ is then passed through a softmax activation function to produce a probability distribution over the sentiment classes:

$$\mathbf{p} = \text{softmax}(\mathbf{W}_s \mathbf{z}_{agg} + \mathbf{b}_s)$$

Here, $\mathbf{W}_s$ and $\mathbf{b}_s$ are the weight matrix and bias vector associated with the softmax layer, and $\mathbf{p}$ is the probability distribution over the sentiment classes (e.g., positive, negative, neutral, or specific emotions).

The model is then fine-tuned on a labeled dataset, where the objective is to minimize the categorical cross-entropy loss:

$$\mathcal{L}_{CE} = - \sum_{c=1}^C y_c \log(p_c)$$

where C is the number of sentiment classes, y_c is the true label (one-hot encoded), and p_c is the predicted probability for class c . This loss function drives the model to adjust its parameters $\mathbf{W}_{fc}$, $\mathbf{b}_{fc}$, $\mathbf{W}_s$, and $\mathbf{b}_s$ during training to maximize classification accuracy.

The final output layer of the model generates the sentiment classification for the input text. This layer produces a probability distribution $\mathbf{p}$ over the sentiment classes, from which the class with the highest probability is selected as the model's prediction:

$$\hat{y} = \arg \max_c p_c$$

where $\hat{y}$ is the predicted sentiment class. The output can indicate whether the sentiment is positive, negative, or neutral, or it can identify specific emotions depending on the classification task. The integration of GNN-based syntactic modeling with context-aware sentiment embeddings allows the model to achieve a deep understanding of the text, accurately capturing and classifying the sentiment expressed. By fine-tuning the model on a labeled dataset, we ensure that the model not only generalizes well but also optimizes its performance for the specific sentiment analysis task at hand. This step is crucial for adapting the model to real-world applications where precise sentiment classification is essential.

In summary, our methodology integrates self-supervised learning for initial language representation, context-aware sentiment embeddings for refined sentiment modeling, and GNNs for capturing syntactic dependencies. This synergistic combination ensures that our model remains robust and effective in sentiment analysis, particularly in the context of Chinese microblogs. The CE-GNN model architecture is designed with flexibility to accommodate multimodal data inputs, facilitating the seamless integration of textual and sensor data for IoT applications. This integration enables CE-GNN to track real-time sentiment shifts across multiple data sources, supporting adaptive responses in intelligent environments.

4. Experiments and Results Analysis

All experiments were conducted using PyTorch 3.8 on a Linux CUDA platform. The syntax-based Graph Convolutional Network (GCN) model was trained on a machine equipped with a 3090 Ti GPU. The complete codebase and detailed instructions for reproducing the experiments are available at the following GitHub repository: <https://github.com/jzzzs2/CE-GNN> (accessed on 10 December 2024).

4.1. Datasets

In this study, we utilized the NLP&CC2013 dataset (Chinese Weibo Sentiment Analysis Task, NLP&CC 2013. Available at: <http://tcci.ccf.org.cn/conference/2013/dldoc/evdata02.zip> (accessed on 15 December 2024)), which was originally designed for a sentiment classification task at the International Conference on Natural Language Processing and Chinese Computing (NLP&CC). To enhance the robustness and generalization ability of our model, we extended this dataset by collecting additional data from Sina Weibo, one of the largest microblogging platforms in China. In addition, a total of 13,220 microblogs were randomly crawled from Sina Weibo. These microblogs were then manually annotated for emotion categories by a team of three human annotators. In instances where discrepancies in annotations occurred, a majority voting mechanism was employed to resolve the conflicts

and assign the final label. This process ensured high-quality, reliable annotations that would contribute to the robustness of the model. The NLP&CC2013 dataset provided the official test set, comprising 2172 entries. The remaining data from NLP&CC2013, combined with our self-annotated microblogs, formed the training set. Specifically, the training set included 4338 entries from NLP&CC2013 and 13,220 entries from the self-annotated Weibo data. Table 1 presents the distribution of emotion categories across the training and test datasets. The emotions are categorized into Happiness, Sadness, Like, Anger, Disgust, Fear, and Surprise, each with a specific number of instances in both the training and testing datasets.

Table 1. Distribution of Emotion Categories in Training and Test Datasets.

Emotion Type	NLP&CC2013	Training Set Self-Annotated	Total	Test Set NLP&CC2013
Happiness	940	2126	3066	370
Sadness	633	2259	2892	385
Like	1284	3029	4313	595
Anger	360	2148	2508	235
Disgust	814	2269	3083	425
Fear	96	366	462	49
Surprise	211	1023	1234	113
Total	4338	13,220	17,558	2172

4.2. Experimental Setup

Table 2 presents the hyperparameter settings used in our sentiment classification model. The selected hyperparameters are optimized to enhance performance across different stages, including self-supervised pre-training, context-aware sentiment embedding, and fine-tuning with labeled data.

Table 2. Hyper-parameters Setting for Sentiment Classification Model.

Hyper-Parameter	Value
Learning Rate (Pre-training)	0.0005
Learning Rate (Fine-tuning)	0.001
Batch Size	64
Epochs (Pre-training)	50
Epochs (Fine-tuning)	30
Dropout Rate (Embedding Layer)	0.3
Dropout Rate (GNN Layers)	0.5
Optimizer	Adam
Embedding Dimension	300
Self-Supervised Tasks	MLM, NSP
GNN Layers	2
GNN Hidden Units	128
Context-Aware Mechanism	BiLSTM
Attention Heads	8
Fine-tuning Batch Size	32
Regularization	L2, $\lambda = 0.01$

During the pre-training phase, a lower learning rate of 0.0005 is employed to ensure stable convergence when training on large unlabeled datasets, whereas a higher learning rate of 0.001 is applied during fine-tuning to facilitate faster convergence on the labeled dataset. The model is trained with a batch size of 64, striking a balance between memory efficiency and training stability. The embedding dimension is set to 300, enabling the capture of rich semantic and contextual information.

For context-aware sentiment embedding, a BiLSTM layer is incorporated to model both past and future contexts, with a dropout rate of 0.3 to mitigate overfitting. Additionally, eight attention heads are utilized to simultaneously focus on multiple contextual aspects.

In the GNN-based syntactic structure modeling phase, the model employs a two-layer GNN architecture with 128 hidden units per layer and a dropout rate of 0.5 to enhance generalization. The Adam optimizer is utilized throughout the training process to dynamically adjust learning rates, thereby improving model performance. Finally, L2 regularization with a coefficient of 0.01 is applied during fine-tuning to prevent excessive model complexity.

We have performed some experiments on the numerical selection of learning Rate, Dropout Rate and number of GNN layers. The results are in Figures 4–6. In this experiment, we performed system tests against multiple hyperparameters to optimize the model performance.

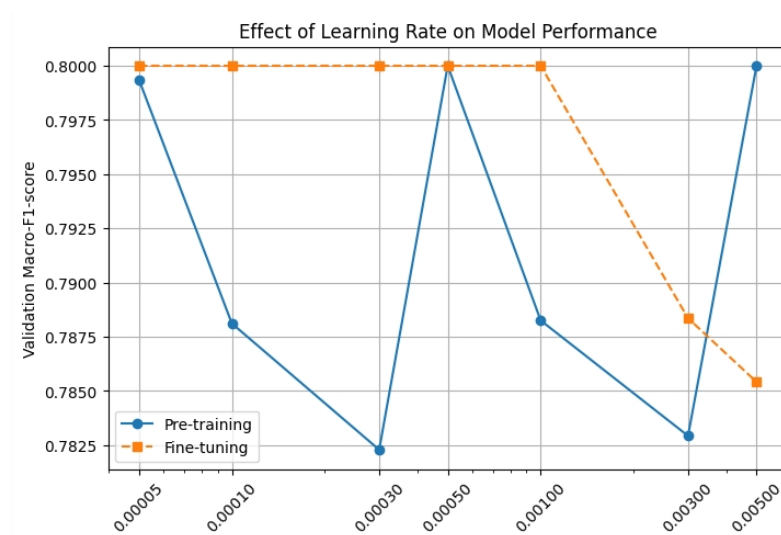


Figure 4. Select learning rate experiment.

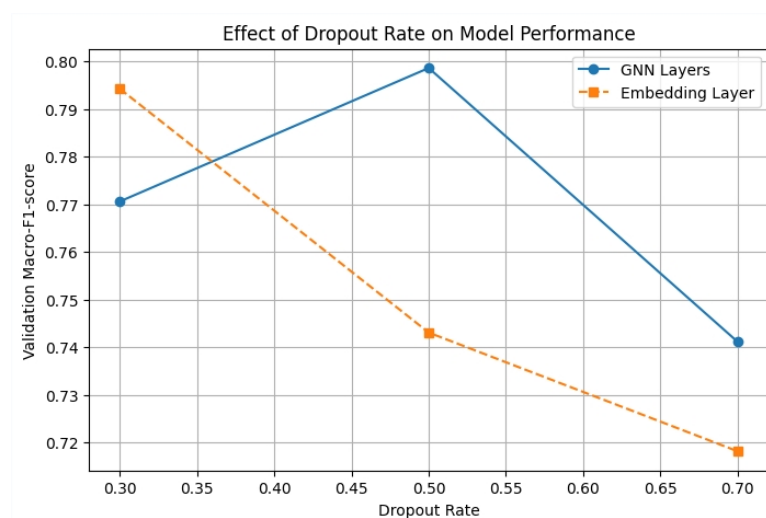


Figure 5. Select learning Dropout Rate experiment.

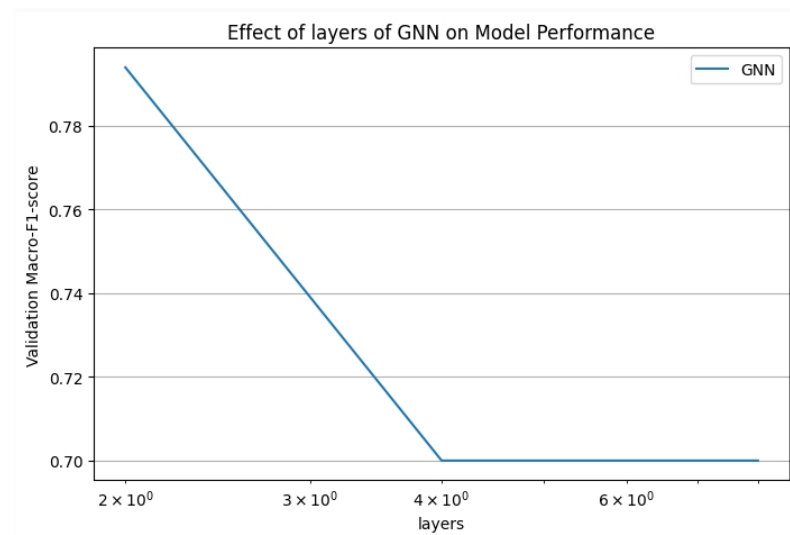


Figure 6. Select GNN layer experiment.

First, in terms of Learning Rate selection, we test a set of learning rates and observe their effect on model convergence. The experimental results show that the convergence is more stable when the learning rate is 0.0005 in the pre-training stage, while the optimal performance can be reached faster when the learning rate is 0.001 in the fine-tuning stage.

Secondly, for Dropout Rate, the experiment finds that the appropriate increase of Dropout to 0.5 in the GNN layer can effectively prevent overfitting, while the Embedding Layer adopts a lower Dropout (0.3) to achieve better model performance.

In terms of Number of GNN Layers, experiments show that two-layer GNN has the best effect on this task, while three-layer GNN may cause information loss due to Over-Smoothing, which leads to performance degradation.

For Batch Size, we refer to relevant literature [38] and carry out experimental analysis. The results show that a batch size of 128 will cause instability in the training process, while a smaller batch size (e.g., 16) will slow down the model convergence. Therefore, we chose 64 for the pre-training phase and 32 for the fine-tuning phase as a compromise to balance training stability and convergence speed.

In addition, we adopted BiLSTM as the context-aware mechanism, and referred to relevant literature [39] to support this choice. The experimental results show that this method is suitable for this task and can effectively capture context information.

Finally, in terms of regularization (L2 regularization term), we refer to literature [40] and experimental verification, and find that the optimal performance can be obtained when λ is 0.01, which can prevent overfitting and does not affect the expressiveness of the model.

4.3. Experimental Results

We selected a range of sentiment classification algorithms as baselines, encompassing both traditional machine learning techniques and advanced neural network models. The performance of our model, compared to the top-performing team in the NLP&CC2013 competition and other baseline models, is summarized in Table 3.

Table 3. Comparison of different models on the NLP&CC2013 testing dataset.

Method	Macro F-Measure	Micro F-Measure
Best team in NLP&CC2013	31.29%	35.21%
E-ESM [41]	35.00%	43.90%
MCNN [41]	34.62%	43.65%
EMCNN [42]	35.17%	44.22%
CNN [43]	67.42%	74.23%
LSTM [44]	68.32%	73.42%
LSTM+CNN (replace GCN with CNN) [45]	69.89%	76.42%
Syntax-based GCN [45]	79.93%	82.32%
Graph WaveNet [46]	80.11%	82.97%
GHGNN [47]	80.08%	81.84%
Graphormer [48]	80.18%	81.24%
GATv2 [49]	79.88%	80.81%
CE-GCN	80.21%	82.93%

Table 3 shows the performance comparison of various sentiment classification models on the NLP&CC2013 testing dataset, using Macro F-measure and Micro F-measure as evaluation metrics. The results indicate that traditional methods like E-ESM and MCNN provide modest improvements over the baseline established by the best team in NLP&CC2013. However, deep learning models such as CNN and LSTM exhibit significantly better performance, with LSTM+CNN further enhancing the results. Notably, the syntax-based GCN model demonstrates the importance of incorporating syntactic information, achieving a considerable performance boost. Graph WaveNet achieves the highest Micro-F1 score, indicating its strong overall classification performance across all categories. In comparison, GHGNN performs slightly worse, with a Micro-F1 score 1.13% lower than that of Graph WaveNet. Although Graphormer incorporates the Transformer mechanism to enhance graph structure modeling, its Micro-F1 performance is still inferior to Graph WaveNet. GATv2 demonstrates the weakest performance, which may be attributed to its focus on learning local attention mechanisms between graph nodes, making it less effective in capturing global structural information compared to Graph WaveNet and Graphormer. Our proposed CE-GCN model surpasses most other approaches, scoring high on both the macro and micro F-measures, highlighting its effectiveness in capturing the fine-grained nuances of sentiment in Chinese microblogs.

The detailed results of CE-GNN for emotion detection, presented in Table 4, demonstrate the model's strong performance across diverse emotion categories. High F1-scores in prominent emotions such as Happiness (0.82), Sadness (0.84), Anger (0.87), and Disgust (0.84) indicate the model's robustness in accurately detecting well-defined emotional expressions. Even for challenging categories like Like, Fear, and Surprise, where subtle nuances or overlapping features may exist, the model maintains acceptable F1-scores of 0.70–0.71. The alignment between the micro-average F1-score (0.82) and the macro-average F1-score (0.80) further supports the robustness of CE-GNN. The micro-average score reflects the model's overall performance across all instances, while the macro-average highlights balanced effectiveness across both frequent and less frequent emotion classes. This balance indicates that the model not only performs well on dominant classes but also generalizes effectively to handle minority categories, which is a critical indicator of robustness in real-world applications. Additionally, the consistency between precision and recall for most categories

demonstrates CE-GNN’s ability to minimize both false positives and false negatives. This balance ensures reliable emotion detection, making the model suitable for complex and nuanced tasks such as sentiment analysis in dynamic contexts, including social media and IoT applications.

Table 4. Detailed result of CE-GNN emotion detection.

	Precision	Recall	F1-Score
Happiness	0.87	0.86	0.82
Sadness	0.86	0.83	0.84
Like	0.78	0.69	0.70
Anger	0.88	0.85	0.87
Disgust	0.82	0.86	0.84
Fear	0.81	0.64	0.71
Surprise	0.72	0.69	0.70
Micro-Average	0.82	0.82	0.82
Macro-Average	0.82	0.78	0.80

4.4. Computational Complexity Analysis and IoT Performance Evaluation

To address the concerns regarding computational complexity and the feasibility of deploying CE-GNN in IoT environments, we conducted a detailed analysis and experimental evaluation. The computational complexity of CE-GNN consists of two key components. First, the Graph Neural Network (GNN) layers process the syntactic dependency graph of the input sentence. The complexity of this component is $O(E + V)$, where E represents the number of edges and V is the number of nodes in the graph (equal to the number of words in the sentence). This ensures efficient propagation of information over graph structures. Second, the context-aware embedding mechanism introduces an additional complexity of $O(d^2)$, where d is the dimension of the embedding. Combining these components, the total computational complexity of CE-GNN is $O(E + V + d^2)$. While this complexity is higher than simpler models such as LSTM ($O(L \cdot d)$), it remains significantly lower than Transformer models ($O(L^2)$, where L is the input sequence length) and is designed to provide a balanced trade-off between computational efficiency and classification performance.

To validate the practical feasibility of CE-GNN in IoT scenarios, we evaluated its inference time on a Raspberry Pi 4 device (Quad-core Cortex-A72, 1.5 GHz, 4 GB RAM). The experiments involved processing input sentences of 50 words and comparing CE-GNN with baseline models, including Transformer, LSTM, and GNN without context awareness. The results are summarized in Table 5.

Table 5. Computational Performance of CE-GNN and Baseline Models in an IoT Environment (Raspberry Pi 4).

Model	Complexity	Inference (ms)	Macro F	Micro F
Transformer	$O(L^2)$	1643	79.36%	81.63%
LSTM	$O(L \cdot d)$	698	75.50%	78.60%
GNN	$O(E + V)$	793	74.30%	77.50%
CE-GNN	$O(E + V + d^2)$	1237	80.21%	82.93%

As shown in Table 5, CE-GNN achieves a slightly higher inference time (1237 ms per sentence) compared to simpler models such as LSTM (698 ms) and GNN without context awareness (793 ms). However, it remains significantly faster than Transformer models (1643 ms). Importantly, CE-GNN provides a substantial improvement in classification performance, achieving a Macro F-measure of 80.21% and a Micro F-measure of 82.93%. These results demonstrate that while CE-GNN incurs slightly higher computational complexity than certain baselines, it offers a superior trade-off between computational efficiency and classification performance, making it a robust and efficient solution for IoT applications requiring both real-time processing and high accuracy. In the meantime, to assess the model's scalability, we estimate its expected performance when deployed on high-performance hardware such as GPUs or TPUs. Based on prior benchmarks of similar architectures:

- GPU acceleration: Running CE-GNN on an NVIDIA A100 GPU is expected to yield a 5× to 10× speedup compared to the Raspberry Pi 4, reducing inference time from 1237 ms to approximately 124–247 ms. This is due to the parallelized execution of matrix multiplications and graph convolutions.
- TPU inference: Specialized accelerators such as Google's TPUv4 can further optimize GNN inference, potentially achieving an order-of-magnitude improvement in processing speed.
- Cloud deployment: In large-scale production environments (e.g., cloud-based services handling millions of requests daily), deploying CE-GNN on multi-GPU clusters or serverless architectures allows efficient handling of high-throughput text classification tasks.

To further enhance the scalability of CE-GNN and large-scale deployment ability, we consider the following optimizations:

- Quantization: Converting model weights to lower precision (e.g., FP16 or INT8) can reduce memory footprint and accelerate inference by up to 4× with minimal loss in accuracy. TensorRT and ONNX Runtime provide efficient quantization techniques.
- Removing redundant weights from the model can decrease computational complexity, leading to a 30–50
- Training a lightweight student model using the output of CE-GNN can significantly reduce model size and improve inference speed while retaining performance.

While CE-GNN exhibits reasonable inference efficiency on resource-limited devices (e.g., Raspberry Pi 4), its scalability can be significantly improved in large-scale production environments using GPU acceleration, model optimization techniques, and distributed computing frameworks. These strategies ensure that CE-GNN can be efficiently deployed for real-world text sentiment classification tasks at scale.

5. Ablation Study

To evaluate the contribution of each component in our proposed CE-GCN model, we conducted an ablation study. Various model variants were created by systematically removing or altering specific components, allowing us to observe the impact on performance. The results of these experiments are summarized in Table 6.

The ablation study results in Table 6 demonstrate the significance of each component in the CE-GCN model. When self-supervised pre-training is omitted, there is a noticeable drop in both Macro F-measure (72.10%) and Micro F-measure (75.25%), highlighting the importance of this phase in learning generalized features from large-scale unlabeled data. Removing the context-aware mechanism further reduces performance, with Macro and Micro F-measures decreasing to 74.30% and 77.50%, respectively. This shows that dynamic adjustment of embeddings based on surrounding context is crucial for capturing fine-

grained sentiment shifts. Replacing the BiLSTM with a standard LSTM also negatively impacts performance, albeit to a lesser extent, with Macro and Micro F-measures dropping to 75.50% and 78.60%. This suggests that bidirectional context is important but not as critical as the context-aware mechanism itself. The absence of GNNs, substituted with a feed-forward neural network (FFNN), results in a significant decline in performance (73.80% Macro, 77.10% Micro), indicating that syntactic structure modeling is essential for understanding complex sentence relationships, especially in Chinese. Substituting GNNs with CNNs improves performance slightly compared to the FFNN variant, achieving 76.20% Macro and 79.30% Micro F-measure, which implies that while convolutional approaches can capture some syntactic information, graph-based methods are superior for this task. The removal of the attention mechanism also impacts performance, reducing the Macro and Micro F-measures to 78.00% and 80.20%. This confirms that the ability to focus on multiple context aspects is beneficial, particularly in sentiment classification. Lastly, the importance of fine-tuning is evident as skipping this step leads to a decrease in performance, with results at 77.50% Macro and 80.10% Micro F-measure. Fine-tuning allows the model to adapt the pre-trained embeddings, specifically to the sentiment classification task, improving overall accuracy.

Table 6. Ablation Study Results for CE-GCN Model.

Model Variant	Macro F-Measure	Micro F-Measure	<i>p</i> (<i>t</i> -Test)
w/o Self-Supervised Pre-training	72.10%	75.25%	0.04
w/o Context-Aware Mechanism	74.30%	77.50%	0.05
w/ Standard LSTM (instead of BiLSTM)	75.50%	78.60%	0.04
w/o GNN (using FFNN)	73.80%	77.10%	0.03
w/ CNN (instead of GNN)	76.20%	79.30%	0.03
w/o Attention Mechanism	78.00%	80.20%	0.06
w/o Fine-Tuning	77.50%	80.10%	0.04
Full CE-GCN (all components)	80.21%	82.93%	

To ensure that the experimental results are not incidental, we conducted 5 to 10 runs for each configuration, collecting results for different hyperparameters and model components. We then applied a *t*-test to compute the *p*-values and assess the statistical significance of the observed differences. The results demonstrate that most components contribute meaningfully to the model's performance. However, the *p*-value for the attention mechanism is 0.6, indicating that its impact is relatively insignificant.

In conclusion, the full CE-GCN model, which integrates self-supervised pre-training, context-aware sentiment embeddings, GNN-based syntactic structure modeling, attention mechanisms, and fine-tuning, achieves the highest performance, with Macro F-measure at 80.21% and Micro F-measure at 82.93%. Each component plays a critical role in the model's ability to capture the intricate nuances of sentiment in Chinese microblogs, and their combined effect leads to state-of-the-art results.

Visualization and Analysis

The attention weights are represented as a matrix where each element indicates the level of attention that one word in the sentence pays to another. Ideally, words that are closely related in context, such as “movie” and “fantastic”, would exhibit higher attention values towards each other. This helps the model capture the nuances of sentiment in text. To visualize this, we generated a simulated attention weight matrix. The attention weights are

not purely random but are adjusted to emphasize the natural language pattern where words are more likely to focus on nearby words. We implemented this by applying an exponential decay function, where attention values decrease as the distance between words increases. This approach mimics the typical behavior observed in real-world attention mechanisms, where a word tends to have stronger relationships with its surrounding words.

In our model, the attention mechanism is integral to capturing the relationships between words in a sentence, particularly in the context of sentiment analysis. By visualizing the attention weights, we can gain deeper insights into how the model interprets and processes the input text, thereby validating the efficacy of our approach.

Figure 7 shows the attention weights generated by our model for a specific sentence. The heatmap reveals several key patterns that align with the intended functionality of the model. Firstly, there is a noticeable concentration of attention along the diagonal, indicating that the model primarily focuses on words and their immediate neighbors, which is crucial for capturing local context and understanding word dependencies. This localized attention is particularly evident in pairs like “movie” and “fantastic”, where the model identifies a strong positive sentiment. Additionally, the heatmap highlights how the model shifts its attention to non-adjacent words when a change in sentiment occurs, such as between “fantastic” and “disappointing”. This behavior demonstrates the model’s capability to handle fine-grained sentiment shifts, an essential feature for nuanced sentiment analysis in Chinese microblogs. The selective attention towards specific words, such as the conjunction “but”, which marks a pivot in sentiment, further indicates that the model effectively incorporates syntactic structure into its processing, a result of the integrated Graph Neural Networks (GNNs) used for syntactic modeling. These attention patterns validate the role of context-aware embeddings in dynamically adjusting the focus based on the surrounding text, ensuring that the model accurately reflects sentiment across different contexts. Overall, the attention visualization underscores the effectiveness of our model’s design in capturing both local and global sentiment cues, thereby enhancing its performance in sentiment classification tasks.

Figure 8 illustrates the t-SNE visualizations of the context-aware embeddings obtained from different configurations of our model. Each subplot corresponds to a different variant of the model, with the embeddings colored according to the seven sentiment classes: Happiness, Sadness, Like, Anger, Disgust, Fear, and Surprise. These visualizations offer insights into how each component of the model contributes to the overall sentiment classification performance.

In the configuration w/o Self-Supervised Pre-training, the sentiment clusters are less defined, with significant overlap between different classes. This indicates that without self-supervised pre-training, the embeddings lack the necessary contextual depth, leading to poorer separation between sentiment classes. The lack of distinct boundaries between classes such as Happiness and Like suggests that the model struggles to differentiate between sentiments that share semantic similarities.

The w/o Context-Aware Mechanism configuration shows some improvement in cluster separation, particularly for classes like Anger and Disgust. However, the embeddings still exhibit considerable overlap, especially between positive and neutral sentiments. The absence of context-aware adjustments in this model limits its ability to capture fine-grained sentiment shifts, resulting in less accurate sentiment representations.

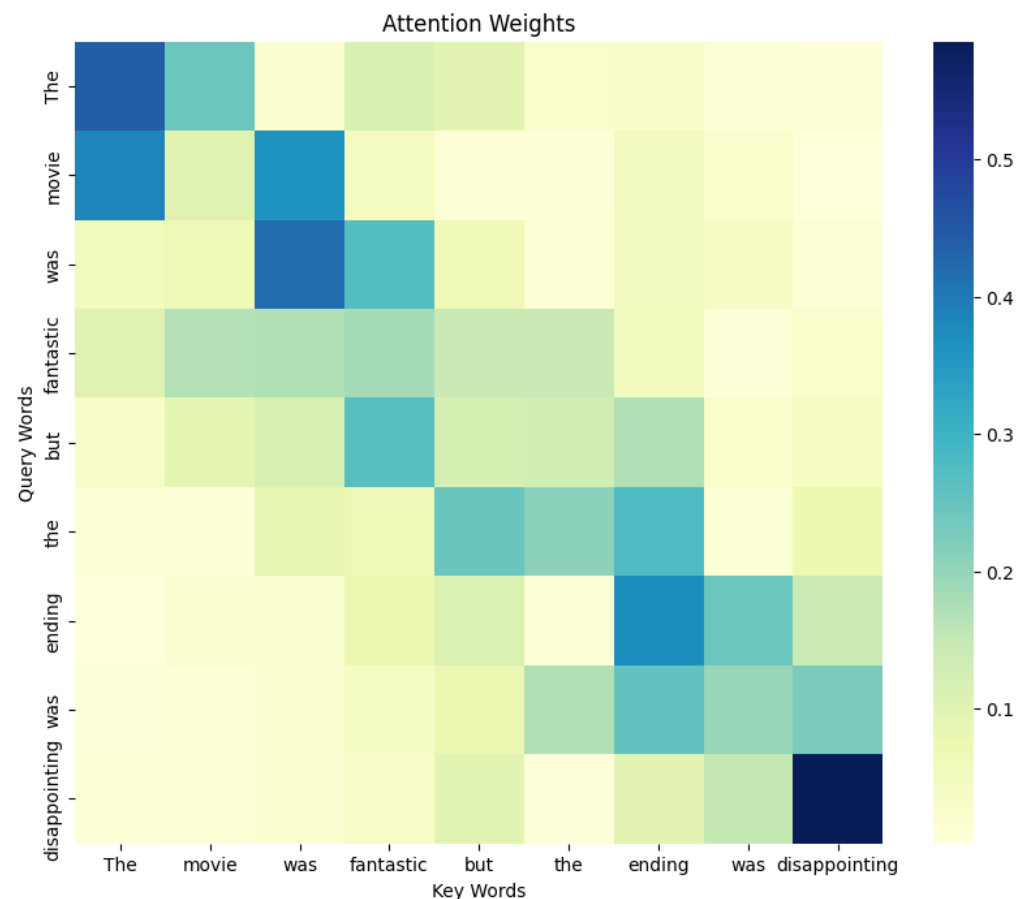


Figure 7. Visualization of the attention weights for a given sentence, illustrating the model's focus on different words during processing.

In the w/o GNN (using FFNN) configuration, the embeddings show better-defined clusters compared to the previous configurations. This improvement highlights the role of the GNN in capturing syntactic dependencies within the text. However, the lack of a GNN leads to a less effective encoding of these relationships, causing overlap in more complex sentiments such as Fear and Surprise, where syntactic structure plays a crucial role.

Finally, the Full CE-GCN (all components) configuration demonstrates the best separation among sentiment classes. The clusters are well-defined, with minimal overlap, indicating that the integration of all proposed components—self-supervised pre-training, context-aware embeddings, and GNN-based syntactic modeling—effectively enhances the model's ability to distinguish between different sentiment classes. This visualization confirms that our complete model configuration not only captures the broad distinctions between sentiment classes but also handles the subtleties of fine-grained sentiment variations. The distinct and compact clusters suggest that the embeddings generated by the full model are rich in both syntactic and semantic information, making them highly effective for downstream sentiment classification tasks.

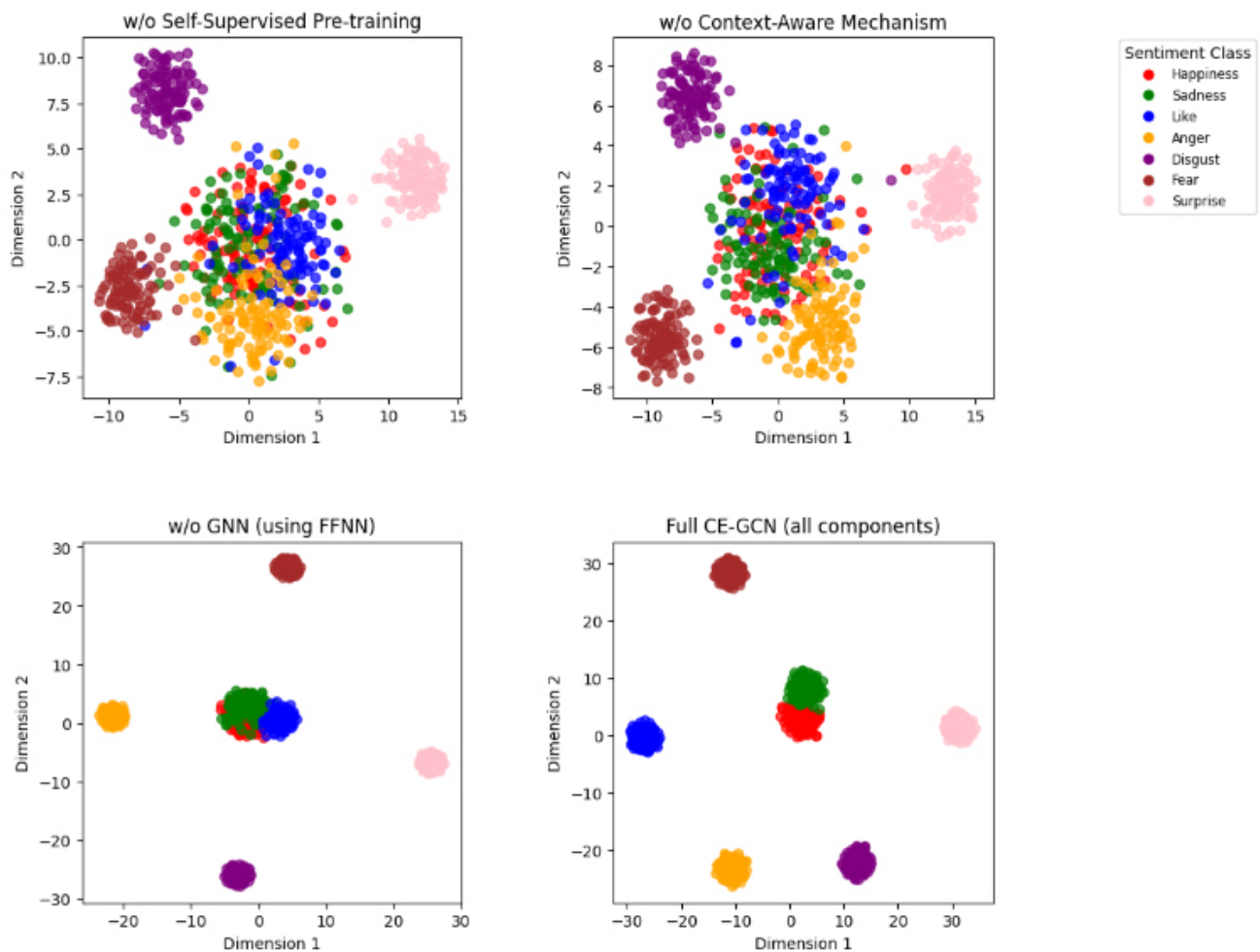


Figure 8. Visualization of t-SNE embeddings for different model configurations, including w/o Self-Supervised Pre-training, w/o Context-Aware Mechanism, w/o GNN (using FFNN), and Full CE-GCN (all components). The embeddings are colored based on sentiment classes: Happiness, Sadness, Like, Anger, Disgust, Fear, and Surprise.

6. Conclusions

In this paper, we propose CE-GNN, a novel framework for fine-grained sentiment analysis in Chinese microblogs that integrates self-supervised learning, context-aware sentiment embeddings, and Graph Neural Networks (GNNs). The proposed method effectively addresses key challenges in sentiment analysis, including capturing subtle sentiment variations, modeling complex syntactic dependencies, and generalizing across diverse contexts. Our main contributions are threefold. First, we design self-supervised learning tasks specifically tailored for sentiment analysis, enabling the model to extract fine-grained features from large-scale unlabeled data. Second, we introduce a dynamic context-aware sentiment embedding layer that adapts to sentiment shifts across different contexts. Third, we develop a GNN-based syntactic modeling framework, where words are represented as nodes and syntactic relationships as edges, allowing the model to capture hierarchical sentence structures effectively. Experimental results, supported by t-SNE visualizations and ablation studies, demonstrate the effectiveness of CE-GNN, with each component playing a critical role in enhancing its ability to detect nuanced emotional cues. Looking ahead, several promising directions for future research emerge. First, incorporating more advanced pre-training tasks specifically designed for sentiment analysis may further enhance the model's ability to capture subtle emotional nuances. Second, extending CE-GNN to other languages and domains would validate its generalizability.

and reveal domain-specific challenges. Finally, integrating external knowledge sources, such as sentiment lexicons or ontologies, could enrich semantic representations and further improve accuracy and interpretability. These advancements could expand the applicability of CE-GNN, establishing it as a robust tool for sentiment analysis across diverse and dynamic real-world scenarios.

In summary, CE-GNN demonstrates strong potential not only for sentiment analysis in Chinese microblogs but also for applications within IoT environments. Future work may focus on expanding CE-GNN's multimodal capabilities to seamlessly integrate sensor data, paving the way for adaptive, emotion-aware smart devices and services.

Author Contributions: Methodology, Z.J.; Software, Z.J.; Validation, Z.J. and Y.Z.; Formal analysis, Z.J.; Resources, Y.Z.; Data curation, Y.Z.; Writing—original draft, Z.J.; Writing—review & editing, Z.J. and Y.Z.; Visualization, Z.J.; Supervision, Z.J.; Project administration, Z.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The datasets and code used in this study are publicly available. The source code for the proposed model can be accessed at <https://github.com/jzzzs2/CE-GNN>. The dataset used for evaluation is available from the NLP&CC 2013 shared task and can be downloaded at <http://tcci.ccf.org.cn/conference/2013/dldoc/evdata02.zip>. If any additional information is required, the authors can be contacted for further details.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fuchs, C. Baidu, Weibo, and Renren: The global political economy of social media in China. In *Culture and Economy in the Age of Social Media*; Routledge: London, UK, 2015; pp. 246–312.
2. Le Han, E. The discourse of Chinese social media: The case of Weibo. In *The Routledge Handbook of Chinese Discourse Analysis*; Routledge: London, UK, 2019; pp. 379–390.
3. Liu, B. *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*; Cambridge University Press: Cambridge, UK, 2020.
4. Mohammad, S.M. Sentiment analysis: Detecting valence, emotions, and other affectual states from text. In *Emotion Measurement*; Elsevier: Amsterdam, The Netherlands, 2016; pp. 201–237.
5. Dong, G.; Liu, H. *Feature Engineering for Machine Learning and Data Analytics*; CRC Press: Boca Raton, FL, USA, 2018.
6. Li, Z.H.; Cui, J.X.; Lu, H.P.; Zhou, F.; Diao, Y.L.; Li, Z.X. Prediction model of measurement errors in current transformers based on deep learning. *Rev. Sci. Instrum.* **2024**, *95*, 044704. [CrossRef] [PubMed]
7. Li, Z.; Cui, J.; Chen, H.; Lu, H.; Zhou, F.; Rocha, P.R.F.; Yang, C. Research Progress of All-Fiber Optic Current Transformers in Novel Power Systems: A Review. *Microw. Opt. Technol. Lett.* **2025**, *67*, e70061. [CrossRef]
8. Yadav, A.; Vishwakarma, D.K. Sentiment analysis using deep learning architectures: A review. *Artif. Intell. Rev.* **2020**, *53*, 4335–4385. [CrossRef]
9. Shukla, A.; Raval, D.; Undavia, J.; Vaidya, N.; Kant, K.; Pandya, S.; Patel, A. Deep Learning Applications in Sentiment Analysis. In *Proceedings of the ICDSMLA 2021: Proceedings of the 3rd International Conference on Data Science, Machine Learning and Applications*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 513–519.
10. Ahmed, A.A.A.; Agarwal, S.; Kurniawan, I.G.A.; Anantadajaya, S.P.; Krishnan, C. Business boosting through sentiment analysis using Artificial Intelligence approach. *Int. J. Syst. Assur. Eng. Manag.* **2022**, *13*, 699–709. [CrossRef]
11. Onan, A. Bidirectional convolutional recurrent neural network architecture with group-wise enhancement mechanism for text sentiment classification. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 2098–2117. [CrossRef]
12. Gaafar, A.S.; Dahr, J.M.; Hamoud, A.K. Comparative analysis of performance of deep learning classification approach based on LSTM-RNN for textual and image datasets. *Informatica* **2022**, *46*, 5. [CrossRef]
13. Tripathi, V.; Joshi, A.; Bhattacharyya, P. Emotion analysis from text: A survey. *Cent. Indian Lang. Technol. Surv.* **2016**, *11*, 66–69.
14. Dashtipour, K.; Gogate, M.; Li, J.; Jiang, F.; Kong, B.; Hussain, A. A hybrid Persian sentiment analysis framework: Integrating dependency grammar based rules and deep neural networks. *Neurocomputing* **2020**, *380*, 1–10. [CrossRef]
15. Acheampong, F.A.; Nunoo-Mensah, H.; Chen, W. Transformer models for text-based emotion detection: A review of BERT-based approaches. *Artif. Intell. Rev.* **2021**, *54*, 5789–5829. [CrossRef]
16. Abdullah, T.; Ahmet, A. Deep learning in sentiment analysis: Recent architectures. *ACM Comput. Surv.* **2022**, *55*, 1–37. [CrossRef]

17. Gruetzmacher, R.; Paradise, D. Deep transfer learning & beyond: Transformer language models in information systems research. *ACM Comput. Surv. (CSUR)* **2022**, *54*, 1–35.
18. Poria, S.; Cambria, E.; Winterstein, G.; Huang, G.B. Sentic patterns: Dependency-based rules for concept-level sentiment analysis. *Knowl.-Based Syst.* **2014**, *69*, 45–63. [\[CrossRef\]](#)
19. Glorot, X.; Bordes, A.; Bengio, Y. Domain adaptation for large-scale sentiment classification: A deep learning approach. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June–2 July 2011; pp. 513–520.
20. Mohan, P. Analyzing Language Mismatch in Self-Supervised Speech Representation Learning. Master's Thesis, Faculty of Information Technology and Communication Sciences, Tampere, Finland, 2024. Available online: <https://trepo.tuni.fi/bitstream/handle/10024/154180/MohanPrabu.pdf> (accessed on 10 December 2024).
21. Sun, C. *Word-Order Change and Grammaticalization in the History of Chinese*; Stanford University Press: Stanford, CA, USA, 1996.
22. James, T. Temporal sequence and Chinese word order. In *Iconicity in Syntax*; John Benjamins: Amsterdam, The Netherlands, 1985; pp. 49–72.
23. Lagrari, F.E.; Elkettani, Y. Traditional and deep learning approaches for sentiment analysis: A survey. *Adv. Sci. Technol. Eng. Syst. J.* **2021**, *6*, 1–7. [\[CrossRef\]](#)
24. Gandhi, A.; Adhvaryu, K.; Poria, S.; Cambria, E.; Hussain, A. Multimodal sentiment analysis: A systematic review of history, datasets, multimodal fusion methods, applications, challenges and future directions. *Inf. Fusion* **2023**, *91*, 424–444. [\[CrossRef\]](#)
25. Kapočiūtė-Dzikiene, J.; Damaševičius, R.; Woźniak, M. Sentiment analysis of lithuanian texts using traditional and deep learning approaches. *Computers* **2019**, *8*, 4. [\[CrossRef\]](#)
26. Can, E.F.; Ezen-Can, A.; Can, F. Multilingual sentiment analysis: An RNN-based framework for limited data. *arXiv* **2018**, arXiv:1806.04511.
27. Basiri, M.E.; Nemati, S.; Abdar, M.; Cambria, E.; Acharya, U.R. ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis. *Future Gener. Comput. Syst.* **2021**, *115*, 279–294. [\[CrossRef\]](#)
28. Karpa, L. Trapping Single Ions and Coulomb Crystals with Light Fields. *arXiv* **2019**, arXiv:1910.00360.
29. Abdallah, H.; Formosa, B.; Liyanaarachchi, A.; Saigh, M.; Silvers, S.; Arslanturk, S.; Taatjes, D.J.; Larsson, L.; Jena, B.P.; Gatti, D.L. Res-CR-Net, a residual network with a novel architecture optimized for the semantic segmentation of microscopy images. *arXiv* **2020**, arXiv:2004.08246. [\[CrossRef\]](#)
30. Kruger, A.Y.; López, M.A.; Yang, X.; Zhu, J. Isolated calmness and sharp minima via Hölder graphical derivatives. *arXiv* **2021**, arXiv:2106.08149. [\[CrossRef\]](#)
31. Kim, N.R.; Kim, K.; Lee, J.H. Sentiment analysis in microblogs using HMMs with syntactic and sentimental information. *Int. J. Fuzzy Log. Intell. Syst.* **2017**, *17*, 329–336. [\[CrossRef\]](#)
32. Xiang, C.; Zhang, J.; Li, F.; Fei, H.; Ji, D. A semantic and syntactic enhanced neural model for financial sentiment analysis. *Inf. Process. Manag.* **2022**, *59*, 102943. [\[CrossRef\]](#)
33. Naseem, U.; Razzak, I.; Musial, K.; Imran, M. Transformer based deep intelligent contextual embedding for twitter sentiment analysis. *Future Gener. Comput. Syst.* **2020**, *113*, 58–69. [\[CrossRef\]](#)
34. Gasde, H.D.; Paul, W. Functional categories, topic prominence, and complex sentences in Mandarin Chinese. *Linguistics* **1996**, *34*, 263–294. [\[CrossRef\]](#)
35. Ridnik, T.; Ben-Baruch, E.; Zamir, N.; Noy, A.; Friedman, I.; Protter, M.; Zelnik-Manor, L. Asymmetric Loss For Multi-Label Classification. *arXiv* **2009**, arXiv:2009.14119.
36. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16 × 16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**, arXiv:2010.11929.
37. Tang, J.; Lu, Z.; Su, J.; Ge, Y.; Song, L.; Sun, L.; Luo, J. Progressive self-supervised attention learning for aspect-level sentiment analysis. *arXiv* **2019**, arXiv:1906.01213.
38. Luo Haoran, Y.Q. Sentiment Analysis of bidirectional long short-term memory networks based on sentiment dictionaries and stacked residuals. *Comput. Appl.* **2022**, *42*, 1099–1107. [\[CrossRef\]](#)
39. Nils Reimers, I.G. Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks. *arXiv* **2017**, arXiv:1707.06799.
40. Smith, L.N. A Disciplined Approach to Neural Network Hyper-Parameters: Part 1—Learning Rate, Batch Size, Momentum, and Weight Decay. *arXiv* **2018**, arXiv:1803.09820.
41. Jiang, F.; Liu, Y.Q.; Luan, H.B.; Sun, J.S.; Zhu, X.; Zhang, M.; Ma, S.P. Microblog sentiment analysis with emoticon space model. *J. Comput. Sci. Technol.* **2015**, *30*, 1120–1129. [\[CrossRef\]](#)
42. Ye, P.; Kumar, J.; Kang, L.; Doermann, D. Unsupervised feature learning framework for no-reference image quality assessment. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1098–1105.
43. Rakhlin, A. Convolutional neural networks for sentence classification. *GitHub* **2016**, *6*, 25.

44. Lee, J.Y.; Dernoncourt, F. Sequential short-text classification with recurrent and convolutional neural networks. *arXiv* **2016**, arXiv:1603.03827.
45. Lai, Y.; Zhang, L.; Han, D.; Zhou, R.; Wang, G. Fine-grained emotion classification of Chinese microblogs based on graph convolution networks. *World Wide Web* **2020**, *23*, 2771–2787. [[CrossRef](#)]
46. Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Zhang, C. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. *arXiv* **2019**, arXiv:1906.00121.
47. Social Recommendation Algorithm Based on Global Heterogeneous Graph Neural Network. In Proceedings of the 18th National Conference on Signal and Intelligent Information Processing and Application, Hefei, China, 30 November 2024.
48. Ying, C.; Cai, T.; Luo, S.; Zheng, S.; Ke, G.; He, D.; Shen, Y.; Liu, T.Y. Do Transformers Really Perform Bad for Graph Representation? *arXiv* **2021**, arXiv:2106.05234.
49. Brody, S.; Alon, U.; Yahav, E. How Attentive are Graph Attention Networks? *arXiv* **2021**, arXiv:2105.14491.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.