International Journal for Housing Science and Its Applications
Publish August 10, 2025. Volume 46, Issue 4 Pages 2725-2738

https://doi.org/10.70517/ijhsa464227

In-depth Research on Chinese Semantic Analysis Technology Empowered by Graph Neural Network Algorithms in the Internet Era

Lu Ye[1,*]
[1] College of Early Childhood Education, Guangxi Vocational & Technical Institute of Industry, Nanning, Guangxi, 530000, China
Corresponding authors: (e-mail: LornaYe0327@163.com).

Abstract Aspect-level sentiment analysis, as a sentiment analysis task, aims to identify the sentiment toward specific aspects or topics mentioned in text. To optimize its performance, which is constrained by internal text information and ignores features such as part-of-speech, dependency relationship types, and syntactic distance in the syntactic dependency graph that could enhance the semantic information of aspect words, this paper combines the syntactic dependency graph with a graph neural network model. By leveraging external knowledge to enhance the graph attention network, we propose a graph neural network-based aspect-level text sentiment analysis method focused on semantic enhancement. We collect theme-related comment text data from various social media platforms to create a dataset tailored for aspect-level sentiment analysis composite tasks. By comparing with multiple baseline methods, we analyze the advantages of the proposed model in Chinese semantic analysis applications. The proposed model, SEGCN, achieves semantic analysis accuracy rates of 92.23% (PTS-1), 83.45% (PTS-2), and 90.76% (PTS-3) across all datasets, outperforming other baseline methods.

Index Terms aspect-level sentiment analysis, graph neural network, external knowledge, syntactic dependency, Chinese semantic analysis

I. Introduction

With the development and widespread adoption of the internet, not only has the way Chinese is used and expressed changed, but it has also brought about a series of new vocabulary and linguistic phenomena, presenting new challenges for Chinese semantic analysis. First, Chinese is inherently ambiguous, with some words having multiple pronunciations or meanings, highlighting the complexity of the language. Second, the internet age has spawned a wealth of online slang and internet-specific terminology, such as “躺平” (lying flat), “绝绝子” (absolutely amazing), “YYDS” (the best), and “666” (excellent), which are widely used online and have gradually infiltrated everyday spoken language [1], [2]. Furthermore, in the internet environment, many words have developed additional meanings, such as “fan” (originally referring to a type of food, but now primarily referring to a fan of a celebrity), while Chinese dialects have become written in the online environment [3], [4]. These linguistic phenomena have made people's expressions more flexible and diverse, while also posing new challenges to the traditional norms and usage of Chinese [5], [6]. Additionally, text on social media has diverse characteristics, such as non-structured features like emojis and multi-modal characteristics like comments [7], [8].

Semantic analysis is one of the core tasks in natural language processing, with the primary goal of extracting meaning from text, understanding semantic information in natural language, and further processing and analyzing it [9]. Unlike traditional grammar-based analysis methods, semantic analysis focuses on extracting deeper meanings from text, with widespread applications including sentiment analysis, question-answering systems, and speech recognition [10]-[12]. With the rapid development of natural language processing technology, text semantic analysis has evolved from the dictionary rule matching and statistical probability calculation methods of the 20th century to the current mainstream intelligent analysis algorithms based on machine learning and deep learning. Chen et al. [13] utilized Chinese dependency syntactic analysis and latent semantic analysis to extract core words and perform semantic analysis on Chinese sentences, respectively, and developed a new latent semantic analysis algorithm capable of analyzing sentences with similar contexts but opposite meanings, thereby mitigating syntactic blind spots. Liu et al. [14] used deep learning to perform semantic analysis on COVID-19-related stigmatizing content on social media during the early stages of the COVID-19 outbreak, providing assistance to government regulation. Zhang et al. [15] shared a semantic analysis platform for Chinese and English in a network environment, namely NLPIR-Parser, which combines a Chinese word segmentation system and features word segmentation, part-of-speech tagging, and new word recognition.

Graph neural networks (GNNs) are deep learning models for processing graph-structured data. They update node representations by aggregating neighborhood information, enabling effective capture of topological structures and node feature information within graphs [16]. Xu et al. [17] combined dictionary-level and chapter-level knowledge from ancient Chinese books with GNN support, using a graph attention mechanism to input knowledge into the model, thereby enhancing semantics and optimizing named entity recognition methods. Wang et al. [18] designed a symbol-signal integration network based on a multivariate GNN. This network combines an attention mechanism to integrate syntactic and semantic features, enhancing the syntactic and semantic analysis of sentiment analysis based on aspects, thereby improving the accuracy of sentiment analysis.

This paper proposes related foundational technologies, including graph convolutional neural networks. To meet the experimental requirements of aspect-level sentiment analysis, a dataset tailored for aspect-level sentiment analysis composite tasks was established. Thematic data was collected from social media comments on platforms like Meituan and Douyin, abnormal texts were processed, duplicate data was removed, and the text data was preprocessed before annotation. We analyzed graph-based dependency syntactic analysis methods to optimize syntactic dependency part-of-speech analysis and syntactic distance, and to obtain more aspect-level semantic information features. We proposed a graph convolutional neural network-based aspect-level text sentiment analysis model enhanced with semantic information. We set up a Chinese semantic analysis experimental environment, adjusted parameter settings, and analyzed the model's Chinese semantic analysis accuracy and F1 score.

II. Related technologies

II. A. Graph Convolutional Neural Networks

Ⅰ Implementation Process of Graph Convolutional Neural Networks

Graph convolutional neural networks consist of an input layer, a convolutional layer, an activation function, and an output layer. The specific steps are as follows [19], [20]:

Step 1: Input a dataset in a non-Euclidean space.

Step 2: In convolutional layer 1, each node performs a convolutional operation with each of its neighboring nodes, and the convolutional results are used to update the node.

Step 3: Pass through an activation function such as ReLU.

Step 4: Enter convolution layer 2, where each node performs a convolution operation with each of its neighboring nodes, and the results of the convolution are used to update the node.

Step 5: Pass through an activation function such as ReLU again.

Step 6: Repeat the above steps until the convolutional layer reaches the desired number of layers, obtaining the local output function or overall output function of the graph convolutional neural network.

Ⅱ Convolution Algorithm Principles

The primary purpose of convolution in graph convolutional neural networks is to extract spatial features from topological spaces. There are two methods for feature extraction: one is based on spatial domain convolution, and the other is based on frequency domain convolution.

In a linear time-invariant (LTI) continuous-time system, the process of calculating the convolution integral can be described as follows: first, the input signal is decomposed into impulse functions using the sampling theorem. Then, the impulse responses of each impulse function in the system are calculated. Finally, these impulse responses are summed to obtain the zero-state response of the input signal in the system. Equation (1) represents the convolution integral of functions $f_1(t)$ and $g_1(t)$:

$$f_1(t) * g_1(t) = \int_{-\infty}^{+\infty} f_1(\tau) g_1(t-\tau) d\tau \tag{1}$$

In LTI discrete systems, discrete signals are primarily digital pulse signals, so it is easy to decompose the input signal into unit sequences. The process of calculating the convolution sum can be described as follows: based on the known unit sequence response of the system, calculate the response of each sequence in the system. Then add the responses of each sequence together to obtain the zero-state response of the input signal. Equation (2) represents the convolution sum of two sequences $f_1(k), g_1(k)$ as:

$$f_1(k) * g_2(k) = \sum_{i=0}^{k} f_1(i) g_2(k-i) \tag{2}$$

Mapping LTI systems to neural networks, $f_1(t), f_1(k)$ can be regarded as features in neural networks and as signals in signal processing. $g_1(t), g_2(k)$ can be regarded as kernels in the neural network and as filters in signal processing.

GSP graphical signal processing method, which treats graphs as signals and uses signal processing methods (Laplace matrix, Fourier transform) to analyze graphs and extract their features.

Convolution based on the frequency domain primarily utilizes the graph Fourier transform to achieve convolution. By deriving the Laplace operator in the frequency domain from the graph's Laplace matrix, and analogizing it to convolution in Euclidean structured data in the frequency domain, the formula for graph convolution is derived. The Fourier transform can transform a function from the time domain to the frequency domain, where time-domain convolution equals frequency-domain multiplication.

Defining the Fourier transform, the expression is:

$$F(j\omega) = \int_{-\infty}^{+\infty} f(t)e^{-j\omega t}dt \tag{3}$$

Fourier transform, expressed by the formula:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} F(j\omega)e^{j\omega t}d\omega \tag{4}$$

Time-domain convolution theorem:

$$f_2(t) \leftrightarrow F_2(j\omega), f_3(t) \leftrightarrow F_3(j\omega), f_1(t) * f_2(t) \leftrightarrow F_1(j\omega)F_2(j\omega) \tag{5}$$

Frequency domain convolution theorem:

$$f_1(t)f_2(t) \leftrightarrow \frac{1}{2\pi}F_1(j\omega) * F_2(j\omega) \tag{6}$$

In Equations (3), (4), (5), and (6), ω is the angular frequency, and $F(j\omega)$ is called the spectral density function of $f(t)$. $f(t)$ is the original function of $F(j\omega)$, and $e^{-j\omega t}$ is the basis function.

The specific steps for calculating the convolution of $f_1(t)$ and $f_2(t)$ are as follows:

Step 1: Use formula (4) to transform the Fourier transforms of $f_1(t)$ and $f_2(t)$ into the frequency domain to obtain $F_1(j\omega), F_2(j\omega)$.

Step 2: According to the time-domain convolution theorem, convolution in the time domain can be converted to multiplication in the frequency domain. From equation (5), we obtain: $f_1(t) * f_2(t) \leftrightarrow F_1(j\omega)F_2(j\omega) = G(j\omega)$.

Step 3: Convert $G(j\omega)$ to the time domain using the inverse Fourier transform (3) to obtain the convolution result $g(j\omega)$.

III Graph Convolutional Neural Networks

In computer science, graphs are mainly used to describe the relationships between data. Based on whether the edges in a graph are directed or undirected, graphs can be divided into two categories: undirected graphs and directed graphs.

A simple undirected graph is represented as: $G = \{V, E, A\}$, where V denotes the set of vertices in the undirected graph, and $|V| = n$ indicates that the graph has n nodes. E denotes the set of edges in the undirected graph, and A denotes the adjacency matrix, indicating the connections between nodes in the undirected graph.

The degree matrix (D), adjacency matrix (A), and Laplacian matrix (L) of an undirected graph are generated as follows:

Define D as an $N \times N$ degree matrix, expressed as:

$$D(i,j) = \begin{cases} d_i & i = j \\ 0 & otherwise \end{cases} \tag{7}$$

Define A as the adjacency matrix of $N \times N$, expressed as:

$$A(i,j)=\begin{cases}1, & if\ x_i \sim x_j\\ 0 & otherwise\end{cases} \tag{8}$$

So, the formula for the Laplace matrix is:

$$L = D - A \tag{9}$$

In GCN, the normalized Laplacian matrix is used more frequently, with the formula being:

$$L^{sys} = D^{\cdot 1/2} A D^{\cdot 1/2} \tag{10}$$

Based on the above derivation, the formula for calculating the feature extraction of the first layer of GCN is:

$$H^{(1)} = \delta(\tilde{A} X W_o) \tag{11}$$

Among them:

$$\tilde{A} = D^{-1/2} A D^{-1/2} \tag{12}$$

D is the degree matrix formula (7), $W_0 \in R^{D\times D}$ is a weight matrix, δ is an activation function, such as ReLu $\delta(x)=\max(0,x)$. The feature extraction formula for the multi-layer neural network GCN is:

$$H^{(l+1)} = \delta(\tilde{A} X^{(l)} W_i) \tag{13}$$

l is the number of layers in the GCN network.

II. B.Aspect-level sentiment analysis

Aspect-based Sentiment Analysis (ABSA) aims to predict the sentiment polarity of a given aspect in a sentence. It can be further divided into several subtasks: aspect term extraction, aspect category extraction, aspect sentiment classification, and aspect category classification. Aspect category extraction identifies implicit aspect categories present in a sentence. Aspect category classification determines the sentiment polarity of one or more implicitly occurring aspect categories in a sentence.

The main research directions of ABSA involve the identification of four key elements: aspect terms, aspect categories, opinion terms, and sentiment polarity. Given the Chinese sentence “Today's weather is so refreshing,” the corresponding sentiment elements are “weather,” “good,” “refreshing,” and “positive,” where “weather” and “refreshing” are explicitly expressed in the sentence, and “weather” and “positive” belong to predefined categories.

II. C.Language characteristics

II. C. 1) Part of speech

Part-of-speech tagging is a fundamental research area in natural language processing, primarily involving the classification of vocabulary. Typically, part-of-speech tagging is applied to words within sentences, with the following seven common categories: noun (NN), verb (VB), pronoun (PN), adjective (JJ), adverb (RB), preposition (IN), and conjunction (CC).

In aspect-level sentiment analysis tasks, part-of-speech information is highly beneficial for determining the sentiment polarity of aspect words and understanding sentence semantics. Since aspect words are typically nouns or noun phrases, while opinion words are usually adjectives or adverbs, part-of-speech tagging can support aspect-level sentiment analysis tasks. On one hand, incorporating part-of-speech information into word embeddings enables models to better extract word feature representations. On the other hand, incorporating part-of-speech representations can better distinguish word meanings, as some words have multiple meanings. Specifically, different part-of-speech categories can represent different meanings. For example, the word “check” as a noun means “bill,” while as a verb it means “to inspect.” Therefore, incorporating part-of-speech representations into word representations helps the model better understand sentence semantics.

II. C. 2) Dependency Syntax Analysis

This paper investigates Chinese semantic computation methods based on joint dependency analysis, taking into account the linguistic characteristics of the Chinese language.

I. Dependency Grammar Theory

Dependency grammar theory is defined as a syntactic system centered on verbs that constructs dependency (or subordination) relationships between words. It is further defined as a structural grammar that represents the deep semantic structure of a sentence as its surface syntactic structure, describing the relationship between dependency grammar and semantic structure [21].

The four construction rules of dependency grammar constrain the dependency syntactic structure of sentences:

a. A sentence has only one head word, and all other words are dependent on this word.

b. Except for the head word of the sentence, all other words in the sentence are directly dependent on a particular word.

c. No word in the sentence can be dependent on two or more words simultaneously.

d. If word A directly depends on word B or word B directly depends on word A, then word C between A and B must directly or indirectly depend on A or B.

These rules constrain the dependency grammar structure into a tree structure with a single root node, connected, and mappable. Researchers typically use words as nodes and directed arcs as dependency relationships between two words to draw the dependency structure of a sentence. Dependency grammar describes the relationships between words, directly extracting information from the lexicon, and is more closely related to morphological analysis. At the same time, dependency syntax expresses the deep semantic structure of a sentence in the form of syntactic relationships, describing semantic structure information, and is also more closely related to semantic analysis. Therefore, dependency grammar has gradually become a research focus in the field of syntactic analysis in recent years.

II Dependency Analysis Algorithms

(1) Transition-based Dependency Analysis Algorithms

The dependency analysis algorithm based on transitions includes a set of analysis states $C=\{c\}$ and a set of transition actions $T=\{t \mid t: C \rightarrow C\}$. Each state $c \in C$ represents a (partially constructed) dependency structure graph G, and each transition action $t \in T$ represents an operation in the process of constructing a dependency tree, guiding the dependency analysis from one state to the next. For a given sentence $x=\{w_0, w_1, w_2, ..., w_n\}$ (where w_0 denotes the added root node, typically denoted by "S", as the starting point of the analysis), the transition-based analysis method ultimately yields a sequence of states $C_{0,m}=(c_0, c_1, ..., c_m)$. c_0 represents the initial state of the analysis, typically an edge-less graph $G_0=\{V, \varnothing\}$. c_m represents the final state of the analysis, denoting the dependency parse tree of the final sentence $G_m=\{V, A_m\}$. For two adjacent states c_{i-1} and c_i in this state sequence, they are obtained under the guidance of a transition action $t_i=c_{i-1} \rightarrow c_i$, Therefore, this sequence of states can be associated with a unique sequence of transition actions T_1, where $m=(t_1, t_2, ..., t_m)$.

(2) Chinese integrated dependency analysis algorithm

The most obvious feature of Chinese compared to English is that it does not have word segmentation tags. Therefore, when performing dependency analysis, it is necessary to first perform word segmentation and other operations to obtain lexical information.

However, Chinese word segmentation is a highly challenging problem. Unlike English, which uses 26 letters as its basic units, the number of commonly used Chinese characters exceeds 3,000, thereby containing rich semantic information. Additionally, Chinese has a high information entropy, and the words that can be formed in different contexts are not entirely consistent. Therefore, to achieve high-precision word segmentation, it is necessary to fully utilize the syntactic and semantic information of the sentence.

The set of transformation operations in Chinese integrated dependency analysis is shown in Figure 1. The set of transformation operations in Chinese integrated dependency analysis includes four transformation operations: Left-Reduce, Right-Reduce, Shift/POS, and Append.

$$\text{Left-Reduce}\ \frac{[\cdots,w_i,w_j]_\sigma \quad [\cdots]_\beta}{[\cdots,w_j]_\sigma \quad [\cdots]_\beta \quad A\cup(j,i,l)}$$

$$\text{Right-Reduce}\ \frac{[\cdots,w_i,w_j]_\sigma \quad [\cdots]_\beta}{[\cdots,w_j]_\sigma \quad [\cdots]_\beta \quad A\cup(j,i,l)}$$

$$\text{Shift}\ \frac{[\cdots]_\sigma \quad [c_j,\cdots]_\beta}{[\cdots,s_i]_\sigma \quad [\cdots]_\beta \quad s_i=c_j}$$

$$\text{Append}\ \frac{[\cdots,s_i]_\sigma \quad [c_{j,}\cdots]_\beta}{[\cdots,s_i']_\sigma \quad [\cdots]_\beta \quad s_i'=s_i+c_j}$$

Figure 1: A collection of moving actions of Chinese integration dependence analysis

III. Dataset for aspect-level sentiment analysis composite tasks

This paper uses comment data from multiple social media platforms (including Meituan, Douyin, Xiaohongshu, and Weibo) as samples to ensure data diversity, covering different topics, emotional tendencies, and text styles. A dataset for aspect-level sentiment analysis experiments, namely PTS, is constructed. PTS is used in the training of the model proposed in this paper, improving the model's ability to handle Chinese text and sentences with multiple entity words, while also providing a data foundation for subsequent research.

III. A. Collection of comment text data based on social media

This paper uses Python to collect relevant comment texts from Meituan, Weibo, and Xiaohongshu in 2025 regarding “food delivery wars,” “TV show recommendations,” and “travel guides,” forming the PTS-1, PTS-2, and PTS-3 datasets, respectively.

First, the required libraries were imported: requests: used to send HTTP requests to retrieve web page content. BeautifulSoup: used to parse HTML content. csv: used to write comment data into CSV files.

Next, the target website URL and request header information are defined: url: the target website URL, which can be replaced as needed. headers: request header information, including user agent information, used to simulate a browser sending requests. Send HTTP requests to retrieve web page content: use the requests.get() method to send HTTP requests, passing in the target website URL and request header information. Store the response content in the response object and extract the HTML content into the html_content variable.

Then, use BeautifulSoup to parse the HTML content: Create a BeautifulSoup object and pass in the HTML content and parser type. Use the find_all() method to find the tags and class names containing comment information in the HTML, and store the results in the comments variable.

Finally, create a CSV file and write the comment data to it.

The data collection environment for this article is shown in Table 1.

Table 1: Data collection environment

System environment	Windows 10
Browser	Chrome 110.0.5481.178
Python version	3.7.0
Development tool	Pycharm2024.1.3
Data marker tool	ROST CM

III. B. Data processing for web comment samples

The raw data obtained needs to undergo data preprocessing, which involves the following steps:

Remove HTML tags and CSS styles: For text from web pages or other HTML formats, use the Beautiful Soup library in Python to process HTML tags and CSS styles, and use regular expressions to remove excess whitespace and special characters, retaining only plain text content.

Handling abnormal text: To simulate real-world scenarios where a sentence contains multiple entity words, during the data preprocessing stage, first merge the original sample files into a single-line format. Split the data by blank lines and merge the sentences, with one sentence per line. Then merge multiple sample files into one for easier reading. Additionally, the maximum length of text in the takeout review data is set to 256 characters. Since extremely long texts are rare, manual removal is sufficient.

Handling duplicate data: The pandas library in Python is used to handle duplicate data in documents.

Text preprocessing: Stopword removal is performed on the text data. Removing stopwords reduces the number of words in the text data, thereby lowering the dimensionality of the text data and accelerating subsequent processing and analysis speeds.

III. C. Data annotation for model composite tasks

In this paper, based on the requirements of aspect-level sentiment analysis experiments, two subtasks need to be completed: entity word recognition and corresponding sentiment classification. Therefore, entity word annotation and sentiment annotation must be performed on the text data.

This paper uses the BIO annotation method to annotate entities in the text. In the BIO annotation scheme, B-ASP represents the starting position of the aspect. The ROST CM software is selected for sentiment annotation of the text data.

Next, the annotated dataset is divided into training, validation, and test sets, with an 80%-10%-10% ratio. After obtaining the data labels, the data is integrated and randomly shuffled. Originally, there were over 20,000 entries, but after processing, there were 18,000 entries in total. Among these, over 9,000 positive entries were labeled as 1, 6,000 negative entries were labeled as 2, and 3,000 neutral entries were labeled as 0.

IV. Chinese semantic analysis technology based on graph neural networks

IV. A. Dependency Analysis Based on Graph Neural Networks

This paper adopts a graph-based dependency parsing method, which can consider global information for dependency analysis decisions. Recent analysis shows that this method outperforms transition-based dependency analysis methods in terms of performance.

Below, we take the sentence $P=\{p_0,p_1,\cdots,p_N\}$ as an example to introduce the dependency parsing module in detail. Following standard practice, the root node identifier "<root>" is added at the beginning of each sentence as p_0.

First, the input word sequence is converted into a numerical vector representation, consisting of three parts: pre-trained word vectors, randomly initialized word vectors, and part-of-speech tag vectors. Let $e(p_i)\in R^d$ denote the pre-trained word vector, $e^{'}(p_i)\in R^d$ denote the randomly initialized word vector, $e(pos_i)\in \mathbb{R}^{d_{pos}}$ denotes the part-of-speech tag vector, where d_{pos} is the embedding dimension for part-of-speech tags. These three components are updated during training. Finally, the representation of each word is calculated using the following equation, where "$\oplus$" denotes concatenation. That is:

$$x_i=(e(p_i)+e^{'}(p_i))\oplus e(pos_i) \tag{14}$$

To capture long-range contextual information in sentences, deep bidirectional LSTM (BiLSTM) is used to learn word representations in sentences. The hidden state at time i (corresponding to the i th word) is represented as follows:

$$h_i=BiLSTM(x_i,\vec{h}_{i+1},\vec{h}_{i-1},\theta) \tag{15}$$

Among them, $\overleftarrow{h_i}$ and $\overrightarrow{h_i}$ are the hidden representations of the forward and backward LSTM at time i respectively. θ is the parameter in BiLSTM.

The calculation is performed using a double affine attention mechanism, as shown in the following equation. Here, s_{ij} represents the score for $p_j\to p_i$, with a higher score indicating a greater likelihood of forming $p_j\to p_i$. That is:

$$r_i^{dep}=MLP^{(dep)}(h_i) \tag{16}$$

$$r_j^{head} = MLP^{(head)}(h_j) \tag{17}$$

$$s_{ij} = r_i^{dep} U r_j^{head} + r_j^{head} u \tag{18}$$

Here, U denotes the weight matrix, and u denotes the bias term.

$s_i = \left[s_{i0}, \cdots, s_{ij}, \cdots, s_{iN}\right](j \in \{0,1,\cdots,N\})$, s_{ij} is the score for the dependency relationship $p_j \to p_i$, where s_{i0} is used to measure the probability of the i th word becoming the root ROOT. Subsequently, normalization is performed using formula (19) to obtain the probability distribution α_i, which is used to construct the dependency probability matrix α from $\alpha_i (i \in \{0,1,\cdots,N)$. Finally, in the final section, cross-entropy is used as the loss function, as shown in equation (20):

$$\alpha_i = softmax(s_i) \tag{19}$$

$$L_0 = -\sum_{k=1}^{N_s} \sum_{i=1}^{N_k} \beta_i^k \log(\alpha_i^k) \tag{20}$$

The objective function L_0 represents the cross-entropy loss, N_s represents the number of sentence pairs in a batch, and N_k represents the number of words in the k th P sentence. β_i^k is the unique hot code representation of the true core word of the i th word in the k th P sentence.

IV. B. Semantically Enhanced Graph Neural Networks for Sentiment Analysis at the Text Level

A sentiment analysis model based on graph neural networks can utilize external information such as syntactic dependency information and common-sense knowledge modeled by graph neural networks to enrich the semantic meaning of target aspect words.

Joint modeling of common-sense knowledge and syntactic knowledge can effectively extract various external knowledge information, but this model ignores features such as part-of-speech, dependency relationship type, and syntactic distance in the syntactic dependency graph that are helpful for enhancing the semantic meaning of aspect words. To address this issue, this paper proposes a semantic-enhanced graph convolutional neural network aspect-level text sentiment analysis model (SEGCN). It constructs a weighted syntactic dependency graph based on syntactic dependency relationship types, syntactic distance, and relative position relationships, while also constructing an external common-sense knowledge subgraph to extract external knowledge information. By utilizing multiple features, it enhances the semantic information of target aspect words.

IV. B. 1) Task Definition

The aspect-level sentiment analysis task addressed in this paper is defined as follows: For a review text $S = \{w_1, w_2, \cdots, w_{n-1}, w_n\}$, the target aspect words contained in the text are represented as $A_{Aspect} = \{w_{start}, w_{start+1}, \cdots, w_{end-1}, w_{end}\}$, where start and end denote the starting and ending indices of the target aspect word sequence A_{Aspect} in sentence s. where A_{Aspect} consists of one or more words. The aspect-level sentiment analysis task involves determining the sentiment polarity represented by the aspect sequence A_{Aspect} in the text s. The sentiment polarity $C \in \{0,1,2\}$ represents "negative," "neutral," and "positive" sentiment polarities, respectively.

IV. B. 2) Specific Modules

(1) Word Embeddings and Bi-GRU Layer

To extract contextual semantic information from review text and target aspect words, the word embedding vectors of text s and target aspect A_{Aspect} are input into the BiGRU layer. The extracted hidden layer states are represented as $H^c = (h_1^c, h_2^c, \cdots, h_t^c, \cdots, h_n^c)$ and $H^a = (h_1^a, h_2^a, \cdots, h_t^a, \cdots, h_k^a)$.

(2) Distance Encoding Module

The syntactic distance vector can be represented as $D = (d_1, d_2, \cdots, d_i, \cdots, d_n)$, where d_i is the syntactic distance between the context word w_i and the target aspect word. The formula for calculating the syntactic distance weight of each word in a sentence is as follows:

$$l_i = 1 - \frac{d_i}{2d_{\max}} \tag{21}$$

Among them, $d_{\max}$ is the maximum value in the syntactic distance.

The relative distance between the context word w_i and the target aspect word a_i also has a different impact on the emotional polarity of the target aspect word. Therefore, it is necessary to calculate the relative distance weight of the text words as follows:

$$l_i^{'} = \begin{cases} 1 - \dfrac{start+1-i}{n}, 1 \le i < start+1 \\ 0, start+1 \le i \le start+k \\ 1 - \dfrac{i-start-k}{n}, start+k < i \le n \end{cases} \tag{22}$$

In this case, start is the starting index of the target aspect word. k is the length of the target aspect word sequence, and n is the length of the text. i is the position of the word in the text. Then, a new distance weight representation is generated by combining the syntactic distance weight and the relative distance weight:

$$q = (q_1, q_2, \cdots, q_n) \tag{23}$$

$$q_i = \frac{l_i + l_i^{'}}{2} \tag{24}$$

Finally, introduce q_i to update the hidden layer representation $H^c = (h_1^c, h_2^c, ..., h_n^c)$, and the updated sentence hidden state $H^d = (h_1^d, h_2^d, \cdots, h_t^d, \cdots, h_n^d)$ contains both word position information and semantic information. The calculation formula is as follows:

$$h_i^d = q_i h_i^c \tag{25}$$

(3) Knowledge Embedding Module

The feature representation of knowledge subgraph nodes is obtained by searching the affective-space file to obtain the initial feature representation $H^{aff} = (h_1^{aff}, h_2^{aff}, \cdots, h_t^{aff}, \cdots, h_m^{aff})$, where m is the number of knowledge nodes in the knowledge subgraph. The overall feature representation of the nodes in graph G^k is denoted as H^k. The set of edges E^k consists of the connection relationships between text word nodes and knowledge nodes, as well as the connection relationships between internal knowledge nodes. The adjacency matrix A^k of the knowledge graph is defined as follows:

$$A_{ij}^k = \begin{cases} 1, \text{Both i and } j \text{ are knowledge nodes}, e_{ij} \in E^k \\ 1, i = j \\ 0, \text{other} \end{cases} \tag{26}$$

(4) Graph Convolutional Neural Network Layer

The graph convolutional neural network processes weighted syntactic dependency graphs and knowledge subgraphs to extract feature representations of target aspects that integrate syntactic dependency types and external common-sense information, thereby enhancing the semantic information of target aspect words.

For the weighted syntactic dependency graph G^s, its adjacency matrix A^{ws} and feature representation H^d are used as inputs to the graph convolutional neural network. The node update formula is as follows:

$$h_i^{s(l)} = \sigma(\sum_{j=1}^{n} A^{ws}(W^{(l)} \cdot h_j^{d(l-1)} + b^{(l)})) \tag{27}$$

Among them, $W^{(b)}$ is the weight matrix of the l th layer. $b^{(b)}$ is the bias of the l th layer, and n is the number of words in the text.

For the knowledge subgraph G^K, its adjacency matrix A^K and feature representation H^k are used as inputs to the graph convolutional neural network. The node update formula is as follows:

$$h_i^{k(l)} = \sigma(\sum_{j=1}^{n+m} A^k(W^{(l)} \cdot h_j^{k(l-1)} + b^{(l)})) \tag{28}$$

Among them, m is the number of knowledge node words.

(5) Attention layer [22]

The model in this paper uses the contextual semantic features of target aspect words $H^a = (h_1^a, h_2^a, \cdots, h_t^a, \cdots, h_k^a)$ and the text features output by the graph convolutional neural network $H^s = (h_1^s, h_2^s, \cdots, h_t^s, \cdots, h_n^s)$ using an attention mechanism, extracting features important for target aspect words from text features that integrate syntactic structure information and distance encoding information. The calculation formulas are shown in Equations (29) to (31), and the output of the attention layer is denoted as H^{sa}. That is:

$$\beta_t = \sum_{i=1}^{k} h_i^s (h_i^a)^T \tag{29}$$

$$\alpha_t = \frac{\exp(\beta_i)}{\sum_{i=1}^{n} \exp(\beta_i)} \tag{30}$$

$$H^{sa} = \sum_{i=1}^{n} \alpha_i h_i^s \tag{31}$$

(6) Emotion classification layer

For the feature representation H^k that integrates emotional common sense knowledge, retain the feature representation H^{ka} of target-related words. Concatenate it with the output H^{sa} of the attention mechanism as the final feature representation r. Use the Softmax function to perform emotional polarity classification as follows:

$$r = Concat(H^k a, H^{sa}) \tag{32}$$

$$y = Softmax(wr + b) \tag{33}$$

The loss function used by the model is the multi-class cross-entropy loss function, where cross-entropy measures the difference between the probability distributions of the predicted values and the true values. The model in this paper has three label values. Let the probability that the i th sample is predicted to belong to the k th label be p_{ik}, and $\sum_{k=1}^{3} p_{ik} = 1$, where 3 represents the number of category labels and N represents the number of samples. Then, the loss function of the model is as follows:

$$L = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{3} y_{ik} lb p_{ik} \tag{34}$$

V. Semantic analysis experiment based on graph neural networks

V. A. Experimental setup

The hardware environment for this experiment was Linux 5.8.0-44-generic x86_6, Anaconda3-5.1.0-Linux-x86_64, NVIDIA 2080Ti graphics card, Pytorch deep learning framework, and Python 3.7.

V. A. 1) Parameter Settings

In the experiment, SEGCN uses Glove word vectors to initialize word embeddings with a dimension of 500. Additionally, 50-dimensional part-of-speech (POS) embeddings and 50-dimensional position embeddings are added, where position embeddings represent the relative position of each word with respect to the reference word in the sentence. Then, the word embeddings, POS embeddings, and position embeddings were concatenated as the input word representation. The concatenated word representation was fed into a Bi LSTM network, with the hidden state dimension set to 100. All sentences were parsed by a syntactic dependency parser, and the edge embedding vectors were initialized to 100 dimensions.

The model's batch size is set to 32, and the number of training epochs is set to 100. Additionally, a dropout function is applied to the hidden states of the Bi LSTM, with a dropout rate of 0.1. The model is trained using an Adam optimizer with a learning rate of 0.001 to optimize the parameters.

V. A. 2) Baseline Method

This paper conducts comparative experiments between the SEGCN model and other aspect-level sentiment analysis models. The models compared in the experiments are as follows:

(1) The TD-LSTM model concatenates the target string with word embedding vectors and inputs them into the LSTM to represent the semantic relationship between the target and the context.

(2) The ATAE-LSTM model incorporates aspect information into word embeddings and uses an attention mechanism to capture important parts of the sentence.

(3) The MemNet model utilizes a multi-hop memory network to focus on aspect words in sentences based on word and position information.

(4) ASGCN employs a multi-layer graph convolutional neural network after the LSTM to extract aspect-specific basic features, and uses masking and attention mechanisms to obtain aspect-specific high-level features.

(5) The BERT model directly processes sentence text using the pre-trained BERT model and fine-tunes aspect-level sentiment classification as a downstream task.

(6) InterGCN constructs aspect-specific graphs and aspect relationship graphs to extract aspect-specific features and aspect interaction features, respectively.

(7) SenticGCN integrates external sentiment knowledge from SenticNet to construct a graph neural network, enhancing the dependency graph of sentences.

(8) CDT first learns sentence feature representations using a bidirectional long short-term memory network, then further enhances the embeddings using a graph convolutional neural network operating on the sentence dependency tree.

(9) BiGCN encodes corpus-level word co-occurrence information to construct a conceptual hierarchy on syntactic and lexical graphs for sentiment prediction.

(10) DGEDT proposes a dual transformer network based on syntactic dependency graphs, combining the word representations learned by the transformer with the graph representations of the dependency tree.

V. B. Experimental Results

The accuracy rates of different models on different datasets are shown in Figure 2.

The MemNet model achieved the lowest Chinese semantic analysis accuracy rate of 71.42% on the PTS-2 dataset. The performance of all comparison models on the PTS-2 dataset was poor, possibly because the PTS-2 dataset is a Chinese text dataset for “TV series recommendations,” which contains TV series characters, background settings, and other elements that are more challenging for Chinese semantic analysis methods.

The proposed SEGCN model achieves semantic analysis accuracy rates of 92.23% on the PTS-1 dataset, 83.45% on the PTS-2 dataset, and 90.76% on the PTS-3 dataset. On the PTS-1 dataset, the SEGCN model outperforms the best baseline method, DGEDT, by 0.27%.

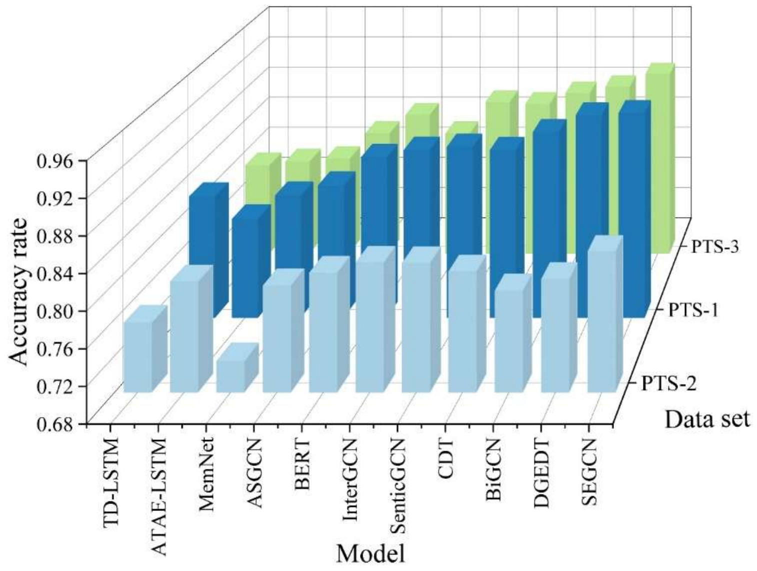

Figure 2: The accuracy of different models on different data sets

The F1 scores of different models on different datasets are shown in Figure 3.

The F1 scores of the TD-LSTM and ATAE-LSTM models are lower than the average F1 score across all datasets.

The F1 scores of the SEGCN model are higher than those of the TD-LSTM model by 14.39% (PTS-3), 18.32% (PTS-2), and 14.02% (PTS-1), respectively.

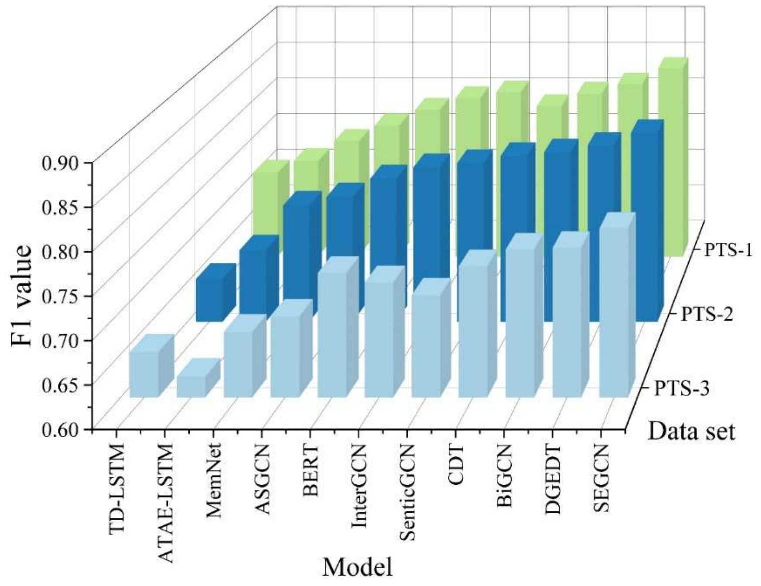

Figure 3: F1 values on different data sets

V. C. Ablation analysis

To verify the impact of different components on model performance, ablation experiments were conducted, with the results shown in Table 2.

The experimental results indicate that the model's accuracy significantly decreases when positional encoding is absent. In the PTS-2 dataset, the F1 score for w/o P is the lowest at 72.58%. This result strongly demonstrates the critical role of positional information in model training. Specifically, positional encoding enables the model to accurately identify the relative positional relationships between different parts of the input data, which is crucial for the model to comprehensively and deeply understand the data content. It helps the model capture the sequential and structural information in the text, thereby improving the accuracy of sentiment analysis.

Additionally, when the attention module is removed, the model's performance on all four datasets decreases to varying degrees. When external sentiment knowledge was removed, the model's performance also generally declined. This result further validates the significant value of external sentiment knowledge in improving model performance. External sentiment knowledge is a quantitative representation of the sentiment orientation of sentiment words in different contexts, providing more precise and detailed sentiment information. By introducing external sentiment knowledge, the model can more accurately judge the sentiment polarity and intensity of sentiment words in the text, which is of great significance for improving the accuracy of sentiment analysis.

Table 2: Ablation experiment results

Model	PTS-1		PTS-2		PTS-3	
	Acc	F1	Acc	F1	Acc	F1
w/o Position coding	0.8212	0.7473	0.7906	0.7258	0.8432	0.7366
w/o Attention layer	0.8356	0.7514	0.7973	0.7311	0.8455	0.7424
w/o External emotional knowledge	0.8374	0.7552	0.8084	0.7339	0.8469	0.7467
SEGCN	0.8528	0.7669	0.8359	0.7445	0.8571	0.7782

V. D. Effect of number of layers

Since SEGCN can be stacked on top of L layers, to investigate the impact of the number of layers L on the final performance, the experiment considered different numbers of layers from 1 to 8 and examined the corresponding accuracy of SEGCN on three datasets.

The results of varying the number of SEGCN layers are shown in Figure 4. From the experimental results, it can be observed that performance varies with the increase in the number of SEGCN layers. However, SEGCN experiences a performance decline after multiple layers, and the SEGCN model achieves optimal performance when L = 4. At L = 4, the SEGCN model achieves the highest accuracy of 90.26% and the highest F1 score of 85.54%.

For this observation, two aspects were considered:

(1) When L=1, SEGCN only considers first-order syntactic relationships on the syntactic dependency tree, and edge vectors can only aggregate features of first-order adjacent nodes. This is insufficient to obtain multi-hop contextual word sentiment features on the syntactic dependency tree and lacks more syntactic information. For some complex sentence structures, it is impossible to obtain the corresponding opinion word expressions.

(2) When the number of layers increases, SEGCN operating on the syntactic dependency tree often leads to overly smooth nodes, making it difficult to distinguish between node representations. This also tends to cause gradient vanishing issues, thereby impairing model performance.

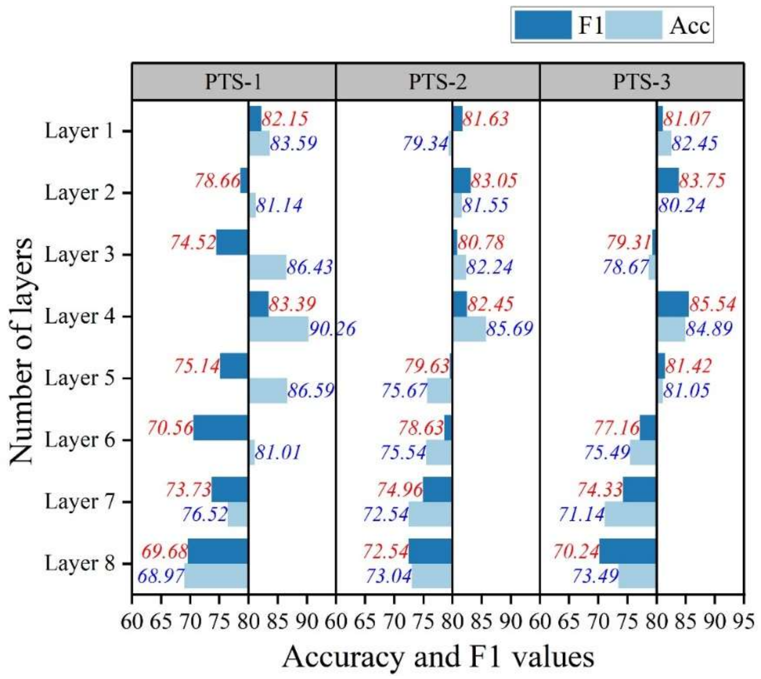

Figure 4: Model SEGCN layer number changes

VI. Conclusion

This paper enhances Chinese text sentiment analysis technology based on graph neural networks by improving semantic analysis capabilities. A Chinese dataset tailored for aspect-level sentiment analysis is developed, and combined with parameter settings, it is compared with baseline methods to evaluate the experimental performance of the graph neural network-based aspect-level text sentiment analysis technology with enhanced semantic capabilities.

In Chinese semantic analysis experiments, the model SEGCN achieved semantic analysis accuracy rates of 92.23%, 83.45%, and 90.76% across all datasets. These results are 12.03% higher than the MemNet model (PTS-2) and 0.27% higher than the DGEDT model (PTS-1). Ablation analysis shows that position encoding, attention modules, and external sentiment knowledge all have a certain impact on the model's performance. The model design in this paper improves traditional graph neural network-based semantic analysis technology.

References

[1] G'affarovna, Z. B. (2025). YOUNG SLANG AND INTERNET LINGUISTICS THE EVOLATION OR DIGITAL COMMUNICATION. Continuing education: international experience, innovation, and transformation, 1(2), 26-28.
[2] Sun, Y. T., & Gao, W. C. (2024). A study on Internet buzzwords in the last four years based on conceptual integration theory. J. Lit. Art Stud, 14(8), 711-717.
[3] Lee, T. Y. (2020). Language Change in Progress: Chinese Internet Lexicon and the Symbolic Meaning in the Harmonious Society. Nowa Polityka Wschodnia, 27(4), 132-145.
[4] Zhang, Y., & Ren, W. (2024). From hǎo to hǒu–stylising online communication with Chinese dialects. International Journal of Multilingualism, 21(1), 149-168.
[5] He, Y. (2024). Exploring Linguistic Innovations Within Chinese Internet Slang in Language and Social Media. Journal of Linguistics and Communication Studies, 3(1), 29-34.

[6] Glushkova, S. Y., & Voronina, M. K. (2018). Trends in the development of Chinese internet language. Modern Journal of Language Teaching Methods, 8(9), 37-43.

[7] Zhang, H., Cai, Y., Ren, H., & Li, Q. (2022). Multimodal topic modeling by exploring characteristics of short text social media. IEEE Transactions on Multimedia, 25, 2430-2445.

[8] Velkova, I. (2023). Unstructured social media data processing with artificial intelligence. Industry 4.0, 8(2), 65-67.

[9] Zhang, B., Yan, H., Wu, J., & Qu, P. (2024). Application of Semantic Analysis Technology in Natural Language Processing. Journal of Computer Technology and Applied Mathematics, 1(2), 27-34.

[10] Araque, O., Zhu, G., & Iglesias, C. A. (2019). A semantic similarity-based perspective of affect lexicons for sentiment analysis. Knowledge-Based Systems, 165, 346-359.

[11] Liu, Y., Zhang, H., Zong, T., Wu, J., & Dai, W. (2023). Knowledge base question answering via semantic analysis. Electronics, 12(20), 4224.

[12] Srinivasan, N., & Balasundaram, S. R. (2023, October). Enhancing Automatic Speech Recognition in Air Traffic Communication Through Semantic Analysis and Error Correction. In The International Conference on Recent Innovations in Computing (pp. 779-794). Singapore: Springer Nature Singapore.

[13] Chen, M., Li, S., Fan, M., & Wu, J. (2024, November). A Potential Semantic Analysis Model that Incorporates Chinese Dependent Syntactic Analysis. In 2024 8th Asian Conference on Artificial Intelligence Technology (ACAIT) (pp. 182-189). IEEE.

[14] Liu, L., Cao, Z., Zhao, P., Hu, P. J. H., Zeng, D. D., & Luo, Y. (2022). A deep learning approach for semantic analysis of COVID-19-related stigma on social media. IEEE Transactions on Computational Social Systems, 10(1), 246-254.

[15] Zhang, H., Miao, J., Liu, Z., Wesson, I. L., & Shang, J. (2020, June). NLPIR-Parser: making Chinese and English semantic analysis easier and complete. In 15th International Conference on the Statistical Analysis of Textual Data.

[16] Guo, W., Fang, Y., Huang, C., Ou, H., Lin, C., & Guo, Y. (2022). HyVulDect: a hybrid semantic vulnerability mining system based on graph neural network. Computers & Security, 121, 102823.

[17] Xu, Y., Mao, C., Wang, Z., Jin, G., Zhong, L., & Qian, T. (2024). Semantic-enhanced graph neural network for named entity recognition in ancient Chinese books. Scientific Reports, 14(1), 17488.

[18] Wang, H., Qiu, X., & Tan, X. (2024). Multivariate graph neural networks on enhancing syntactic and semantic for aspect-based sentiment analysis. Applied Intelligence, 54(22), 11672-11689.

[19] Sakonporn Noree,Willmer Rafell Quinones Robles,Young Sin Ko & Mun Yong Yi. (2025). Leveraging commonality across multiple tissue slices for enhanced whole slide image classification using graph convolutional networks. BMC Medical Imaging,25(1),230-230.

[20] Oumaima Guendoul,Maryem Zobi,Hamd Ait Abdelali,Youness Tabii,Rachid Oulad Haj Thami & Omar Bourja. (2025). Capturing spatio-temporal patterns of falls individuals using efficient graph convolutional network model. Applied Intelligence,55(11),825-825.

[21] Osborne Timothy & Groß Thomas. (2021). Bracketing paradoxes: A dependency grammar analysis in terms of morph catenae. Poznan Studies in Contemporary Linguistics,57(3),397-428.

[22] Jainish G R & Alwin Infant P. (2024). Attention layer integrated BiLSTM for financial fraud prediction. Multimedia Tools and Applications,83(34),80613-80629.