Long Text Classification Model Based on Transformer Sliding Window and Threshold Optimization

Jin Pan[1], Yang Chen[1], Chunlu Zhao[1], Yang Liu[1], Jie Chu[2]*

[1] National Computer Network Emergency Response Technical Team/Coordination Center of China, China
[2] Institute of Network Technology (Yantai), China
jinpancert@163.com, cylovehehe@163.com, chunluzhao@cert.org.cn,
liuyangcert@163.com, chujie@int-yt.com

Abstract

In long-text classification tasks, the main issue we face is that traditional text classification methods lack the capability to analyze complex contextual information and implicit semantics, resulting in poor classification performance. To address the shortcomings of existing long-text classification models—such as poor accuracy, low efficiency, inadequate dynamic adjustment, and poor threshold adaptability—we propose a Transformer-based sliding window threshold optimization long-text classification model. We investigate an automatic classification and dynamic partitioning method for long-text semantic analysis. The approach utilizes context-aware fusion of semantic information from long texts to improve the accuracy of automated long-text classification. Finally, we employ an error feedback mechanism to dynamically adjust the classification threshold, achieving optimized threshold settings for long-text content classification. The experiment shows that this model surpasses the baseline model in accuracy and F1 performance on the dataset and exhibits good convergence speed, validating its effectiveness.

Keywords: Text classification, Transformer, Context-aware fusion, Threshold optimization, Sliding window

1 Introduction

In the digital age, the volume of text data on the internet is exploding, including a vast amount of long-text information such as news articles, blog posts, forum discussions, and comments. These texts may contain specific content, such as targeted information or negative remarks, which, if not controlled, can have detrimental effects on society and individuals. Therefore, effectively identifying and classifying specific content from a large volume of text has become increasingly important.

Traditional text classification methods mainly rely on simple keyword matching or shallow machine learning models. While these methods work reasonably well for short texts, they often perform poorly with long texts, especially those containing complex context and implicit semantics. Long texts typically include richer information and more complex structures, making it difficult to capture deep meanings with keyword matching alone. Traditional machine learning models struggle with long-distance dependencies and complex contextual information. Deep learning methods address issues such as intricate feature engineering in traditional algorithms by extracting effective low-dimensional dense features from high-dimensional sparse text representations, thus better handling various complex and challenging tasks [1-2]. Deep learning has become one of the primary methods for solving complex problems, with algorithms like Recurrent Neural Networks (RNNs) [3-5], Long Short-Term Memory networks (LSTMs) [6], Self-Attention Networks (SANs) [7], Transformers [8], and Bidirectional Encoder Representations from Transformers (BERT) [9] widely used in natural language processing.

Compared to short text classification, classifying long texts such as scientific resources, media articles, web information, and literature is more complex. Such texts are often composed of multiple sentences or paragraphs, containing a large number of words that reflect the text's category features. As the length of the text increases, traditional machine learning algorithms and memory-based models like RNNs struggle to maintain contextual information over very long texts. Consequently, some text classification models handle long texts by only considering fixed-length segments to represent the global text features. While this approach is simpler, it overlooks the remaining contextual information and cannot extract complete semantic features from long texts with uneven distribution of meaningful information. To address the issue of long-distance dependencies and to mine comprehensive information from long texts, two classic architectures have evolved in long text classification: the Transformer architecture and the hierarchical architecture. The core of the Transformer architecture is the self-attention mechanism [10], which has a time and space complexity of $O(n^2)$, meaning that as the input sequence length increases, the complexity grows geometrically. As a result, some Transformer variants focus on optimizing the self-attention mechanism to reduce the complexity of attention calculations for handling long text inputs.

*Corresponding Author: Yang Chen; E-mail: cylovehehe@163.com
DOI: https://doi.org/10.70003/160792642025032602008

2 Related Works

Recent advancements in deep learning technology have rapidly progressed in text classification. Deng et al. [11] proposed an attention-based BiLSTM model that integrates CNN and gating mechanisms (ABLG-CNN) for text classification. To address sentences that may involve other topic information in long texts, they introduced a gating mechanism to weight the features output by BiLSTM and CNN, aiming to obtain fusion features beneficial for classification. Li et al. [12] constructed an LSTM_CNN hybrid model for news text classification. It first uses LSTM to learn the long-term dependencies of the text, then designs a shallow convolutional structure to further extract semantic features of the text, and finally applies max-pooling to filter important features for classification. Mao et al. [13] proposed a text classification model based on multi-level semantic features. Initially, they incorporated category-related coefficients into TF-IDF and frequency concentration coefficients into CHI, using an improved algorithm to extract keyword semantic features. They then used TextCNN to extract local semantic features from BiLSTM and pay attention to symmetric channels and global semantic information. Finally, they integrated three semantic features for text category prediction. Ai et al. [14] introduced a heterogeneous attention network method based on a multi-semantic transmission framework. They designed a flexible heterogeneous information graph to model long texts by extracting information, and a multi-semantic transmission framework capable of extracting semantic and structural information from the constructed heterogeneous information graph through specific structural semantic degrees. Wang et al. [15] proposed a domain-specific long-text classification model based on knowledge graphs and graph convolutional neural networks, which further improves classification accuracy by integrating knowledge features and data features. Piao et al. [16] proposed a novel GNN-based sparse structure learning model for inductive document classification. The proposed model outperformed most existing models and revealed the necessity of learning sparse structures for long text. Jin et al. [17] introduced a new multi-element hypergraph gated attention network that achieves accurate document classification by capturing word positions and multi-element information.

In 2017, A. Vaswani [8] introduced the Transformer model to the field of natural language processing, achieving better performance than previous models. In recent years, a large number of researchers have conducted studies on the Transformer model. Wang et al. [18] in 2020 demonstrated that the random matrices generated by the self-attention mechanism can be approximated as low-rank matrices, leading to a new self-attention mechanism with linear time and space complexity. In the same year, Beltagy et al. [19] introduced Longformer, which designs a dilated sliding window to sparsify the attention matrix. Longformer employs a hierarchical architecture, performing local attention before global attention, and showed excellent performance in text classification. Zheng et al. [20] in 2020 proposed a hierarchical network structure using Bidirectional Gating Recurrent Units (BiGRU), replacing the Transformer structure to model local text segment features and extract more expressive global text features. The algorithm achieved outstanding results in sentiment classification tasks. Khandve et al. [21] in 2022 proposed a new hierarchical long-text classification algorithm, utilizing LSTM networks to fuse BERT output word vectors and extract text features, further reducing model parameters. Zhang et al. [22] propose a sentiment classification model based on the proposed Sliced Bidirectional Gated Recurrent Unit (Sliced Bi-GRU), Multi-head Self-Attention mechanism, and Bidirectional Encoder Representations from Transformers embedding. Dai et al. [23] investigated the roles of different components in Transformer-based long text classification models, applying the pre-training and fine-tuning paradigm to long text classification. They improved sparse attention and hierarchical approaches to optimize long text classification. Their proposed model was tested on public datasets, demonstrating significant performance improvements in long text classification.

With the advancement of deep learning technology, significant progress has been made in text classification methods based on deep neural networks. However, despite the success of these models in understanding text semantics, directly applying them to the automatic classification of long texts with specific content still poses challenges. On one hand, efficiently processing large-scale long text data, optimizing the use of computational resources, and improving processing speed remain technical difficulties. On the other hand, flexibly adjusting classification thresholds based on the characteristics of long texts to adapt to different text features and improve classification accuracy is also an urgent issue that existing technologies need to address. Therefore, there is a need for an automatic classification and threshold optimization method for specific content in long texts that is capable of accurate classification, high efficiency, dynamic adjustment, and strong threshold adaptability.

3 The Proposed Approach

To address the deficiencies of existing long text processing methods, such as poor classification accuracy, low efficiency, inadequate dynamic adjustment, and poor threshold adaptability, this paper proposes a Transformer-based sliding window threshold optimization long text classification model. The approach utilizes context-aware fusion of semantic information from long texts to improve the accuracy of automated long-text classification. This model builds upon the Transformer framework by adding a dedicated classification layer and fine-tuning using the cross-entropy loss function. To mitigate the impact of noise words in long texts on classification results, a semantic purification broadcasting algorithm is introduced. Additionally, a sliding window processing algorithm is employed to handle long texts in segmented regions. To enhance classification accuracy, various influencing

factors are considered, and the classification threshold is dynamically adjusted. An adaptive error feedback mechanism continuously optimizes the adjustment factors, allowing the classification threshold to better adapt to the actual data distribution and thereby reduce the misclassification rate. The framework of the proposed approach is shown in Figure 1.

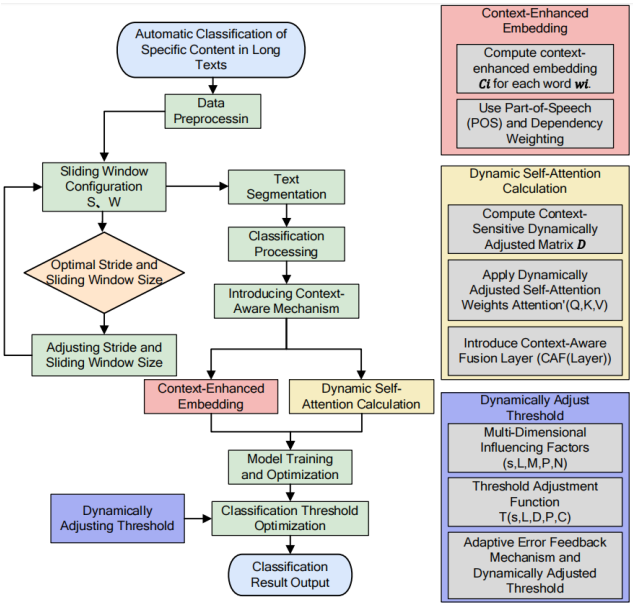

Figure 1. The framework of the proposed approach

3.1 Deep Learning-Based Long Text Semantic Analysis Model

To address the complex semantic relationships in long texts, we have developed a deep learning-based long text semantic analysis model for automatic classification of preprocessed long text information data, enabling the automatic classification of specific content within long texts. This model framework is based on the Transformer architecture, serving as the deep learning-based long text semantic analysis model.

Initially, existing web scraping tools are used to automatically collect long text information data, resulting in raw long text information. The raw data is then preprocessed to obtain preprocessed long text information data. The preprocessing of long text data includes text cleaning using regular expressions to obtain cleaned long text information data. Subsequently, existing automatic language detection technologies are employed to detect and filter the language of the cleaned long text information data, resulting in filtered long text information data. Using current technologies, the data undergoes tokenization, stop-word removal, stemming, lemmatization, and text vectorization to produce a data format suitable for subsequent processing, namely, the preprocessed long text information data.

The Transformer model is widely used in text analysis tasks due to its ability to capture long-range dependencies. It learns general language representations from large volumes of text data through pretraining, and then fine-tunes these representations for specific tasks to obtain a deep learning-based long text semantic analysis model. The fine-tuning process involves adding a classification layer to the Transformer model, where the number of nodes in this layer matches the number of categories. A loss function, such as cross-entropy loss, is used to generate prediction probabilities for each category. These prediction probabilities are then compared to thresholds set according to expert knowledge and predefined rules. Each text sample is automatically classified into the most probable category, thereby enabling automatic classification of long text information. The transformer model architecture is shown in Figure 2.

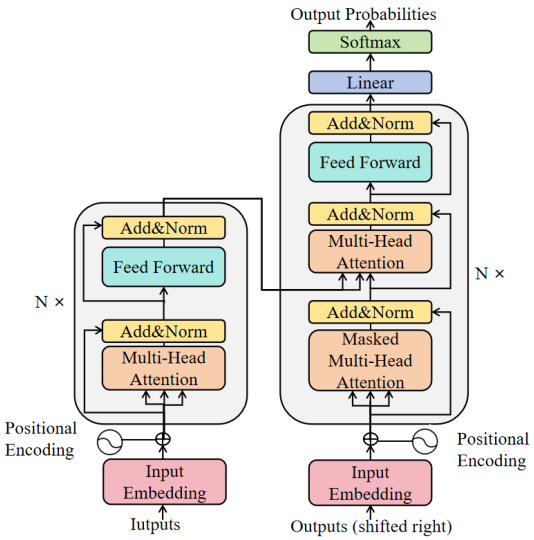

Figure 2. The transformer-model architecture

To address the issue of noise interference in long texts, the Semantic Purification Wave Algorithm is introduced. This algorithm dynamically adjusts the weight distribution of text data to reduce the impact of noisy data on the final classification results. For each text paragraph p_i in the long text data to be processed, initialize the weight $W_i = 1$. Each paragraph is then evaluated for noise, using regular expressions and Natural Language Processing (NLP) tools to identify potential noise patterns in the current paragraph. Weighted coefficients based on word frequency and word position are introduced, and Gaussian kernel density estimation is used to calculate the distribution density of noise words within the paragraph, which is part of the noise scoring n_i index calculation:

$$n_i = \log(1 + \int_{-\infty}^{\infty} \sum_{j=1}^{M} f_j \exp\left(-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right) dx) \quad (1)$$

Here, n_i represents the noise score index of text paragraph p_i ; M is the total number of detected noise words in text paragraph p_i ; f_j denotes the frequency of the j-th noise word, indicating the number of occurrences of the current noise word in text paragraph p_i ; x is a positional variable representing the relative position of the current noise word within text paragraph p_i ; μ_j is the position of noise word j in text paragraph p_i ; and σ_j^2 is the

variance used to control the influence range of noise word j, thereby affecting the rate at which the weight of noise word j changes with its position.

Dynamic weight adjustment of noise words is performed. To dynamically adjust the weights while considering the information entropy of the paragraph, the Beta function is used to combine the noise score index with the information entropy, thereby adjusting the weight of the current text paragraph:

$$w_i^{'} = \frac{B\left(n_i, H\left(p_i\right)\right)}{\sqrt{1+e^{-n_i}}} \tag{2}$$

Where w_i' is the adjusted weight of text paragraph p_i; $B(x, y)$ is the Beta function, which combines the noise score index n_i and the information entropy $H(p_i)$ to balance the impact of noise and information content; $H(p_i)$ is the information entropy of text paragraph p_i , used to measure the uncertainty and diversity of word distribution within p_i ; and n_i is the noise score index, derived from Gaussian kernel density estimation.

Reweight the text paragraph data by transforming the feature vectors using a polynomial kernel function and applying matrix multiplication for weighting:

$$v_i^{'} = (B \cdot \left(w_i^{'} \times V_i\right))^T \cdot K\left(V_i, V\right) \tag{3}$$

Where v'_i is the weighted feature vector of text paragraph p_i , used as input for the deep learning model; B is the transformation matrix learned from the data, used to extract and emphasize important features; w'_i is the weight of text paragraph p_i ; V_i is the original feature vector of text paragraph p_i ; $K(V_i , V)$ is the polynomial kernel function, used to enhance the relationship between V_i and the entire feature set V ; $K(V_i, V) = (V_i \cdot V + c)^d$, where c is the free parameter of the polynomial kernel, used to adjust the bias of the kernel function; and d is the degree of the polynomial, used to control the complexity of the kernel function.

This paper effectively captures subtle contextual relationships and long-distance dependencies in long texts using Transformer-based deep learning models, improving the accuracy of automatic classification for specific content. By adding a classification layer to the Transformer model and fine-tuning it using a cross-entropy loss function, the model can flexibly adapt to various types and structures of long text data. It performs well across different domains and languages. Additionally, the introduction of the Semantic Purification Wave Algorithm effectively addresses noise issues in long text data. This algorithm uses Gaussian kernel density estimation to dynamically assess and adjust the impact of noisy words within text paragraphs, significantly reducing the negative impact of noise on classification accuracy.

3.2 Real-Time Data Stream Processing Algorithm with Sliding Window

In the process of automatic classification of specific content in long texts, a real-time data stream dynamic processing algorithm based on sliding windows is used for dynamic regional processing of long text information. Additionally, by incorporating a deep context-aware attention mechanism, the model enhances its ability to understand contextual information in long texts. This combination aims to optimize both the efficiency and accuracy of the deep learning-based semantic analysis model for long text content classification.

In the automatic classification of specific content in long texts, a real-time data stream dynamic processing algorithm based on sliding windows is used for dynamic segmentation of long text information, optimizing the efficiency of automatic content classification. First, the size and step S of the sliding window W are defined, which affect the coverage and overlap of the information data as well as the computational load of subsequent processing. The size and step S of the sliding window W are adaptively adjusted based on the average paragraph length L_{avg} of the text.

$$W = \alpha \times L_{avg} \tag{4}$$

$$S = \beta \times W \tag{5}$$

Where α and β are adjustment parameters set empirically, used to control the ratio between window size, step size, and the average length of the text.

After dynamically adjusting the sliding window W and step size S, for a given long text T with length L, the text is segmented using the sliding window W with step size S to obtain a series of sub-texts Ti. For each segmentation point i, a weighting function $f(i)$ is introduced, which considers sentence boundaries near the end of the window and dynamically adjusts the starting point of the sliding window W to reduce cases where sentences are split between two windows. The formulas for the starting point $P_{start}(i)$ and ending point $P_{end}(i)$ of the i-th sub-text are as follows:

$$P_{start}\left(i\right) = i \times S + f\left(i\right) \tag{6}$$

$$P_{end}\left(i\right) = P_{start}\left(i\right) + W \tag{7}$$

$$f\left(i\right) = \arg\min_{b \in B} \left| b - \left(i \times S + W\right) \right| \tag{8}$$

where B is the set of all sentence boundary positions in the text; b denotes the positions of all sentence boundaries in the text.

3.3 Context-Aware Fusion and Classification Threshold Optimization Method Based on Long Texts

The deep context-aware attention mechanism is introduced to enhance the ability of deep learning-based long text semantic analysis models to understand contextual information in long texts, optimizing the accuracy of automatic classification of specific content

within the text. The deep context-aware attention mechanism improves the accuracy of long text automatic classification by enhancing the model's understanding of contextual information in different parts of the text. First, a weighting mechanism based on part-of-speech and dependency analysis is introduced to dynamically adjust the contextual weight of each word. Then, context-enhanced embeddings are computed, allowing the model to capture fine-grained contextual information. For each word w_i in the text, with its original embedding represented as v_i , the context-enhanced embedding C_i is computed as follows:

$$C_i = v_i + \sum_{j=-k}^{k} \dot{\alpha}_j \cdot v_{i+j} \tag{9}$$

$$\dot{\alpha}_j = \frac{\exp\left(score\left(j\right)\right)}{\sum_{m=-k}^{k} \exp\left(score\left(m\right)\right)} \tag{10}$$

$$score\left(j\right) = \frac{1}{distance\left(j\right) + \varepsilon} + POS_{weight(j)} + Dependency_{weight(j)} \tag{11}$$

where k is the size of the context window; $\dot{\alpha}_j$ is the attention weight of the word at position j in the context, reflecting the importance of that word to the current word; *score*(j) is the weighted score calculated based on position, part-of-speech, and dependency relations, used to dynamically adjust attention weights, enabling the model to more accurately capture linguistic features and structural information; *distance*(j) is the relative positional distance between the word w_i and the word at position j in the context; ϵ is a small positive number used to prevent division by zero, ensuring stability; and $POS_{weight(j)}$ and $Dependency_{weight(j)}$ are adjustments based on part-of-speech and dependency relations, respectively, to enhance the model's perception of grammatical structure.

Introduce a context-sensitive dynamic adjustment matrix D, so that the self-attention mechanism can more accurately reflect the contextual relationships between words when computing attention weights. Specifically, for each pair of words (i, j) in the context-sensitive dynamic adjustment matrix D, the calculation of d_{ij} considers the similarity of the contextual embedding representations between word i and word j:

$$d_{ij} = \tanh\left(C_i W^{context} C_j^T\right) \tag{12}$$

where d_{ij} is the additional attention adjustment amount for the i-th word with respect to the j-th word in a specific context, and is a context-sensitive dynamic adjustment factor; $W^{context}$ is a learnable weight matrix used to adjust the similarity calculation between contextual embedding representations; C_i and C_j represent the context-enhanced embedding representations of the i-th and j-th words, respectively.

Furthermore, based on the context-sensitive dynamic adjustment matrix D, calculate the dynamic self-attention weights as follows:

$$Attention'\left(Q,K,V\right) = soft\max\left(\frac{QK^T}{\sqrt{d_k}} + D\right)V \tag{13}$$

where $Attention'(\cdot)$ represents the deep context-aware attention mechanism; Q represents the Query matrix, which provides the representation of the current word (or subsequence) and is used to match with Keys (K). In self-attention mechanisms, the role of Q is to find the words (or subsequences) that are most relevant to the current word (or subsequence). K represents the Key matrix, which provides the representation of all words (or subsequences) and is used for matching with Queries Q. The purpose of K is to provide a searchable representation for each word (or subsequence) so that Q can find the most relevant content. V represents the Value matrix, which provides the representation of all words (or subsequences). When the most relevant K is found, the corresponding V will be used to compute the final attention output. The presence of V allows the model to extract the most useful information once relevance has been determined. D is the Dynamic Adjustment Matrix; d_k represents the dimension of the K vector, used to compute the scaling factor $\sqrt{d_k}$. The purpose of this scaling factor is to control the gradient vanishing problem caused by the growing inner product, ensuring the stability of training. The softmax normalization function is used to convert attention scores into a probability distribution. The softmax function outputs a vector of the same dimension, with values between 0 and 1, where the sum of all components equals 1.

After a layer of self-attention computation, in addition to applying the standard LayerNorm normalization and the feed-forward network (FFN), an additional context-aware fusion layer is introduced:

$$CAF(Layer) = Layer + \gamma \cdot Contextual\_Enhance(Layer) \tag{14}$$

In this, *CAF*(*Layer*) is the output of the context-aware fusion layer, which combines the original layer's output with context-enhanced representations to strengthen the model's contextual understanding. *Layer* is the output of the current layer, *Contextual_Enhance* is a function defined empirically to enhance context representation based on the current layer's output, and γ is a learnable fusion coefficient.

3.4 Adaptive Dynamic Classification Threshold Optimization Method

This paper addresses the challenges of processing long texts using a dynamic sliding window processing algorithm. By segmenting the text into regions, the algorithm reduces the computational resources required for processing large amounts of data all at once, thereby enhancing the efficiency of long text handling. The

introduced deep context-aware attention mechanism allows the model to more accurately understand complex contextual information and details within long texts. This is particularly effective in capturing subtle semantic differences, thereby improving classification accuracy. The Transformer-based model architecture, combined with the deep context-aware attention mechanism, enhances the model's ability to capture long-distance dependencies.

In the process of automatically classifying specific content in long texts, an adaptive error feedback mechanism is used to dynamically adjust the classification threshold based on multidimensional influencing factors, optimizing the threshold for automatic classification of specific content in long texts. The multidimensional influencing factors are defined as follows: confidence score s, text length L, topic diversity M, sentiment polarity P, and text complexity N. The confidence score represents the model's confidence in the presence of specific content in the text; text length refers to the number of words or characters in the text; topic diversity denotes the number or breadth of different topics in the text; sentiment polarity indicates the text's emotional orientation, such as positive or negative sentiment; and text complexity reflects the structural or content complexity of the text.

Next, the multidimensional influencing factors are quantified to obtain the quantitative representation of the aforementioned factors:

$$f_L(L) = \log(L+1) \tag{15}$$

$$f_M(M) = -\log(M+\varepsilon) \tag{16}$$

$$f_P(P) = \sigma(P) \tag{17}$$

$$f_N(N) = \sqrt{N} \tag{18}$$

Where, ε is to prevent the small positive number that the logarithm function is not defined when M is 0; $f_L(L)$ represents the quantitative representation of text length L, $f_M(M)$ represents the quantitative representation of topic diversity M, $f_P(P)$ represents the quantitative representation of emotion polarity P, and $f_N(N)$ represents the quantitative representation of text complexity N.

According to the quantized representations, define the threshold adjustment function $T(s, L, D, P, C)$:

$$\begin{aligned} T(s,L,D,P,C) = s + \hat{\alpha}\cdot f_L(L) + \hat{\beta}\cdot f_D(D) \\ + \hat{\gamma}\cdot f_P(P) + \hat{\delta}\cdot f_C(C) \end{aligned} \tag{19}$$

where $\hat{\alpha}$, $\hat{\beta}$, $\hat{\gamma}$, and $\hat{\delta}$ are adjustment factors. $\hat{\alpha}$ represents the adjustment factor for text length, $\hat{\beta}$ represents the adjustment factor for topic diversity, $\hat{\gamma}$ represents the adjustment factor for sentiment polarity, and $\hat{\delta}$ represents the adjustment factor for text complexity. These should be adjusted according to the specific application scenario. Further adaptive adjustments are made through an error feedback mechanism. Define the error feedback mechanism to dynamically update the aforementioned adjustment factors:

$$\Delta\hat{\alpha} = -\eta_{\hat{\alpha}} \frac{\partial \varepsilon}{\partial \hat{\alpha}} \tag{20}$$

$$\Delta\hat{\beta} = -\eta_{\hat{\beta}} \frac{\partial \varepsilon}{\partial \hat{\beta}} \tag{21}$$

$$\Delta\hat{\gamma} = -\eta_{\hat{\gamma}} \frac{\partial \varepsilon}{\partial \hat{\gamma}} \tag{22}$$

$$\Delta\hat{\delta} = -\eta_{\hat{\delta}} \frac{\partial \varepsilon}{\partial \hat{\delta}} \tag{23}$$

where $\Delta\hat{\alpha}$ represents the adjustment factor for text length after dynamic updating, $\Delta\hat{\beta}$ represents the adjustment factor for topic diversity after dynamic updating, $\Delta\hat{\gamma}$ represents the adjustment factor for sentiment polarity after dynamic updating, and $\Delta\hat{\delta}$ represents the adjustment factor for text complexity after dynamic updating. $\eta_{\hat{\alpha}}$, $\eta_{\hat{\beta}}$, $\eta_{\hat{\gamma}}$, and $\eta_{\hat{\delta}}$ are the learning rates for each adjustment factor, respectively. ε is the classification error rate.

The final dynamic adjustment threshold θ' is calculated as follows:

$$\theta' = \theta + T(s,L,D,P,C) \tag{24}$$

where θ is the initial threshold, and θ' is the threshold dynamically adjusted after considering multidimensional influencing factors.

Through the implementation process described, the paper achieves optimization of thresholds in the automatic classification of specific content within long texts. By considering multiple influencing factors, the dynamic adjustment of thresholds allows for more precise classification decisions based on the specific characteristics of different texts. This enhances the flexibility and adaptability of the classification process, especially when handling long texts. The adaptive error feedback mechanism enables the dynamic updating of adjustment factors, achieving continuous optimization of classification thresholds. This approach ensures that the classification process aligns more closely with the actual data distribution, thereby reducing misclassification rates. The dynamic adjustment of thresholds based on multi-dimensional influencing factors helps in more accurately identifying specific content, particularly in boundary cases, by minimizing false positives and false negatives through detailed adjustments.

4 Experiment Analysis

4.1 Dataset

In this paper, three benchmark datasets are used to

evaluate the model, namely the 20NG (20newsgroups) dataset [24], the MR dataset [25] and Amazon dataset. The 20NG dataset is an news dataset consisting of approximately 15076 news articles, categorized into 20 different classes. The MR dataset is a collection of movie reviews designed for sentiment binary classification. It consists of 10,662 documentst. The Amazon dataset has 34,657 pieces of product review data, divided into five categories. In the experiment, the dataset was divided into three groups with an 8:1:1 ratio, where 80% of the data was used for training, 10% for validation, and 10% for testing.

4.2 Experimental Environment and Parameter Configuration

The model in this article is implemented using Python 3.9 and the PyTorch deep learning framework. During the training process, the AdamW optimizer is used for training, with a batch size of 8 and a learning rate of 2 × 10-5. The early stop strategy has a patch value of 3. The specific environment configuration used on the experimental platform is shown in Table 1.

Table 1. Experimental platform configuration

Hardware information	Related configuration
Operating system	Ubuntu
CPU	Intel Xeon Gold 6240C
GPU	RTX4090
Cuda	12.0
Python	3.9
Pytorch	2.0.1

4.3 Evaluation Metric

This paper uses accuracy and F1 score to measure classification performance. Accuracy is a commonly used metric in deep learning research, while the F1 score is a standard measure for classification tasks. In single-label classification results, the outcomes can be categorized into True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN). Using these four types of results, evaluation metrics for each class can be computed. Assume the confusion matrix is shown in Figure 3.

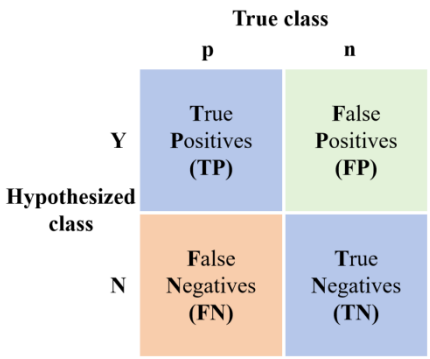

Figure 3. The confusion matrix

This section uses the following common metrics to evaluate the model's performance:

Average Accuracy (avg-Accuracy): In the performance evaluation of classification models, accuracy is the most common, fundamental, and intuitive metric. Accuracy refers to the ratio of the number of correctly classified samples to the total number of samples in a binary classification problem. For multi-label problems with a total of (K) classes, the avg-Accuracy is used to calculate the average accuracy across all classes. It is computed using the following formula:

$$avg-Accuracy=\frac{1}{n}\sum_{i=1}^{K}\frac{TP_i+TN_i}{TP_i+FP_i+TN_i+FN_i} \tag{25}$$

Average Recall (avg-Recall): In the performance evaluation of classification models, recall is an important metric that complements accuracy. Recall is used to assess the model's ability to identify TP samples. For multi-label problems, avg-Recall is used to compute the average recall across all classes. The avg-Recall for a total of (K) classes is calculated using the following formula:

$$avg-Recall=\frac{1}{n}\sum_{i=1}^{K}\frac{TP_i}{TP_i+FN_i} \tag{26}$$

avg-F1: Higher accuracy and recall generally indicate better classification performance, but accuracy and recall often present a trade-off in practice. Therefore, neither metric alone can fully reflect the model's classification effectiveness. The F1 Score addresses this by combining accuracy and recall into a single metric, providing a comprehensive measure of model performance. The F1 Score is the harmonic mean of precision and recall, with a higher value indicating better model performance. For multi-label problems, avg-F1 is used to compute the average F1 Score across all classes. The avg-F1 for a total of K classes is calculated using the following formula:

$$avg-F1=\frac{2\times micro-P\times micro-P}{micro-P+micro-P} \tag{27}$$

Here, micro-P represents the average precision, which evaluates the proportion of correctly predicted positive samples among all positive samples. It is calculated using the following formula:

$$micro-P=\frac{1}{n}\sum_{i=1}^{K}\frac{TP_i}{TP_i+FP_i} \tag{28}$$

4.4 Experimental Results

4.4.1 Accuracy of Different Algorithms

To evaluate the performance of the proposed model, we conducted comparative experiments with other models. The comparison included four main types of models: (1) word embedding-based models, such as PV-DBOW and FastText; (2) sequence deep learning models, including CNN and BiLSTM; (3) graph neural network-

based models, including TextSSL [20] and MHGAT [21]; and (4) Transformer-based models, including Sliced Bi-GRU [22] and TrLDC [23]. The benchmark models were comprehensively evaluated using the benchmark dataset to thoroughly assess the performance and effectiveness of the proposed model.

The results in Table 2 indicate that our proposed model outperforms other models on 20NG and Amazon datasets, and also achieves high accuracy on MR Datasets, which further proves the effectiveness and robustness of our proposed model in text classification. The "-" in the table represents no data. The bold in the table indicates the best results of the model. Specifically, word embedding models like FastText and PV-DBOW, which rely on word embeddings to capture semantic similarity, have shown some effectiveness as baseline methods. However, our analysis reveals that they have limitations in comprehensively capturing the complexity and contextual nuances of diverse datasets, as reflected in their performance. Sequence deep learning models, such as CNN and BiLSTM, demonstrated better performance by leveraging the sequential nature of text, particularly in capturing context and long-range dependencies within text sequences. These findings highlight the importance of sequence modeling in text classification tasks. The model based on Graph Neural Networks utilizes graph structures to capture relationships and contextual information between words, demonstrating good performance in certain scenarios. However, graph structures may not accurately reflect semantic relationships in text data, and the generalization ability of Graph Neural Network-based models might be limited by the graph structure. In contrast, Transformer models can more directly capture long-distance dependencies and contextual relationships through self-attention mechanisms, exhibiting better generalization capabilities. Our proposed model shows superior performance in capturing the complexity and subtle nuances of datasets, indicating its potential in text classification tasks.

Table 2. Test accuracy (%) comparison with baselines on benchmark datasets

Model	20NG	MR	Amazon
FastText	50.53	74.15	61.32
PV-DBOW	73.58	60.84	62.08
BiLSTM	71.18	76.21	62.34
CNN	52.34	76.82	63.68
TextSSL	89.50	76.12	-
MHGAT	90.18	77.11	-
Sliced Bi-GRU	87.23	80.48	62.85
TrLDC	85.68	**86.14**	-
Our	**90.61**	84.62	**65.83**

Table 3 shows that our proposed model is effective across multiple datasets. It also highlights the model's ability to accurately understand complex contextual information and details in long texts, effectively capture and utilize key information, and reduce misjudgments and missed judgments through careful adjustment of classification thresholds.

Table 3. Test avg-F1 score (%) comparison with baselines on benchmark datasets

Model	20NG	MR	Amazon
FastText	52.95	78.12	60.82
PV-DBOW	71.32	58.57	61.88
BiLSTM	75.18	76.18	62.19
CNN	79.85	76.47	62.27
TextSSL	82.87	75.12	-
MHGAT	84.21	78.18	-
Sliced Bi-GRU	86.83	79.88	62.45
TrLDC	84.90	**85.63**	-
Our	**90.43**	84.07	**64.41**

4.4.2 Optimization of Sliding Window and Step Size for Real-Time Data Stream Processing

To find the optimal sliding window and step size for the real-time data stream processing algorithm, we conducted experiments on the Amazon dataset. The results showed that when the sliding window is set to 256 and the step size is set to 128, the proposed model achieved the best performance. The optimal window and step size are illustrated in Table 4. This configuration effectively captures key information in the data stream while avoiding excessive computation and information loss, thereby enhancing the model's accuracy and efficiency in long text classification tasks.

Table 4. Test optimal sliding window and step size on the Amazon dataset

Sliding window_step size	Accuracy (%)	avg-F1 (%)
512_256	63.65	63.87
512_128	62.23	62.14
256_128	**65.83**	**64.41**
256_64	62.91	61.30
128_64	62.23	61.93

4.4.3 Convergence Experiment of the Model

To validate the convergence of the proposed model during training, one model from each of four types was selected and experiments were conducted using the 20NG dataset. The model made 10 epochs during training, and the loss values were recorded. It is compared with four baseline models of FastText, BiLSTM, TextSSL, and Sliced Bi-GRU.

As shown in Figure 4, although this model converges more slowly than the Sliced Bi-GRU model, its loss value is lower than the other four models as the number of epochs increases. During the first four epochs, the Sliced Bi-GRU baseline model converges the fastest. The model used in this study achieves its lowest loss value by the 6th

epoch and maintains the lowest loss in subsequent epochs. Therefore, after a certain number of epochs, this model will demonstrate its advantages.

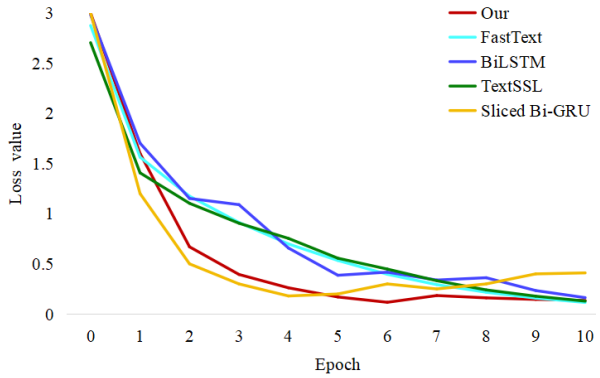

Figure 4. Loss curves of different models on the training set

5 Conclusion

In this paper, we propose a Transformer-based long text classification model with sliding window threshold optimization. This model incorporates a classification layer into the Transformer architecture and fine-tunes it using a cross-entropy loss function. We address the impact of noise words in paragraphs through the Semantic Purification Wave algorithm and process long texts regionally using a sliding window approach. Additionally, we dynamically adjust the classification threshold considering various influencing factors through an adaptive error feedback mechanism, which continuously updates the adjustment factors. This results in a sustained optimization of the classification threshold, making the classification process more aligned with the actual data distribution and reducing misclassification rates. The model has been validated on datasets and compared with existing text classification methods. In long text classification tasks, this model demonstrates significantly superior performance by accurately understanding complex contextual information and details in long texts, effectively capturing and utilizing key information, and reducing false positives and false negatives through precise threshold adjustments.

References

[1] M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, U. R. Acharya, ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis, *Future Generation Computer Systems*, Vol. 115, No. 1, pp. 279-294, February, 2021.

[2] M. G. Huddar, S. S. Sannakki, V. S. Rajpurohit, Correction to: Attention-based multimodal contextual fusion for sentiment and emotion classification using bidirectional LSTM, *Multimedia Tools and Applications*, Vol. 80, No. 9, pp. 13059-13076, April, 2021.

[3] I. Sutskever, J. Martens, G. E. Hinton, Generating text with recurrent neural networks, *Proceedings of the 28th international conference on machine learning (ICML-11)*, Bellevue, Washington, 2011, pp. 1017-1024.

[4] V. Diaz, W. E. Wong, Z. Chen, Enhancing Deception Detection with Exclusive Visual Features using Deep Learning, *International Journal of Performability Engineering*, Vol. 19, No. 8, pp. 547-558, August, 2023.

[5] V. Sudha, A. S. Vijendran, OSD-DNN: Oil Spill Detection using Deep Neural Networks, *International Journal of Performability Engineering*, Vol. 20, No. 2, pp. 57-67, February, 2024.

[6] S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural Computation*, Vol. 9, No. 8, pp. 1735-1780, November, 1997.

[7] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, *arXiv preprint arXiv*: 1409.0473, March, 2015. https://arxiv.org/abs/1409.0473v5

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, USA, 2017, pp. 30-41.

[9] J. Devlin, M. W. Chang, K. Lee, Kristina Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv*: 1810.04805, May, 2019. https://arxiv.org/abs/1810.04805v2

[10] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, T. Liu, On layer normalization in the transformer architecture, *International Conference on Machine Learning*, Vienna, Austria, 2020, pp. 10524-10533.

[11] J. Deng, L. Cheng, Z. Wang, Attention-based BiLSTM fused CNN with gating mechanism model for Chinese long text classification, *Computer Speech & Language*, Vol. 68, Article No. 101182, July, 2021.

[12] X. Li, H. Ning, Chinese text classification based on hybrid model of CNN and LSTM, *Proceedings of the 3rd international conference on data science and information technology*, Xiamen, China, 2020, pp. 129-134.

[13] K. Mao, J. Xu, X. Yao, J. Qiu, K. Chi, G. Dai, A text classification model via multi-level semantic features, *Symmetry*, Vol. 14, No. 9, Article No. 1938, September, 2022.

[14] W. Ai, Z. Wang, H. Shao, T. Meng, K. Li, A multi-semantic passing framework for semi-supervised long text classification, *Applied Intelligence*, Vol. 53, No. 17, pp. 20174-20190, September, 2023.

[15] Y. Wang, Y. Wang, H. Hu, S. Zhou, Q. Wang, Knowledge-Graph-and GCN-Based Domain Chinese Long Text Classification Method, *Applied Sciences*, Vol. 13, No. 13, Article No. 7915, July, 2023.

[16] Y. Piao, S. Lee, D. Lee, S. Kim, Sparse structure learning via graph neural networks for inductive document classification, *Proceedings of the AAAI conference on artificial intelligence*, Virtual Event, 2022, pp. 11165-11173.

[17] Y. Jin, W. Yin, H. Wang, F. He, Capturing word positions does help: A multi-element hypergraph gated attention network for document classification, *Expert Systems with Applications*, Vol. 251, Article No. 124002, October, 2024.

[18] S. Wang, B. Z. Li, M. Khabsa, H. Fang, H. Ma, Linformer: Self-attention with linear complexity, *arXiv preprint arXiv*: 2006.04768, June, 2020. https://arxiv.org/abs/2006.04768

[19] I. Beltagy, M. E. Peters, A. Cohan, Longformer: The long-document transformer, *arXiv preprint arXiv*: 2004.05150, December, 2020. https://arxiv.org/abs/20 04 .05150

[20] J. Zheng, J. Wang, Y. Ren, Z. Yang, Chinese Sentiment Analysis of Online Education and Internet Buzzwords

Based on BERT, *Journal of Physics: Conference Series*, Vol. 1631, No. 1, Article No. 012034, 2020.
[21] S. I. Khandve, V. K. Wagh, A. D. Wani, I. M. Joshi, R. B. Joshi, Hierarchical neural network approaches for long document classification, *14th International Conference on Machine Learning and Computing (ICMLC2022)*, Guangzhou, China, 2022, pp. 115-119.
[22] X. Zhang, Z. Wu, K. Liu, Z. Zhao, J. Wang, C. Wu, Text sentiment classification based on BERT embedding and sliced multi-head self-attention Bi-GRU, *Sensors*, Vol. 23, No. 3, Article No. 1481, February, 2023.
[23] X. Dai, I. Chalkidis, S. Darkner, D. Elliott, Revisiting Transformer-based Models for Long Document Classification, *Findings of the Association for Computational Linguistics: EMNLP 2022*, Abu Dhabi, United Arab Emirates, 2022, pp. 7212-7230.
[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in python, *The Journal of Machine Learning Research*, Vol. 12, No. 85, pp. 2825-2830, December, 2011.
[25] B. Pang, L. Lee, Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales, *arXiv preprint arXiv*: cs/0506075, June, 2005. https://arxiv.org/abs/cs/0506075

Biographies

Jin Pan received M.D. degree from Beijing University of Posts and Telecommunications (BUPT) in 2011. He is currently a senior engineer in National Computer Network Emergency Response Technical Team/ Coordination Center of China (CNCERT/CC). His research interests include network security and blockchain.

Yang Chen received the Ph.D. degree in computer science and technology from the Beijing University of Posts and Telecommunications (BUPT) in 2020. He is currently an engineer in the National Computer Network Emergency Response Technical Team/Coordination Center of China (CNCERT/CC). His research interests include cryptography, cloud computing and information security.

Chunlu Zhao received M.D. degree from Beijing University of Posts and Telecommunications (BUPT) in 2013. He is currently a senior engineer in National Computer Network Emergency Response Technical Team/ Coordination Center of China (CNCERT/CC). His research interests include cloud computing and artificial intelligence.

Yang Liu received M.D. degree from Beijing University of Posts and Telecommunications (BUPT) in 2009. She is currently a senior engineer in National Computer Network Emergency Response Technical Team/ Coordination Center of China (CNCERT/CC). Her research interests include network security and information security.

Jie Chu graduated from Shandong Technology and Business University. He is currently the assistant director of Institute of Network Technology (Yantai). His main research interests include artificial intelligence and information security.