
Web Interactive Content Filtering Model Integrating Prompt Classifier and Intent Understanding

Yanghao Li*, Shisong Wu, Pengfei Zhang,
Shengyao Zhuang and Lulu Chen

*China Southern Power Grid Artificial Intelligence Technology Co. Ltd., Guangzhou
510700, Guangdong, China*

*E-mail: yanghaoli01@163.com; wuss@csg.cn; zhangpengfei2022@csg.cn;
shengyao_Zhuang.CSG@outlook.com; fllch27@gmail.com*

** Corresponding author*

Received 11 May 2026; Accepted 02 July 2026

Abstract

To accurately identify and intercept harmful content generated in user interactions on Web platforms such as social media and online forums, this study proposes a Web interactive content filtering model that integrates Prompt classifiers and intent understanding. The model adopts a hierarchical review architecture with dual-module collaboration. The front-end deploys a Prompt classifier based on Deep Contrastive Learning (DCL), and uses the attention mechanism and deep convolutional network to perform rapid semantic analysis and preliminary screening of real-time comments and conversations. The back-end introduces an intent understanding module based on search-enhanced generation and Large Language Models (LLMs) to conduct in-depth contextual analysis of preliminary screening warning content to distinguish malicious attacks from legitimate sensitive topic discussions. Experiments showed that the system performed well on a test set containing 52,397 Chinese Web interactive texts. The Prompt classifier module achieved a precision of 95.2% and a recall of 96.5%. After collaborative calibration

Journal of Web Engineering, Vol. 25_7, 1483–1508.

doi: 10.13052/jwe1540-9589.2578

© 2026 River Publishers

with the intent understanding module, the false alarm rate of the system was significantly reduced to 4.4%, while the recall rate remained at 96.1%. From the results, this study provides a content security solution for Web interactive scenarios that combines high accuracy, strong semantic understanding capabilities, and real-time response performance. It can effectively purify the cyberspace environment while ensuring freedom of interaction, with important practical application value.

Keywords: Content filtering, prompt classifier, intent understanding, web interactive content, large language model.

1 Introduction

Driven by Internet technology, online platforms such as social media and online forums have become important platforms for the public to express their opinions and communicate with each other [1]. However, with the increase in online interactive content, it also gradually includes harmful content such as online violence, bad speech, and false information [2]. The above-mentioned harmful content will pollute the network environment and have a negative impact on citizens' rights and interests. Therefore, building a model that can filter content in cyberspace is an urgent problem.

Early Web content filtering technology relied on keyword matching and regular expressions, but this method struggled to identify the implicit meaning in the language and had poor accuracy [3]. The Prompt classifier has also been used to detect text, but it has inaccurate detection. Deep Contrastive Learning (DCL) is also used in content filtering models to detect bad content by comparing the gaps between different texts. However, due to the insufficient feature extraction capability of DCL, its filtering effect is still poor [4]. Large Language Models (LLMs) were once introduced into filtering models to improve the model's depth of semantic understanding, but its application in complex scenes still has limitations [5].

To solve the above problems, this research proposes a Web interactive content filtering model that integrates Prompt classifier and intent understanding. The proposed model uses a hierarchical review structure. In the front-end, a Convolutional Neural Network (CNN) and a Convolutional Block Attention Module (CBAM) are used to optimize the feature extraction stage in DCL, and a new Prompt classifier is designed to improve its feature extraction capability. At the back-end, the LLM is optimized by Retrieval-Augmented Generation (RAG) technology, and an intent understanding

module is designed to enhance the depth of LLM semantic understanding. The designed front-end and back-end are combined with the Web page interactive content filtering model to enhance the model's filtering accuracy for various harmful information in Web pages and maintain a stable network environment. The main contributions and innovations of this research are as follows: (1) The existing "DCL+CNN+CBM" mostly consists of module concatenation, with attention optimization and contrastive learning objectives separated. This study explicitly incorporates the attention weight of CBAM into the DCL loss constraint, enabling feature extraction to simultaneously optimize the manifold objective of "intra class aggregation and inter class separation", jointly optimizing attention mechanism and contrastive constraint. (2) The existing RAG review methods rely on general knowledge and are difficult to cope with domain value drift in sensitive content judgments. This study constructs a three-level knowledge base consisting of historical cases, sensitive topics, and dynamic rules as a content security "cognitive coordinate system". Through a reordering machine, the LLM general semantic space is projected onto the domain coordinate system to calibrate intent understanding, rather than simple knowledge retrieval concatenation. (3) The existing two-stage review framework separates the semantic space between the front-end and back-end, and the front-end false positive samples need to be re understood from scratch in the back-end. This study establishes a mapping relationship between the front-end comparative feature space and the back-end retrieval semantic space: the embedded features of the front-end high confidence warning samples are directly used as semantic query anchors for the back-end RAG retrieval, and the back-end intention understanding results are calibrated through dynamic threshold feedback to form a progressive closed-loop of "feature level initial screening semantic level fine tuning strategy level optimization".

2 Related Work

Many scholars have explored the interactive filtering model of Web pages. For example, Widayanti et al. and Jiang et al. used keyword matching methods to detect and filter the content in text messages. Results showed that the filtering accuracy was lower than 85% and the content filtering efficiency was poor [6, 7]. Xu et al. used DCL to establish a text filtering model to improve the model's filtering accuracy for bad information. The filtering accuracy could reach 90.2% [8]. Jing et al. also used comparative supervised learning in the text filtering model. The false positive rate was 9.3% [9]. However, due to

the insufficient feature extraction capability of DCL, its filtering effect still needed to be improved. Wu et al. and Zhao et al. used the LLM to inspect and filter the content in Web pages. The filtering efficiency was 89.3% and 90.1%, respectively [10, 11]. Liu et al. analyzed the text filtering effect of the Prompt classifier. The filtering accuracy reached 90.2% [12].

Both the CNN algorithm and the CBAM algorithm can extract features to improve the feature extraction capabilities of DCL. Many scholars have analyzed them. For example, Yu et al. used CNN to optimize DCL. The test results showed that the feature extraction ability of DCL was improved after optimization [13]. The Xia team used neural networks to optimize DCL and constructed a contrastive learning filtering method to filter out unwanted information in the recommendation system. The test results showed that its filtering effect was good [14]. Ma et al. also used CNN algorithm to improve DCL. The results showed that the feature extraction ability of DCL was improved [15]. Wang et al. and Botello et al. used RAG to optimize LLM to improve its semantic understanding accuracy. The test results showed that LLM's ability to understand text language was improved [16, 17].

To sum up, the current text filtering model still has poor filtering effects due to poor model feature extraction efficiency and weak text semantic understanding ability. To solve the above problems, this study uses CNN and CBAM to optimize DCL, and introduces them into Prompt to build a Prompt classifier to improve the feature extraction accuracy. Then, the RAG is used to optimize LLM to build a back-end intent understanding model to improve semantic understanding. The built front-end and back-end models, respectively, solve the poor feature extraction efficiency and weak text semantic understanding in the current filtering model, so as to identify and filter bad information in Web pages and create a stable network atmosphere.

3 Research Methods

3.1 Prompt Classifier Module Design

With the continuous development of the Internet, the probability of sensitive topics and inappropriate content appearing on online platforms such as social media and online forums is increasing. To accurately identify and intercept harmful content generated during user interactions on network platforms such as media and online forums, a Web interactive content filtering model is designed to filter harmful speech. In the traditional NLP security field,

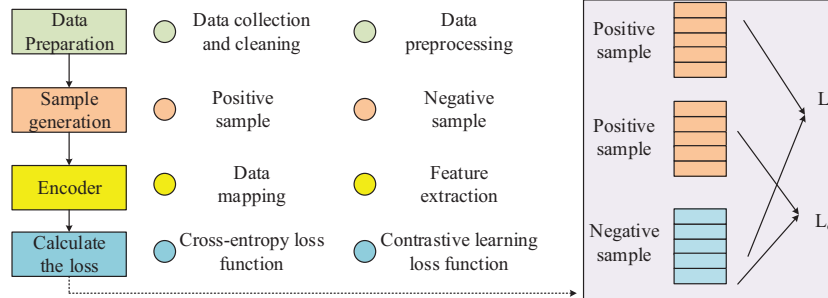


Figure 1 Basic process of DCL.

the Prompt classifier specifically refers to a defense module that identifies whether user input contains jailbreak attacks, instruction overwriting, or malicious prompt injection against LLM. To accurately describe user interaction text as a property to be detected as a system pending instruction (Prompt), user input is treated as a prompt to be detected, and DCL is used to determine whether it triggered harmful content patterns. This definition and Prompt Injection detection for LLM jailbreak attacks belong to different application scenarios, but they share the same core mechanism of “pre-semantic analysis and risk assessment of user input”. DCL can learn effective feature representations by comparing the differences and similarities between different samples [18]. In this study, DCL is introduced into the Prompt classifier, and DCL is used to conduct DCL on the content in the network, find the differences between instructions, and then interpret the content in the Web page. The basic process of DCL is shown in Figure 1.

From Figure 1, when DCL compares content on Web pages, it must first prepare data. This step includes data collection and cleaning. When collecting and cleaning data, first capture user comments, posts and conversation records from social media and online forums. Afterwards, the sensitive content is manually annotated, then the HTML tags, special characters and repeated content of the data are removed, and the encoding format of the data is unified. When preprocessing data, first standardize the text, unify upper and lower case, remove extra spaces, and standardize punctuation marks. After the data is prepared, the sample generation phase is entered, which is the core of DCL. This stage consists of positive sample construction and negative sample construction. To ensure alignment between DCL feature representation and malicious detection targets, a supervised contrastive learning strategy is adopted in the study. Positive samples are generated through text

augmentation strategies such as synonym replacement and back translation, and are constrained by label consistency to maintain the same semantic labels as the original samples. Negative samples consist of two categories: all samples with opposite category labels, and samples belonging to the same malicious category but different subclasses. Based on the above rules, the optimization objective of contrastive learning loss, namely task consistency formalization, can be expressed as minimizing the distance between similar samples (same labels) and maximizing the distance between dissimilar samples (different labels) in the feature space.

Formally, for samples in a batch, the contrastive loss is $Loss = -\log \frac{\exp(\text{sim}(z_i, z_i^+)/\tau)}{\exp(\text{sim}(z_i, z_i^+)/\tau) + \sum_{j=N(i)} \exp(\text{sim}(z_i, z_j)/\tau)}$, where z_i is the original sample feature, z_i^+ is the positive sample feature, z_j is the negative sample feature, and $N(i)$ is the negative sample index set. Through this alignment mechanism, the DCL-CNN-CBAM encoder forms a manifold structure of “same class aggregation, different class separation” in the feature space, enabling the front-end classifier to achieve high-precision initial screening of malicious content based on clear category boundaries. The encoder stage maps the input data into a potential representation space. A neural network is used to extract high-level features of the input data. The high-level features are compared with the annotated information to identify whether it is sensitive information. Finally, the loss is calculated, which has the cross-entropy loss function and the contrastive learning function, as presented in Equation (1).

$$\begin{cases} L = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \\ L_C = -\frac{1}{N} \sum_{i=1}^N \log \frac{\exp(\text{sim}(z_i, z_i^+)/\tau)}{\exp(\text{sim}(z_i, z_i^+)/\tau) + \sum_{j=1}^M \exp(\text{sim}(z_j, z_j^-)/\tau)} \end{cases} \quad (1)$$

where L signifies the cross-loss function value, N represents the number of batch samples, y_i signifies the true label of sample i , $\hat{y}_i$ represents the model prediction probability, L_C represents the contrastive learning loss function value, z_i represents the feature vector obtained by sample i after projection mapping, z_i^+ is a positive sample of sample i , z_i^- is a negative sample, $\text{sim}()$ is cosine similarity, τ is the temperature coefficient, M signifies the total negative samples. After calculating the loss value, backpropagation is performed to optimize the DCL and improve its ability to identify sensitive information. When identifying sensitive information, to enhance DCL's

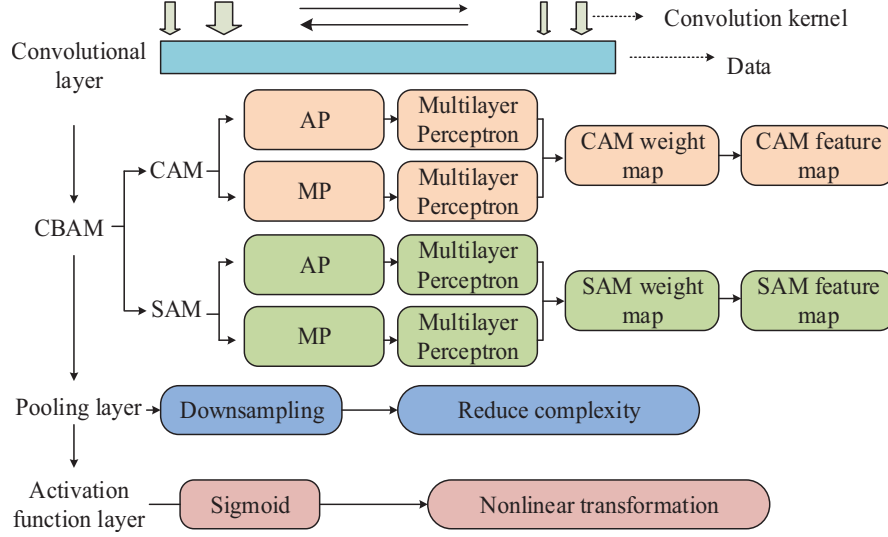


Figure 2 CNN-CBAM structure.

ability to extract advanced features from positive and negative samples, this study introduces CNN and CBAM mechanisms. CNN can extract features through the local receptive field and parameter mechanisms, and the CBAM mechanism can make the network focus on more important features through channel attention and spatial attention [19]. CBAM is added after the CNN convolutional layer to further enhance feature extraction capabilities. Figure 2 presents the CNN-CBAM.

From Figure 2, CNN-CBAM has a convolution layer, a CBAM layer, a pooling layer, and an activation function layer. In the convolutional layer, the CNN adopts different size convolution kernels to slide on the input data and extract local features by calculating the dot product of local areas. The mathematical expression in the convolutional layer is shown in Equation (2).

$$Y_{con} = W_{con} * X + b \quad (2)$$

where $*$ represents the convolution operation, b represents the bias term, X represents the input data, W represents the convolution kernel. The CBAM layer consists of a Channel Attention Module (CAM) and a Spatial Attention Module (SAM). The CAM is composed of an Average Pooling (AP) layer and a Maximum Pooling (MP) layer in parallel. It performs AP and MP operations on the input data respectively to obtain two one-dimensional

vectors. The expressions of AP and MP are presented in Equation (3).

$$\begin{cases} z_{avg}^C = \frac{1}{H \times W} \sum_{a=1}^H \sum_{b=1}^W X_c(a, b) \\ z_{max}^C = \max_{a,b} X_c(a, b) \end{cases} \quad (3)$$

where z_{avg}^C represents the global AP result of the C -th channel, H signifies the height of the feature map, W_{con} signifies the width of the feature map, $X_c(a, b)$ signifies the pixel value of the input feature map at the c -th channel position (a, b) , z_{max}^C represents the global MP result of the c -th channel. The two vectors are then passed through a multi-layer perceptron to perform nonlinear transformation on the vectors to mine deeper and more complex nonlinear relationships between features. The nonlinear transformation expression is shown in Equation (4).

$$\begin{cases} s_{avg} = W_1(\text{Re lu}(W_0(z_{avg}))) \\ s_{max} = W_1(\text{Re lu}(W_0(z_{max}))) \end{cases} \quad (4)$$

where s_{avg} represents the feature after nonlinear transformation of the AP result z_{avg} , W_0 represents the first level linear number, W_1 represents the second layer transformation, s_{max} represents the feature after nonlinear transformation of the MP result z_{max} . The vectors output by the multi-layer perceptron are added, and the activation function is used to process the output vectors to generate a channel attention weight map, which is expressed as shown in Equation (5).

$$\text{Mc}(X) = \sigma(s_{avg} + a_{max}) \quad (5)$$

where $\text{Mc}(X)$ represents the channel attention weight, σ represents the Sigmoid. The channel attention feature map is multiplied by the input feature map to obtain the CAM, as presented in Equation (6).

$$X' = \text{Mc}(X) \otimes X \quad (6)$$

where X' represents the CAM feature map. Then, the feature map is input into the SAM. The AP and MP are performed first to obtain two feature maps. The two feature maps are then spliced, and then the spliced feature map is passed through a convolution layer to reduce the number of channels to 1. Subsequently, the Sigmoid activation function is used to generate a spatial

attention weight map, and the weight map is multiplied by the input feature map to obtain the final feature map. The expression of this process is shown in Equation (7).

$$X'' = Ms(X') \otimes X' \quad (7)$$

where X'' signifies the final feature map, $Ms(X)$ signifies the spatial attention weight map. In the pooling layer, the obtained feature map is downsampled to reduce the size and parameter amount of the feature map and reduce the computational complexity. Finally, in the activation function layer, the Sigmoid is introduced again to enable the model to learn complex nonlinear relationships and alleviate the vanishing gradient. Through the above process, the feature information in the data is accurately extracted. Through DCL, it is identified whether the data contains sensitive information. The overall architecture of this study follows the data flow of “input text, embedding layer, CNN-CBAM feature extractor, projection head, and contrastive learning space”. Firstly, the features of the input text are extracted through CNN-CBAM. Then, the features are mapped to the space in DCL through the two-layer multilayer perceptron in the projection head, thus completing the feature fusion between CNN-CBAM and DCL.

3.2 Intention to Understand Module Design

The Prompt classifier is designed through DCL-CNN-CBAM and processed in real-time on the front-end of the Web page. However, this Prompt classifier is prone to misjudgment in complex semantic scenarios such as irony and metaphor. Therefore, a back-end intention understanding module needs to be introduced. The LLM can be trained with massive data, and RAG technology can improve the language generation model [20]. The research uses RAG technology to optimize LLM, introduce external knowledge sources into LLM, and make up for the shortcomings of outdated LLM training data or insufficient domain coverage. The basic process of LLM optimized by RAG is shown in Figure 3.

From Figure 3, the RAG-LLM consists of four core modules: search engine, knowledge base, reorderer and LLM generator. The search engine is the module in the RAG-LLM responsible for quickly finding relevant information. When the model receives input task instructions, the retrieval engine will search and match in the knowledge base based on the input content. The knowledge base is a warehouse for storing knowledge and a retrieval warehouse for search engines. This knowledge can come from various sources such as industry reports, professional literature, and news reports.

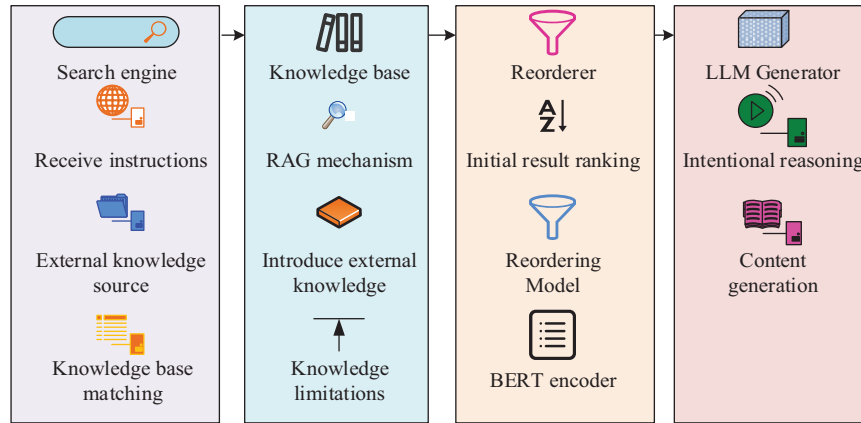


Figure 3 RAG-LLM flowchart.

The knowledge in the knowledge base is organized, classified and structured so that search engines can search and match more efficiently. By introducing a knowledge base, the RAG-LLM can break through the limitations of its own training data and acquire broader knowledge, thereby providing more accurate information when generating answers. The introduced knowledge base is the core support of the RAG mechanism and can provide real-time knowledge supplements for LLM. The reorderer can further filter and sort the preliminary results returned by the search engine. The re-ranker includes a re-ranking model based on the cross-encoder. This model splices the query text and the candidate documents, and performs deep interactive encoding through the BERT encoder. The attention mechanism is used to capture the fine-grained semantic matching relationship between the query and the candidate documents. Finally, the correlation score is output through the fully connected layer to evaluate the true semantic correlation between the candidate document and the current query term, effectively filtering out the noisy documents. The LLM generator performs final intent reasoning and content generation based on enhanced contextual knowledge. The knowledge base in the RAG-LLM is the external knowledge introduced by RAG. RAG is used to build the knowledge base, which contains a three-level hierarchical architecture. Its structure is shown in Figure 4.

From Figure 4, the three-tier architecture knowledge base built by RAG consists of a historical case knowledge base, a sensitive topic knowledge base and a dynamic rule base. The historical case knowledge base gathers rich case resources accumulated in various scenarios in the past, including

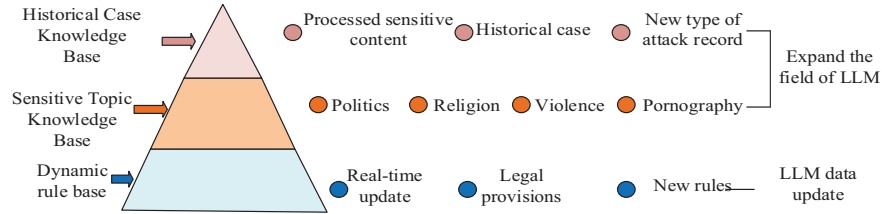


Figure 4 Construction of RAG knowledge base.

processed sensitive content samples, case analysis, and new attack mode records. This knowledge base achieves efficient retrieval through vectorized storage technology and assists the model in identifying hidden malicious intentions in complex contexts. The sensitive topic knowledge base has also changed the classification system and characteristic descriptions of multi-dimensional sensitive topics such as politics, religion, race, violence and pornography, and includes terminology libraries unique to each field, slang variations, and explanations of differences in cultural contexts. The above two databases can make up for the shortcomings of insufficient domain coverage of the LLM. The dynamic rule base is responsible for storing real-time updated platform community rules, laws and regulations, ensuring that the model has the ability to respond quickly to emerging sensitive topics and dealing with the defects of outdated LLM training data. The three-level knowledge base adopts a dynamic priority arbitration mechanism to achieve collaborative retrieval and conflict resolution. The search engine performs parallel queries on the three-level knowledge base, and the returned results are accompanied by source identification and confidence weights. When there is a conflict between multiple sources of results, the system arbitrates based on the priority of the dynamic rule base, historical case base, and sensitive topic knowledge base. The reordering machine integrates semantic scores and source weights into collaborative scores. If the score difference of conflicting documents is less than the threshold of 0.1, they will be marked as difficult samples and submitted for manual review. The third level repository maintains consistency through incremental synchronization: the dynamic rule repository synchronizes platform policies every 34 hours, the historical case repository is manually reviewed and incrementally written, the sensitive topic knowledge repository is maintained quarterly, and version control adopts a Git-like mechanism to ensure traceability of knowledge evolution. When the reorderer in RAG-LLM sorts the preliminary search results, its basic process is shown in Figure 5.

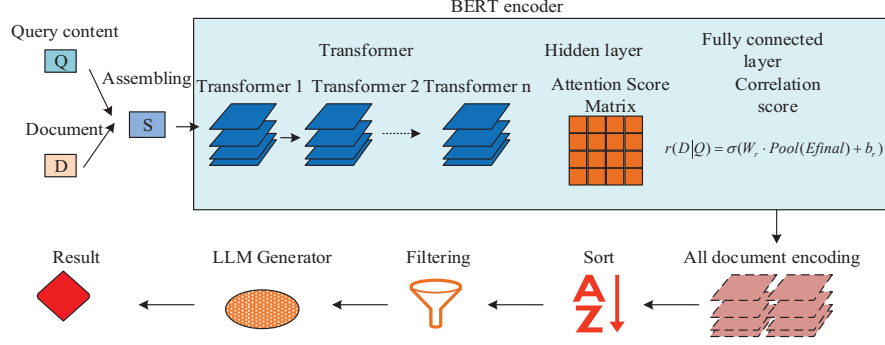


Figure 5 Sorting process.

From Figure 5, when the reorderer performs sorting, it first splices the query content and documents to form a single input sequence. Its splicing expression is shown in Equation (8).

$$S = \text{Concat}([CLS], Q, [SEP], D, [SEP]) \quad (8)$$

where S represents the spliced sequence, $[CLS]$ represents the classification mark, $[SEP]$ represents the delimiter, Q represents the input query content, D represents the candidate document. The spliced sequence is then input into the BERT encoder, which performs deep interactive encoding through the self-attention mechanism. The encoding expression is shown in Equation (9).

$$\text{Attention}(Q_h, K_h, V_h) = \text{softmax}\left(\frac{Q_h K_h^T}{\sqrt{d_k}}\right) V_h \quad (9)$$

where d_k represents the query dimension of each attention head, Q_h represents the query matrix of the h -th head, K_h signifies the key matrix of the h -th head, V_h represents the value matrix of the h -th head. Then, fine-grained semantic matching is performed. In each hidden layer of BERT, the model calculates the attention score matrix between the query and the candidate document, while the fully connected layer of BERT calculates the correlation score to represent the true semantic correlation between the document and the query term. The calculation is shown in Equation (10).

$$r(D | Q) = \sigma(W_r \cdot \text{Pool}(E_{\text{final}}) + b_r) \quad (10)$$

where $\text{Pool}()$ represents the pooling function, W_r represents the fully connected layer weight matrix, b_r represents the fully connected layer bias

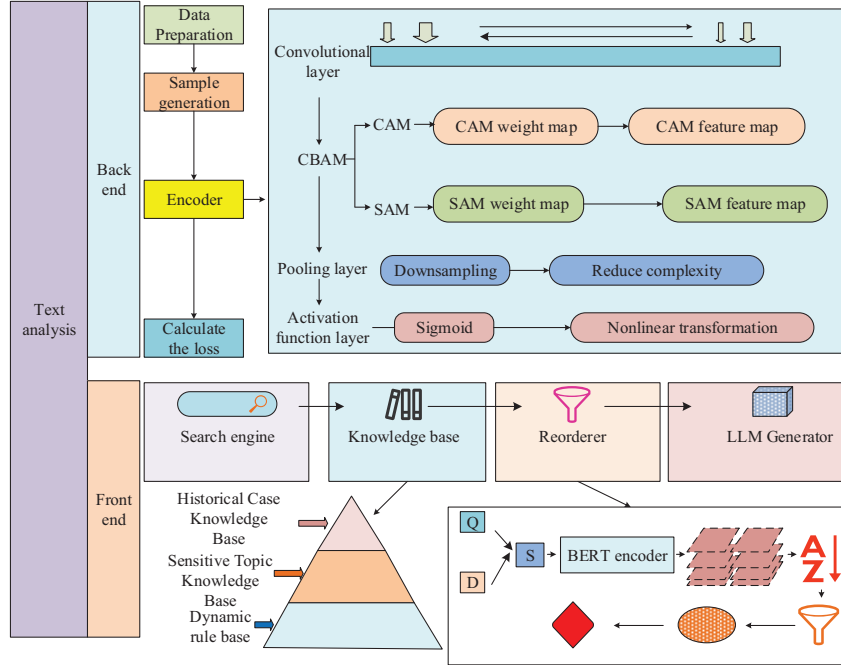


Figure 6 Basic structure of DCCRL.

term. The above steps for all candidate documents are repeated to obtain the combination of relevance scores, and then arranged in descending order of scores. Low-relevance documents are filtered out and only high-quality documents are fed into the LLM generator. Through the above process, a back-end intent understanding module is constructed to further improve the accuracy of filtering bad information on Web pages. The DCL-CNN-CBAM Prompt classifier is combined with the RAG-LLM intent understanding module to design a dual-module collaborative hierarchical review model, which is named DCCRL model, a dual-module collaborative hierarchical review framework. Figure 6 presents the basic structure of DCCRL.

From Figure 6, the dual-module collaborative hierarchical review framework adopts a collaborative working model of fast screening on the front-end and in-depth analysis on the back-end. On the Web front-end, the Prompt classifier performs real-time semantic analysis and preliminary screening of input content based on the DCL-CNN-CBAM architecture. When potentially sensitive content is detected, an early warning mechanism is triggered and the early warning content is transmitted to the back-end intent understanding

Table 1 Experimental environment configuration

Indicators	Specification	Indicators	Specification
GPU	NVIDIA GeForce RTX 4090 Ti 64GB DDR5	CPU	AMD Ryzen 7 7800X3D 8-Core Processor
Memory	5600MHz	OS	Ubuntu 22.04 LTS
Programming language	Python 3.10.12	Deep learning framework	PyTorch 2.1.0
CUDA version	CUDA 12.1	Large Language Model	ChatGLM3-6B

module. After receiving the warning content, the intent understanding module retrieves relevant background knowledge in the three-tier architecture knowledge base through the search engine in RAG-LLM. The re-ranker sorts the search results by semantic relevance and selects high-quality documents. Finally, the LLM generator performs in-depth intent reasoning based on the context and outputs the final review decision to intercept and filter malicious content.

4 Experimental Results

4.1 Experimental Setup

To verify the Web content filtering model, this study selected the Weibo Senti 100k sentiment analysis data set as the data set for this study. The data set can be downloaded directly from the GitHub Website. The data set contains 100,000 tagged Weibo texts, including 10,705 positive texts and 26,399 negative texts, and has been subjected to heavy data cleaning processing. During testing, 52,397 texts were selected as the test set. Table 1 presents the experimental environment configuration.

During the test, the parameters of the Web front-end classifier module and the intent understanding module were set as follows: CNN convolution kernel sizes were 3, 4, and 5, respectively, the convolution kernel is 256, the CBAM channel compression ratio was 16, the DCL temperature coefficient was 0.07, the batch size was 128, the learning rate was 1e-3, the training round was 20, and the loss function weight was 0.5. The search engine was FAISS-GPU, the maximum input length of LLM was 4096, and the LLM temperature parameter was 0.3. The dynamic rule update cycle is 34 hours. Sensitivity analysis was conducted on the front-end threshold and back-end threshold. The results are shown in Figure 7.

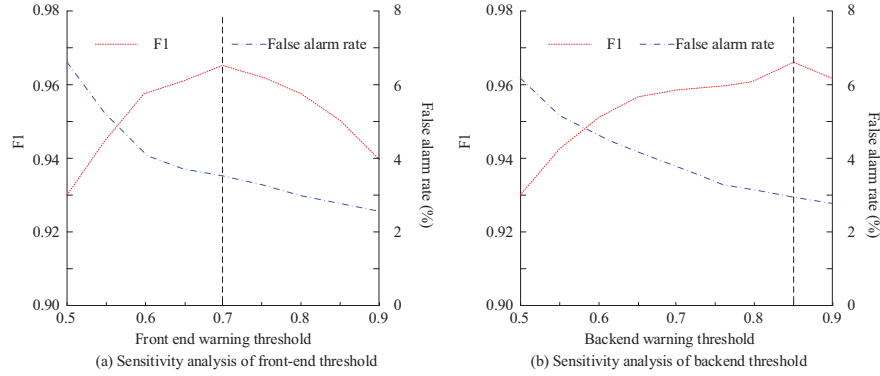


Figure 7 Sensitivity analysis.

As shown in Figure 7, when the current threshold was 0.70 and the backend threshold was 0.85, the F1 value of the system was the highest. Therefore, it is chosen as the initial threshold. Different threshold settings are recommended for different application scenarios based on the threshold in Figure 7. For example, on social media real-time comment platforms, which have large user base, fast content updates, and high real-time requirements, a balance needs to be struck between filtering efficiency, and user experience. Therefore, the front-end threshold was set to 0.70 and the back-end threshold was set to 0.85. The platform for protecting minors has fragile user groups, irreversible harmful content hazards, and extremely low security and fault tolerance space. It can implement a relaxed strategy by setting the front-end threshold to 0.55 and the backend threshold to 0.75.

4.2 Model Performance Analysis

To analyze the DCCRL, this study compared this filtering model with the currently used filtering model based on the Whale Optimization Algorithm-Long Short-Term Memory (WOA-LSTM), the Gaussian Mixture Models-Graph Convolution Neural Network (GMM-GCNN) filtering model, and the Variational Mode Decomposition-Whale Optimization Algorithm (VMD-WOA) filtering model. First, the filtering time and filtering accuracy of the four filtering models are analyzed. The results are shown in Figure 8.

From Figure 8(a), among the four filtering models, the average filtering time of the DCCRL model was 6.21 ms, while the filtering time of other filtering models was slightly lower than that of the DCCRL model, but the difference was less than 3 ms. From Figure 8(b), the filtering accuracy of

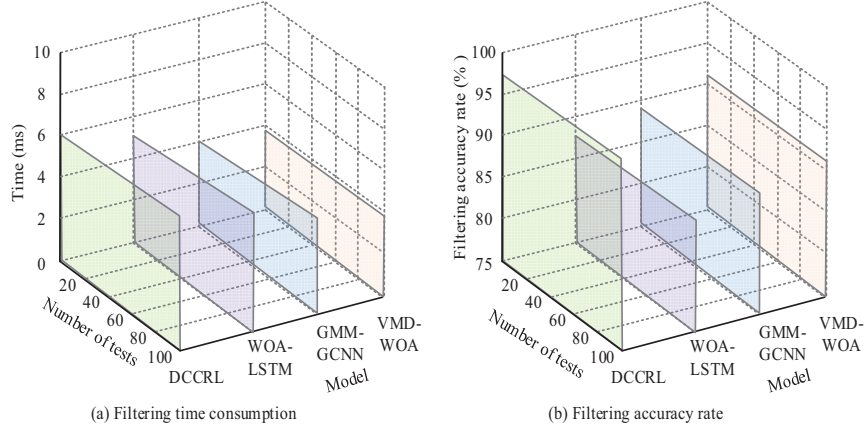


Figure 8 (a) Filtering time and (b) filtering accuracy.

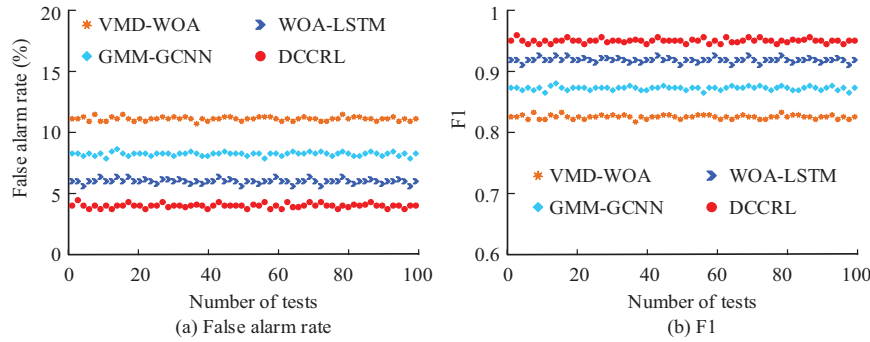


Figure 9 Comparison of (a) false alarm rate and (b) filtering efficiency.

DCCRl reached 95.2%, which was much higher than the 90.3% of the VMD-WOA model, the 88.2% of the GMM-GCNN model and the 86.1% of the WOA-LSTM model. The false alarm rate and filtering efficiency of the four filtering models are compared, as shown in Figure 9.

From Figure 9(a), among the four models, the DCCRl model had the lowest false alarm rate of only 4.4%. From Figure 9(b), the DCCRl model had the highest F1 value, at 0.97. Although the DCCRl model took slightly longer to filter than other models, the filtering accuracy and F1 value of the DCCRl were better than those of other models. The performance of the model was compared with benchmark models such as BERT, RoBERTa, TextCNN, and ERNIE. During performance testing, the HateTech dataset was used to map mild hatred, severe hatred, and inciting hatred in HateTech to

Table 2 Comparison of model performance

	Model	BERT-base	RoBERTawwm	TextCNN	ERNIE	DCCRL
Accuracy	Data	92.1%	92.4%	92.4%	93.1%	95.2%
	<i>p.vsDCCRL</i>	<0.05	<0.05	<0.05	<0.05	\
Recall	Data	93.2%	94.1%	93.4%	94.2%	96.5%
	<i>p.vsDCCRL</i>	<0.05	<0.05	<0.05	<0.05	\
False positive rate	Data	7.3%	5.5%	5.7%	6.1%	4.4%
	<i>p.vsDCCRL</i>	<0.05	<0.05	<0.05	<0.05	\
F1	Data	0.89	0.92	0.90	0.93	0.97
	<i>p.vsDCCRL</i>	<0.05	<0.05	<0.05	<0.05	\

malicious labels, ensuring that the model maintains consistent interception tendencies for harmful content of varying degrees. During preprocessing, regular expressions are used to filter user IDs, URLs, and duplicate emoticons, and Chinese text is subjected to full width and half width unification and simplified to ensure consistency with the input format of Weibo Senti 100k. A total of 100 repeated experiments were conducted, and significance tests were performed on BERT, RoBERTa, TextCNN, and ERNIE using the DCCRL model. The results are shown in Table 2.

According to Table 2, in the HateTech dataset, the DCCRL model still performed the best, significantly outperforming other models ($p < 0.05$).

4.3 Ablation Experiment

After verifying the performance of the dual-module collaboration review model and optimizing other current filtering models, the model was subjected to ablation testing to analyze the contribution in the dual-module collaboration module to the overall performance of the system. The front-end DCL-CNN-CBAM classifier module was subjected to ablation testing, as shown in Figure 10.

From Figure 10(a), in a separate Prompt prompter, the average prompt accuracy was 79.4%, which was low. After introducing DCL, the average accuracy of the front-end prompter could increase to 90.3%. After adding CNN to DCL, the prompt accuracy increased again to 93.4%. After introducing DCL and CNN-CBAM, the average prompt accuracy could reach 95.2%. From Figure 10(b), after introducing DCL and CNN-CBAM, the recall rate of the prompter can reach 96.5%. Then, the ablation test was performed on the back-end RAG-LLM intent understanding module and the overall DCCRL model. The results are shown in Figure 11.

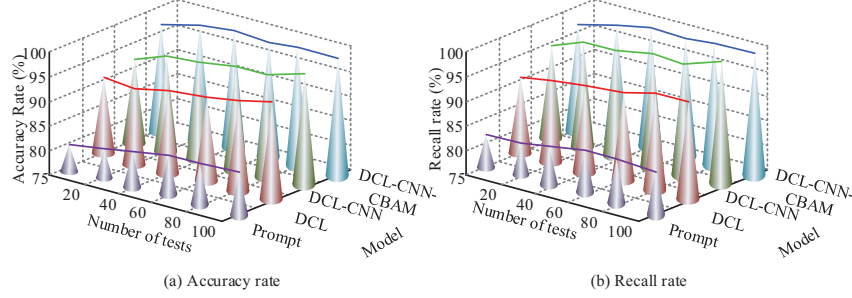


Figure 10 Front-end DCL-CNN-CBAM Prompt classifier ablation test.

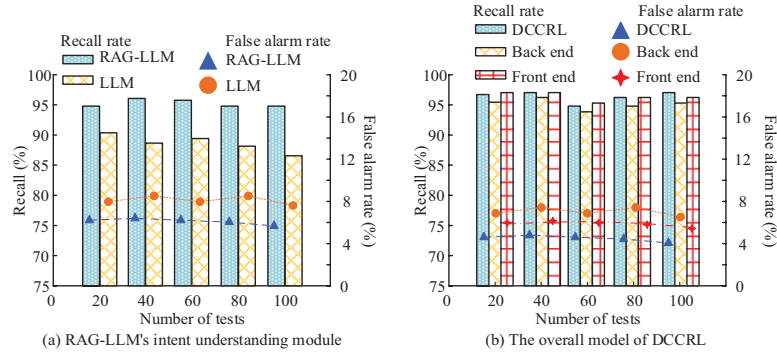


Figure 11 Abandonment test of (a) intention understanding module and (b) overall model.

From Figure 11(a), in the back-end intent understanding module, the average recall rate of using the LLM alone was 89.5%, while the average recall rate of using the RAG-LLM rose to 94.3%. The false alarm rates of the LLM and RAG-LLM were 9.3% and 5.1%, respectively. From Figure 11(b), the recall rates of the model front-end, back-end and DCCRL model were 96.5%, 95.7% and 96.1%, respectively, and the gap was relatively low. After the front-end and back-end were combined, the false alarm rate of the prompter was reduced to 4.4%. From this result, the RAG can enhance LLM's in-depth understanding of text. The DCL-CNN-DBAM model and RAG-LLM can improve the overall performance of the front-end and front-end DCCRL model.

4.4 Real-Time Analysis

To evaluate the dual-module collaborative review model in actual scenarios, experiments were conducted using the following testing environment

Table 3 Test results of typical cases

Case Number	Input	True label	Front-end Warning	Back-end Warning
1	“This statement is very stupid.”	Malicious attack	Warning	Warning
2	“This plan needs further discussion.”	Normal discussion	Without warning	Without warning
3	“There is something wrong with the brain.”	Malicious attack	Warning	Warning
4	“Really rubbish!”	Malicious attack	Warning	Warning
5	“Add your QQ number and you’ll get a surprise!”	Malicious manipulation	Warning	Warning

and inference process. The front-end is based on TensorRT 8.6.1 FP16 acceleration and CUDA Graph optimization, the back-end is deployed with ChatGLM3-6B (PagedAttention, KV Cache = 2048) based on vLLM, and FAIS-GPU uses IVF-FLAT indexing. The reasoning adopts a conditional triggering hierarchical architecture: all inputs are processed by the front-end module (preprocessing → CNN-CBAM feature extraction → comparative classification → threshold judgment). Only when the confidence level is greater than 0.70, the back-end module (vector retrieval → reordering → LLM inference → decision output) is triggered. The performance timing adopts CUDA Event synchronization to eliminate I/O delay, with each configuration repeated for 10 rounds and 1000 samples per round. Typical samples were selected for case analysis, as presented in Table 3.

From Table 3, when words related to “This statement is very stupid.” and “Really rubbish!” were entered into a Web page, the front-end of the filtering model could give an early warning, and the back-end also give an early warning. For the statement text of normal discussions, the filter did not issue relevant warnings. In the actual scenario, the relevant 1000 sentences in Table 2 were entered to test the filtering effect of the filtering model. The results are shown in Figure 12.

From Figure 12(a), after entering relevant sentences into the Web page, the filtering accuracy reached 96.3 and 96.2%, respectively. From Figures 12(b) and 12(c), the underreporting rate and false alarm rate were both less than 4%. From Figure 12(d), when the model was filtered, its filtering time was relatively low, both below 5 ms. From Figures 12(e) and 12(f), the F1 value was higher than 0.95, and the recall rate was also higher than 95%.

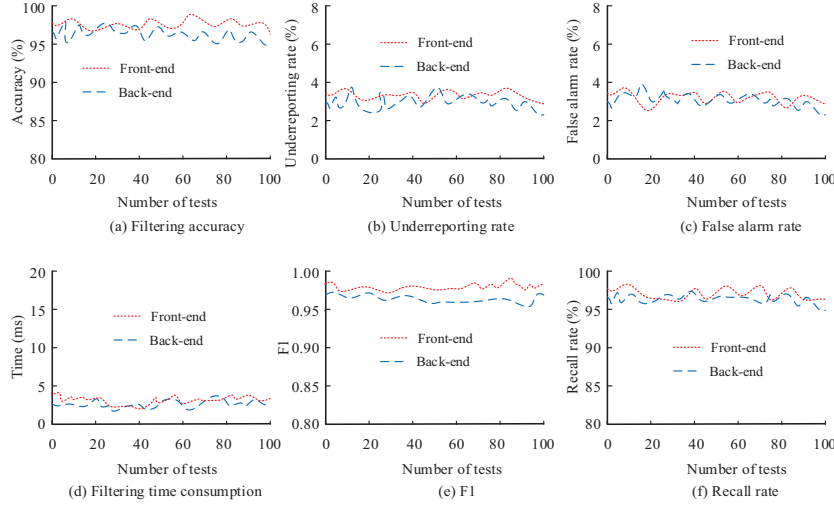


Figure 12 Analysis of actual filtration effect.

During the actual test process, its filtering effect remains favorable. According to the above test results, the DCCRL model can filter bad information and negative remarks in the interactive content of Web pages through its front-end and back-end collaboration, thereby maintaining the environment in the network.

5 Discussion

To enhance the current Web page interactive content filtering model's filtering accuracy for bad information and maintain a stable network environment, this research built a front-end Prompt classifier based on DCL-CNN-CBAM, and a back-end intent understanding module that combined RAG and LLM to jointly build a dual collaborative review model DCCRL to improve the filtering efficiency of the content interaction model through this model. The test results in the Weibo Senti 100k sentiment analysis data set showed that compared to VED-WOA, GMM-GCNN and WOA-LSTM filtering models, the filtering accuracy of DCCRL could reach up to 95.2%. This result is similar to the research conducted by Yu and Yin using CNN for intelligent filtering of inappropriate content, but the filtering accuracy of DCCRL in Web pages in this study is slightly higher than its research results. Analysis shows that this study adopts a front-end and back-end collaborative filtering model, which has significant advantages in Web page interaction [21].

Moreover, the actual test results of DCCRL show that the filtering accuracy of the front-end and back-end of the content filtering model reached 96.3% and 96.2%, respectively, and the underreporting rate and false alarm rate of bad information at both ends were less than 4%. From this result, DCCRL still has a robust filtering effect when filtering information in Web pages in actual situations. This result is similar to the personalized recommendation of Web page content constructed by Li using collaborative filtering, both of which can filter out unwanted information [22]. The DCCRL model not only has excellent filtering effects in public data sets, but also has robust performance in actual tests, and can be used to filter actual network interaction content to maintain a healthy network environment.

6 Conclusion

To improve the filtering accuracy of the Web page interactive content filtering model and maintain a stable network environment, this study built a DCCRL dual collaboration review model, hoping to optimize the filtering efficiency of the content interactive model through this model. The DCCRL had the best performance. Moreover, the performance of the DCCRL filtering model was also satisfactory, and the underreporting rate and false alarm rate of bad information at the front-end and back-end were relatively low. From the above results, the Web interactive content filtering model can accurately filter bad information in Web pages, so that administrators can maintain the network environment. However, because this filtering model introduces many algorithms, its training takes a long time. In the future, pruning technology and model compression can be used to shorten the training time of the model.

References

- [1] Budak C, Nyhan B, Rothschild D M, Thorson E. Misunderstanding the harms of online misinformation. *Nature*, 2024, 630(8015): 45–53. DOI: 10.1038/s41586-024-07417-w
- [2] Arcila Calderón C, Sánchez Holgado P, Gómez J, Barbosa M, Qi H, Matilla A. From online hate speech to offline hate crime: The role of inflammatory language in forecasting violence against migrant and LGBT communities. *Humanities and Social Sciences Communications*, 2024, 11(1): 1–14. DOI: 10.1057/s41599-024-03899-1
- [3] Rahman M R, Hezaveh R M, Williams L. What are the attackers doing now? Automating cyberthreat intelligence extraction from text on pace

- with the changing threat landscape: A survey. *ACM Computing Surveys*, 2023, 55(12): 1–36. DOI: 10.1145/3571726
- [4] Wang C, Ma W, Chen C, Zhang M, Liu Y, Ma S. Sequential recommendation with multiple contrast signals. *ACM Transactions on Information Systems*, 2023, 41(1): 1–27. DOI: 10.1145/3522673
 - [5] Lin J, Dai X, Xi Y, Liu W, Chen B, Zhang H, Zhang W. How can recommender systems benefit from Large Language Models: A survey. *ACM Transactions on Information Systems*, 2025, 43(2): 1–47. DOI: 10.1145/3678004
 - [6] Widayanti R, Chakim M H R, Lukita C, Rahardja U, Lutfiani N. Improving recommender systems using hybrid techniques of collaborative filtering and content-based filtering. *Journal of Applied Data Sciences*, 2023, 4(3): 289–302. DOI: 10.47738/jads.v4i3.115
 - [7] Jiang K, Jin G, Zhang Z, Cui R, Zhao Y. Incorporating external knowledge for text matching model. *Computer Speech & Language*, 2024, 87(7): 101638–101652. DOI: 10.1016/j.csl.2024.101638
 - [8] Xu L, Xie H, Li Z, Wang F L, Wang W, Li Q. Contrastive learning models for sentence representations. *ACM Transactions on Intelligent Systems and Technology*, 2023, 14(4): 1–34. DOI: 10.1145/3593590
 - [9] Jing M, Zhu Y, Zang T, Wang, K. Contrastive self-supervised learning in recommender systems: A survey. *ACM Transactions on Information Systems*, 2023, 42(2): 1–39. DOI: 10.1145/3627158
 - [10] Wu L, Zheng Z, Qiu Z, Wang H, Gu H, Shen T, Chen E. A survey on Large Language Models for recommendation. *World Wide Web*, 2024, 27(5): 60–78. DOI: 10.1007/s11280-024-01291-2
 - [11] Zhao Z, Fan W, Li J, Liu Y, Mei X, Wang Y, Li Q. Recommender systems in the era of Large Language Models (LLMs). *IEEE Transactions on Knowledge and Data Engineering*, 2024, 36(11): 6889–6907. DOI: 10.1109/TKDE.2024.3392335.
 - [12] Liu P, Yuan W, Fu J. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 2023, 55(9): 1–35. DOI: 10.1145/3560815
 - [13] Yu H, Wu H, Fang X, Li M, Zhang H. SR-CIBN: Semantic relationship-based consistency and inconsistency balancing network for multimodal fake news detection. *Neurocomputing*, 2025, 635(28): 1–12. DOI: 10.1016/j.neucom.2025.129997
 - [14] Xia B, Qin J, Han L, Gao A, Ma C. Knowledge filter contrastive learning for recommendation. *Knowledge and Information Systems*, 2024, 66(11): 6697–6716. DOI: 10.1007/s10115-024-02158-8

- [15] Ma J, Liu Y, Han M, Hu C, Ju Z. Propagation structure fusion for rumor detection based on node-level contrastive learning. *IEEE Transactions on Neural Networks and Learning Systems*. 2023, 35(12): 18649–18660. DOI: 10.1109/TNNLS.2023.3319661
- [16] Wang H, Li R, Jiang H, Tian J, Wang Z, Luo C, Tang X, Cheng MX, Zhao T, Gao J. Blendfilter: Advancing retrieval-augmented Large Language Models via query generation blending and knowledge filtering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024, 23(4): 1009–1025. DOI: 10.18653/v1/2024.emnlp-main.58
- [17] Botello J G, Frew L, Padilla J J, Weigle M C. Exploring Large Language Models for analyzing changes in Web archive content: a Retrieval-Augmented Generation approach. In *2024 IEEE International Conference on Big Data (BigData)*, 2024, 1(5): 2410–2418. DOI: 10.1109/BigData62323.2024.10826069
- [18] Sun P, Wu L, Zhang K, Chen X, Wang M. Neighborhood-enhanced supervised contrastive learning for collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering*, 2023, 36(5): 2069–2081. DOI: 10.1109/TKDE.2023.3317068
- [19] Hosseini S M, Ebrahimi A, Mosavi M R, Shahhoseini H S. A novel hybrid CNN-CBAM-GRU method for intrusion detection in modern networks. *Results in Engineering*. 2025, 28(1): 107103–107109. DOI: 10.1016/j.rineng.2025.107103
- [20] Lyu Y, Li Z, Niu S, Xiong F, Tang B, Wang W. Crud-rag: A comprehensive Chinese benchmark for Retrieval-Augmented Generation of Large Language Models. *ACM Transactions on Information Systems*, 2025, 43(2): 1–32. DOI: 10.1145/3701228
- [21] Yu Y, Yin X. A hypersensitive intelligent filter for detecting explicit content in learning environments. *Journal of Web Engineering*, 2024, 23(1): 89–110. DOI: 10.13052/jwe1540-9589.2314
- [22] Li X. Design of a Web content personalized recommendation system based on collaborative filtering improved by combining k-means and LightGBM. *Journal of Web Engineering*, 2025, 24(2): 267–290. DOI: 10.13052/jwe1540-9589.2425

Biographies



Yanghao Li (1996–), male, graduated from Guangdong University of Technology, China. He works as an assistant engineer at Southern Power Grid Artificial Intelligence Technology Co. Ltd. His research focus is large models in the power industry, power informatization, and power marketing.



Shisong Wu, born in April 1986, male, graduated from Tsinghua University, China. He works as a senior engineer at Southern Power Grid Artificial Intelligence Technology Co. Ltd. His current research focus is artificial intelligence, large models, and intelligent customer service.



Pengfei Zhang, born October 1994, male, graduated from Taiyuan University of Technology, China. He works as an engineer at Southern Power Grid Artificial Intelligence Technology Co. Ltd. His research focus is large models in the power industry, multimodal interaction, intelligent question answering, and intelligent question number processing.



Shengyao Zhuang (1990–), male, graduated from Yunnan University, China. He works as an engineer at Southern Power Grid Artificial Intelligence Technology Co. Ltd. His current research focus is artificial intelligence models.



Lulu Chen (2000–), female, graduated from South China University of Technology, China. She works as an engineer at Southern Power Grid Artificial Intelligence Technology Co. Ltd. Her current research focus is time series prediction.