applied sciences

MDPI

Article

Advancing Quality Assessment in Vertical Field: Scoring Calculation for Text Inputs to Large Language Models

Jun-Kai Yi and Yi-Fan Yao *

College of Automation, Beijing Information Science and Technology University, Beijing 100192, China; yijk@bistu.edu.cn
* Correspondence: afflatus1998@bistu.edu.cn

Abstract: With the advent of Transformer-based generative AI, there has been a surge in research focused on large-scale generative language models, especially in natural language processing applications. Moreover, these models have demonstrated immense potential across various vertical fields, ranging from education and history to mathematics, medicine, information processing, and cybersecurity. In research on AI applications in Chinese, it has been found that the quality of text generated by generative AI has become a central focus of attention. However, research on the quality of input text still remains an overlooked priority. Consequently, based on the vectorization comparison of vertical field lexicons and text structure analysis, proposes three input indicators D_1, D_2, and D_3 that affect the quality of generation. Based on this, we studied a text quality evaluation algorithm called VFS (Vertical Field Score) and designed an output evaluation metric named V-L (Vertical-Length). Our experiments indicate that higher-scoring input texts enable generative AI to produce more effective outputs. This enhancement aids users, particularly in leveraging generative AI for question-answering in specific vertical fields, thereby improving response effectiveness and accuracy.

Keywords: VFS evaluation algorithm; text quality; large language models; generative AI

check for updates

Citation: Yi, J.-K.; Yao, Y.-F. Advancing Quality Assessment in Vertical Field: Scoring Calculation for Text Inputs to Large Language Models. *Appl. Sci.* **2024**, *14*, 6955. https://doi.org/10.3390/app14166955

Academic Editor: Douglas O'Shaughnessy

Received: 10 May 2024
Revised: 3 August 2024
Accepted: 7 August 2024
Published: 8 August 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, Transformer-based models have taken a dominant position in the field of natural language processing (NLP) due to their outstanding performance [1,2]. Prior to this, Recurrent Neural Networks (e.g., LSTM) were widely used to solve a variety of existing NLP problems [3–5]. However, Recurrent Neural Networks exhibit limitations in capturing long-distance dependencies within data sequences, particularly struggling with information at the beginning or end of texts, as well as with distant information [6]. Furthermore, its architecture does not effectively support the parallelization of training and inference processes, which poses a significant challenge in terms of computational resource requirements [7]. In contrast, the advent of the Transformer architecture has effectively overcome these issues. It was initially proposed as a sequence-to-sequence encoder–decoder model [8], with its advantages lying in the use of attention mechanisms to capture long-distance relationships in text, and its ability to easily achieve the parallelization of computations.

With the development of more powerful GPUs and TPUs [9], it has become possible to create models with an increased number of parameters, allowing these models to achieve or even surpass human-level performance in an expanding array of tasks [10–12]. To make the model's responses more aligned with human needs, the development of the InstructGPT model employed a reinforcement learning from human feedback approach [13], which is particularly important for model fine-tuning.

However, due to the characteristics of the encoder-decoder model in the Transformer architecture, large language models (LLMs) need to convert input data into fixed-length representations, which the decoder then uses to generate output. Despite the "multi-head"

design of the Transformer architecture offering significant improvements in context and semantic understanding, its effectiveness remains limited when dealing with complex text problems and specific tasks. In particular, within specific vertical fields, to accomplish certain tasks, it necessitates preprocessing the input text to enhance the efficiency of the encoder and increase the accuracy of the generated text. For example, Xuemei Dong and colleagues, when utilizing ChatGPT-3.5-Turbo-0301 for zero-shot Text-to-SQL tasks, employed structured inputs and prompt engineering, effectively improving the success rate of SQL statement generation on the Spider test set [14]. Furthermore, to enhance the text generation effect, Le Xiao and colleagues combined an additional genetic algorithm with ChatGPT's natural language processing capabilities to generate news headlines, achieving notable results [15].

Based on these considerations, we designed an input text quality evaluation algorithm, VFS (Vertical Field Score), which is divided into three parts: prompt word score, vertical industry relevance score, and text logicality score. By performing a weighted calculation of the scores from these three aspects, we obtained the total score. Through experimental testing, we found that higher-scoring inputs were able to achieve higher quality, more specialized output results. This provides strong support for enhancing the output effects of generative AI in vertical industries.

2. VFS Algorithm Model

The VFS (Vertical Field Score) algorithm comprises three components: D_1, D_2, and D_3. The overall score, S, can be calculated using the following formula (Formula (1)):

$$S = \frac{\ln(e + D_1)(D_2 + D_3)}{2\ln(e + 1)} \tag{1}$$

where D_1 represents the prompt word score, D_2 represents the text structure score, and D_3 represents the content relevance score.

When designing evaluation metrics, it is crucial to understand the semantic processing logic of large language models (LLMs). LLMs first process the input information through their multi-layer Transformer architecture in a weighted manner, aiming to extract more comprehensive information. However, when processing human input, most model frameworks pay special attention to the logical structure of human language, such as conjunctions (e.g., "therefore", "then", "finally", etc.), during the training process. The models are highly sensitive to these terms and will segment sentences containing these words for analysis, in order to more accurately understand the meaning of the input text and its contextual logic. Therefore, large language models have a higher requirement for the logical coherence of lengthy texts, which includes a deep understanding of complex issues. The lack of structural words and overly verbose text narration can lead to biases in the model's information processing.

Additionally, large models are extremely sensitive to specific prompt words. For example, Xuemei Dong and others improve the accuracy of ChatGPT-3.5-Turbo-0301 in handling specific tasks, such as zero-shot Text-to-SQL tasks, by prompting ChatGPT to play the role of a professional SQL code engineer. The higher the level of specialization in the information, the more specialized the answers generated by the large model will be.

3. VFS Algorithm Analysis

3.1. Prompt Word Score D_1

Prompt words play a crucial role in the output of large language models (LLM). They enable the model to lock onto specific areas of knowledge within the vast training data, improving accuracy in subsequent tasks. In the application process of large language models, the role of prompt words is extremely broad. For example, in handling visual problems, specific prompt words can significantly improve the accuracy of recognition and analysis [16]. Similarly, in studying the programming capabilities of ChatGPT-3.5, appropriate prompt words are also used to enhance the success rate of code generation [17].

When calculating the prompt word score, the prompt words in the input text are evaluated. In the experiment, it was found that when the same prompt appears n times ($n > 1$), the generated results do not improve and may even deteriorate. However, when $n = 1$, the generated results show a significant improvement. Therefore, when setting the metrics, only the case where $n = 1$ is considered. If the text contains prompt words, the D_1 value is set to 1; if the text does not contain prompt words, the D_1 value is set to 0. Through this approach, it is possible to quantify the impact of prompt words on the output quality of large language models and reflect this in subsequent evaluation metrics.

3.2. Text Structure Score D_2

The degree of text structuring directly affects the output efficiency of large language models (LLMs). Input texts with strong structuring typically yield higher quality output results. In research on the generative effects of large language models across many fields, researchers often subconsciously opt for structured questions and input formats. For instance, in studies on extracting character relationships from texts, questions, and task requirements are clearly segmented through formatting or punctuation. This structuring of inputs helps ChatGPT-3.5 to provide more accurate answers [18]. However, in most practical application scenarios, users do not fully recognize the importance of input format, leading to varying degrees of errors in understanding by large language models, especially in the field of mathematics [19]. Therefore, it is clear that structuring is crucial for enhancing the output effectiveness of large language models.

Evaluating the degree of text structuring is a complex task. Typically, the degree of text structuring is directly positively correlated with its logical coherence, but the judgment of logical coherence often carries strong subjectivity. Therefore, when assessing the degree of text structuring, we choose to use text structure parameters to quantify the overall structuring score of the text. The text structure score, D_2, is composed of three parts: s_1, s_2, and s_3. The method for calculating its total score is shown in the following formula (Formula (2)).

$$D_2 = \frac{1}{3}(s_1 + s_2 + s_3) \tag{2}$$

The overall calculation process of D_2 is shown in Figure 1 below. The problem presented in the figure consists of five sentences and two logical connectors, with the logical connectors already highlighted in bold.

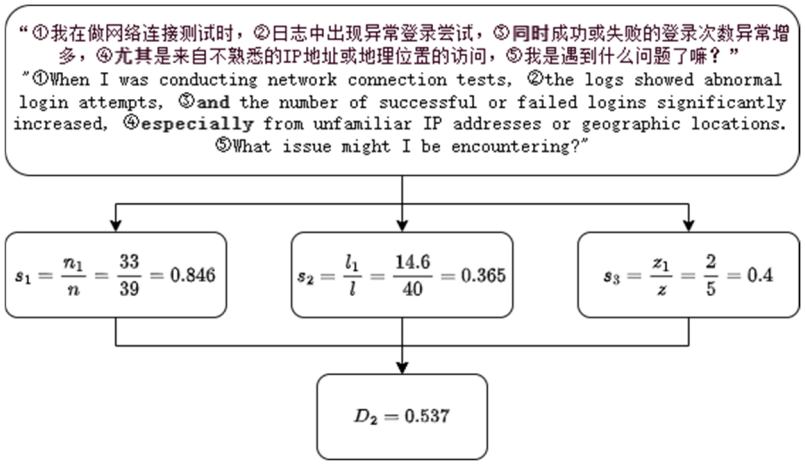


Figure 1. Text structure scoring flowchart.

Where s_1 represents the lexical richness of the text, calculated by segmenting the text into words to count the occurrence of each unique word n_1 and comparing the number of unique words to the total number of words n to derive s_1.

$$s_1 = \frac{n_1}{n} \tag{3}$$

In the evaluation metric system of this text, s_2 is a parameter used to measure the complexity of sentences. The specific method involves splitting the text into individual sentences using periods, then calculating the length of each sentence, and based on this, determining the average sentence length, l_1. According to related research [20], an analysis of a Chinese corpus consisting of 1.2 million characters found that it contained a total of 112,431 sentences, with an average sentence length of 10.91 characters. The study also showed that sentences of 30 characters or fewer accounted for over 95% of the total, while sentences of 40 characters or fewer accounted for over 99%. Based on these data, 40 characters are set as the maximum sentence length l, and the average sentence length l_1 of each text is compared to this maximum length l to calculate the score for s_2. Here, if the average sentence length of the text exceeds 40 characters, the score of l_1 will take its maximum value l, and s_2 will take its maximum value ($s_2 = 1$). Through this method, it is possible to quantitatively assess the sentence structure complexity of the text and incorporate it into the overall evaluation metric.

$$s_2 = \frac{l_1}{l} \tag{4}$$

The s_3 parameter is aimed at measuring the extent of logical connective words used in sentences. Logical connectives play a key role in the recognition and understanding of text by large language models. The primary function of these terms is to help the model more accurately understand the contextual relationships and core content of the text, thereby avoiding misunderstandings of synonymous sentences. Therefore, logical connectives are an important aspect of structuring sentence construction.

In the Chinese context, logical connectives encompass a variety of types including causal relationships, parallel relationships, transitional relationships, comparative-relationships, progressive relationships, and conclusions. By integrating open-source Chinese word libraries from the internet, these logical connectives have been extracted and compiled into a connective word dictionary.

Based on this, a formula for calculating the s_3 parameter was designed, as shown in Equation (5). Through this approach, it is possible to quantitatively assess the usage of logical connectives in text and incorporate it as an important component of the D_2 evaluation metric.

$$s_3 = \frac{z_1}{z} \tag{5}$$

In the evaluation model, z and z_1 are two key parameters, representing the theoretical maximum number of logical connectors in the input text and the actual number of logical connectors in the input text, respectively. First, the text is segmented into individual sentences using punctuation marks, thereby determining the total number of sentences. Subsequently, the maximum number of logical connectors in each sentence is determined. Here, the theoretical maximum number of logical connectors in the input text is equal to the total number of sentences. Next, during the analysis process, all logical connectors that appear in the text will be checked. These connectors are sourced from a predefined dictionary of connectors. Through this process, the number of logical connectors within each sentence, z_1, can be calculated. If z_1 exceeds z, then z_1 is set to z, and s_3 is assigned a value of 1.

This analysis method can quantitatively assess the frequency and distribution of logical connectives in the text. This assessment is important for understanding the degree of sentence structuring and the logical relationships within the context, while also providing crucial quantitative data for the D_2 evaluation score.

3.3. Content Relevance Score D_3

In the application of large language models in vertical fields, the importance of content relevance cannot be overlooked. For instance, in the application of large models in the medical field, when processing patient symptom information and providing suggestions,

the specificity of the patient's input text has a significant impact on the output results [21]. Similarly, in specialized models for the legal field, such as Chatlaw, given the complexity of real cases, slight differences in words can lead to drastically different outcomes. Therefore, in professional fields, the precise handling of content-relevant input becomes particularly crucial. Despite the Transformer architecture's excellent performance in natural language processing, the specificity and relevance of the input content still directly affect its output results [22].

Therefore, the stronger the relevance of content to vertical industries, the richer and more professional the output content of large language models becomes. Based on this premise, this study designed a vertical industry-specific vocabulary comparison algorithm based on the Word2vec model. By integrating genetic algorithms, the issue of irrelevant word vectors was effectively resolved, and ultimately, a score for evaluating the relevance of text content was obtained through vector comparison. The specific process for scoring content relevance is illustrated in Figure 2.

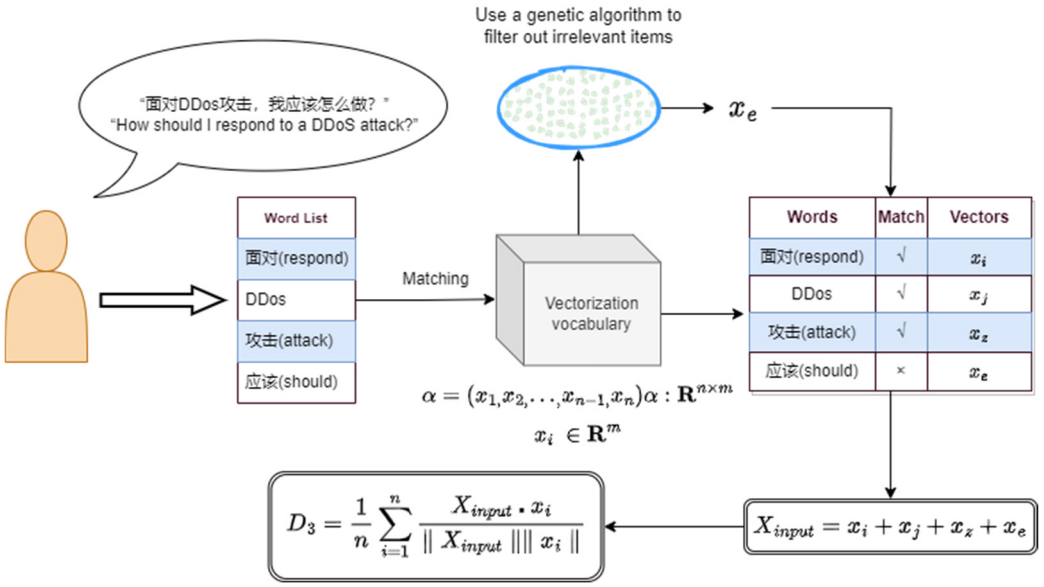


Figure 2. Content relevance scoring flowchart.

This algorithm not only enhances the model's understanding of professional terminology but also improves its accuracy and effectiveness in specific field applications.

As shown in Figure 2, after obtaining the input text, it undergoes preprocessing to generate tokenization results, which are then matched against a vectorized lexicon. In the specialized domain vectorized lexicon, if a word is present, its word vector is used. If a word is not present, it is assigned an unrelated vector, x_e. The vector x_e is obtained through a genetic algorithm screening of the vectorized lexicon. Finally, the vectors of all words are summed to obtain the input text vector X_{inout}. Then, it is compared with all words in the vectorized lexicon using cosine similarity to obtain the final content relevance score, D_3.

The calculation of D_3 primarily involves three main modules: the vectorized lexicon module, the unrelated word vector screening module, and the D_3 calculation module. These will be detailed in Sections 3.3.1–3.3.3, respectively.

3.3.1. Vectorized Lexicon Module

Word2vec is a word embedding model used for obtaining word vectors. It can vectorize words in text, with the word vectors containing semantic information between words and their context. This allows for a good measure of the relationships between words and provides a basis for understanding their meanings [23].

Word2vec includes two architectures: Continuous Bag of Words (CBOW) and Skip-gram. In the CBOW model, the central word is predicted based on the words surrounding

it, up to c words before and after. Based on this, each word serves as a central word to adjust the word vectors, as illustrated in Figure 3. Conversely, the Skip-gram model predicts the surrounding words, up to c words before and after, based on the central word, as shown in Figure 4.

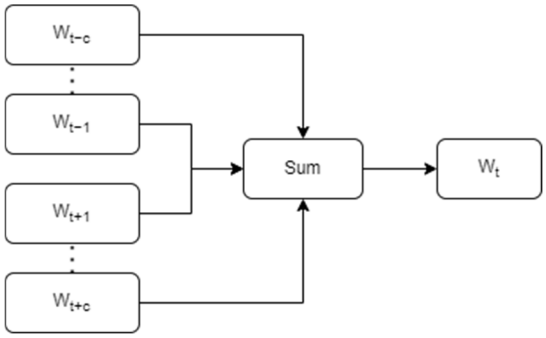


Figure 3. CBOW model.

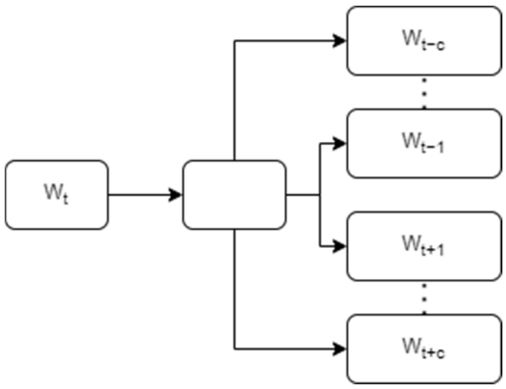


Figure 4. Skip-gram model.

In this module, due to the high number of specialized vocabulary terms contained in the vectorized lexicon designed for the vertical domain, the Skip-gram model is used as the training model to train the lexicon.

Based on a general Chinese cybersecurity textbook and 75 related papers in the field of cybersecurity, a cybersecurity knowledge base was compiled. A Word2vec word vector model with a vector dimension of 200 was trained, serving as the vectorized lexicon for the cybersecurity domain.

3.3.2. Unrelated Vector Screening Module

Genetic algorithms [24] are optimization methods derived from biological evolution theory, simulating natural selection and genetic mechanisms to search and optimize the solution space of problems. The core idea of this algorithm is to establish a population composed of a series of individuals, each representing a potential solution to the problem. Through multiple generations of iteration, experiencing genetic operations such as selection, crossover, and mutation, the population continuously improves, thereby gradually approaching the optimal solution to the problem.

In this study, genetic algorithms are applied to optimize the vector representation of text analysis. Specifically, the operation begins by randomly generating 200 vectors of 200 dimensions each, forming the initial population. These vectors represent potential solutions used to filter out the vectors most suitable for representing irrelevant word vectors (x_e). We define an adaptive function $F(v)$ to assess the fitness of each vector relative to the problem. Through multiple iterations, this function guides the evolution of the population, thereby filtering out the optimal vector representation for the subsequent evaluations of text content relevance.

The application of this method has improved the accuracy and efficiency of our algorithm, ensuring the relevance and professionalism of model outputs within specific fields.

$$F(v) = \left| \frac{1}{|V|} \sum_{v_i \in V} Similarity(v, v_i) \right| \tag{6}$$

$$Similarity(a, b) = \cos(\theta) = \frac{a \cdot b}{\|a\| \|b\|} \tag{7}$$

$$|V| = n \tag{8}$$

In this paper, each individual in the initially generated random population is first compared with all word vectors in the vectorized vocabulary V, and the fitness score of each individual in the population is calculated using the adaptive function $F(v)$. Subsequently, these scores are sorted, and the top 80 individuals with the highest adaptive scores are selected as the parents for the next generation of the population.

By combining the 100-dimensional features from both parents, new individuals in the population are generated. This process produces a total of 120 new individuals, who, together with the original parent individuals, form a new population. To increase the diversity of vectors and enhance the likelihood of high-quality target vectors emerging, a 5-dimensional genetic mutation is introduced into the offspring population.

In the implementation of the genetic algorithm, the convergence speed of the algorithm needs to be considered. The number of generations required varies for vectorized vocabularies of different scales and dimensions. In this study, based on practical requirements, 50 generations were chosen as the iteration number for the genetic algorithm, which has been proven to yield relatively good target vector individuals, meeting the needs of the research outcomes.

Through this method, the performance of the algorithm can be effectively improved, ensuring that the generated vectors accurately reflect the relevance of the text content, thereby enhancing the effectiveness of large language models in vertical field applications.

3.3.3. D_3 Calculation Module

Subsequently, the results of the segmentation are matched with the vocabulary in the professional lexicon, counting the number of successfully matched words (denoted as M_1) and extracting their corresponding vector representations (marked as Equation (9)). At the same time, the number of words not matched is recorded (denoted as M_2).

Next, as previously mentioned, the genetic algorithm is applied to generate the initial population for the lexicon α. The goal is to filter out a specific vector, referred to as the irrelevant vector x_e, whose absolute cosine similarity with every vector in the lexicon is as small as possible.

The vectors of unmatched words will be replaced by x_e. Subsequently, all obtained word vectors are normalized and summed up to obtain the final input text vector X_{input}. The final step involves calculating the cosine similarity between X_{Input} and each vector in the lexicon α, and then taking the arithmetic mean of these similarity values to obtain the content relevance score D_3. The specific calculation formula is shown in Equation (11).

Through this process, we can quantitatively assess the content relevance of the input text, thereby improving the accuracy and effectiveness of large language models in specific field applications.

$$x_i, i = 1, 2, 3 \cdots M_1 \tag{9}$$

$$X_{input} = \sum_{i}^{M_1} x_i + M_2 x_e \tag{10}$$

$$D_3 = \frac{1}{n} \sum_{i=1}^{n} \frac{X_{input} \cdot x_i}{\| X_{input} \| \| x_i \|} \tag{11}$$

4. Experimental Results

In this study, the most advanced Transformer large language model to date, ChatGPT-4, was selected as the primary experimental testing tool. The aim is to evaluate the correlation and effectiveness between the proposed evaluation criteria and the quality of generation.

To conduct this experiment, we generated 326 input question texts in the field of cybersecurity based on common cybersecurity issues. These texts were used to verify the effectiveness of the D_1, D_2, and D_3 metrics, as well as the final score S. The specific problem classifications are shown in Table 1. The problems are mainly divided into five major categories: information protection, network defense, application security, user education, and physical security. Corresponding input question texts were generated based on common issues in each domain.

Table 1. Types of cybersecurity-related questions.

Question Type	Quantity	Content
Information protection	72	Data tampering, theft, destruction, information leakage, unauthorized access, etc.
Network defense	103	DDoS attacks, firewall vulnerabilities, protocol vulnerabilities, IPS configuration issues, etc.
Application security	66	SQL injection, cross-site scripting (XSS), MFA anomalies, etc.
User education	32	Phishing emails, malicious links, scam messages, etc.
Physical security	53	Server failures, data center protection, computer hardware failures, etc.

4.1. Evaluation Index V-L

This experiment designed a generation evaluation metric, V-L (Vertical-Length), to more intuitively demonstrate the quality of generation by large language models (LLMs) in vertical fields. For the quality evaluation of generated results for questions in vertical fields, two elements were selected to constitute the details of the V-L metric: the content relevance score of the generated result, D_{output}, and the score for the length of the generated content, D_L. The specific formula for D_L is as follows (Equation (12)), where L_{Max} is the maximum content length among all answers and L is the content length of each answer. After obtaining D_{output} and D_L, they are combined to form the evaluation score for the V-L metric, $S_{V\text{-}L}$, with the specific formula as follows (Equation (13)).

$$D_L = \frac{L}{L_{Max}} \tag{12}$$

$$S_{V\text{-}L} = \frac{1}{2}(D_{output} + D_L) \tag{13}$$

4.2. Result Analysis

When designing algorithms, prompts are a crucial metric, especially in the application of large language models in vertical domains. Prompts directly affect the generated results. For the same input text, there may be multiple identical prompts as well as multiple different prompts. However, since this algorithm is used in a vertical domain where prompts are relatively fixed, the scenario of multiple different prompts is not considered. Therefore, experiments were conducted to address the situation where multiple identical prompts exist for the same problem.

The experiment randomly selected five different problems and, while keeping the problems unchanged, added n identical prompts to each problem ($n = 1$, $n = 2$, $n = 3$). Since the quality of answers generated by the large language model can fluctuate even for the same input text, each input text was tested three times. The values of L and D_{output} were averaged before calculating $S_{V\text{-}L}$ to eliminate the variability in generation quality. The experimental results are shown in Figure 5 below.

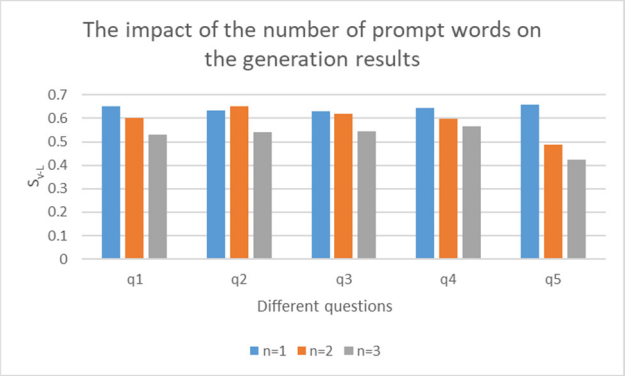


Figure 5. The effect of the number of prompt words.

The experiment demonstrates that when the number of prompts is 1, the overall generation quality is better than when the number of prompts is greater than 1. Moreover, as the number of prompts n increases, the generation quality deteriorates. Therefore, when designing the prompt scoring system, it is preferable to consider only the case where the number of prompts is 1.

In this study, for each problem, univariate adjustments were made to the variables D_1, D_2, and D_3 to pose questions to ChatGPT.

D_1 Effectiveness Validation Experiment: For the same problem, questions were posed to ChatGPT both with and without added prompts. Each state was tested three times. The values of L and D_{output} obtained were averaged, and then $S_{V\text{-}L}$ was calculated to validate the effectiveness of D_1. The scores from the 326 comparison groups were averaged separately to obtain the mean scores for $D_1 = 1$ and $D_1 = 0$. These results are recorded in Figure 6 below.

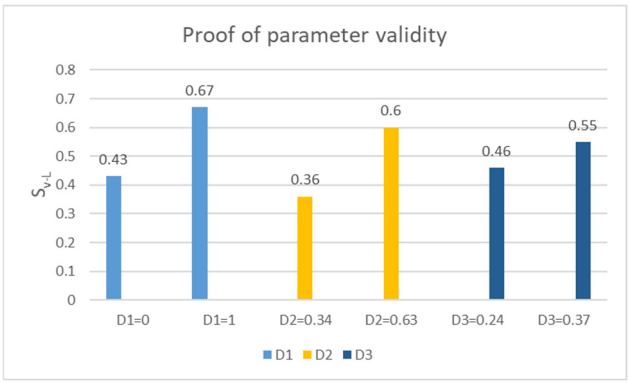


Figure 6. Validity proof of D_1, D_2, and D_3.

D_2 Effectiveness Validation Experiment: Under the condition of $D_1 = 0$, each problem was rewritten by adding logical connectors, increasing sentence length, and incorporating more cybersecurity domain vocabulary. Ensuring that the D_3 scores remain essentially the same, the D_2 scores were increased, and questions were posed to ChatGPT. Both the original and modified input texts were tested three times each, and the average values of L and D_{output} were calculated to determine $S_{V\text{-}L}$. The scores from the 326 comparison groups were averaged, and the average D_2 scores were also calculated. These results are recorded in Figure 6 below.

D_3 Effectiveness Validation Experiment: Under the condition of $D_1 = 0$, each problem was rewritten by changing colloquial descriptions to professional expressions and replacing replaceable words with cybersecurity-related vocabulary. Ensuring that the D_2 score error remains minimal, the D_3 scores were increased, and questions were posed to ChatGPT. Both the original and modified input texts were tested three times each, and the average values of L and D_{output} were calculated to determine $S_{V\text{-}L}$. The scores from the 326 comparison

groups were averaged, and the average D_3 scores were also calculated. These results are recorded in Figure 6 below.

Detailed data are shown in Figure 6. The results show that as the scores of D_1, D_2, and D_3 increase, the $S_{V\text{-}L}$ score exhibits a growing trend. This finding indicates that as the scores of these variables improve, the quality of the generated answers correspondingly enhances.

Based on the aforementioned experimental framework, tests were conducted to examine the impact of the evaluation score S from low to high on the effectiveness of generative AI answers by Transformer large models. The experimental results indicate that as the S score increases, the answers generated by the large model become progressively more detailed and professional. This finding confirms the effectiveness of the evaluation score S in guiding and predicting the output quality of large language models. Questions with higher S scores, due to optimizations in cue words, structuring, and professional content, enable the model to extract and process information more effectively, thereby generating more precise and in-depth answers.

Therefore, these experimental results not only validate the VFS (Vertical Field Score) input evaluation method but also provide valuable insights into how to utilize generative AI large models effectively.

Within this research framework, a series of continuous experiments were conducted on a total of 326 questions. During the experiment, the VFS of each input question was systematically improved, and the corresponding V-L (Vertical Length) scores were recorded. Each score is the average of three independent experiments.

To more intuitively demonstrate the correlation between the quality of input questions and the quality of generated content, this study randomly selected 10 samples from these questions and subsequently plotted a graph illustrating the relationship between input quality and generation quality.

As shown in Figure 7 for the same question, as the VFS increases, there is a significant improvement in the quality of the generated results. This indicates that, although there are certain limitations to improving the input quality for specific questions, the experimental data show a clear trend: a positive correlation exists between higher VFS scores and better quality of generated results. Notably, when the VFS score begins to rise from a lower level, the improvement in generated quality (i.e., V-L score) is relatively rapid. However, as the score further increases, the rate of improvement in generated quality gradually diminishes, indicating a decreasing marginal effect of score improvement on the quality of generation.

As shown in Figures 8–10, with the gradual increase in the Vertical Field Score (VFS), it is observed that the quality and length of ChatGPT's generated responses for the same question exhibit a corresponding upward trend.

In Figure 11, the generated response is displayed when the input question is modified to $S = 0.51$. Compared to Figures 8–10, the input problem in Figure 11 is more complete, featuring clearer descriptions of the details. Each sentence includes a logical connective to articulate the relationships between different sentences. Additionally, there is an enhanced use of specialized vocabulary.

It is evident that, compared to Figures 8–10, ChatGPT's response in Figure 11 shows a significant improvement in both length and professionalism.

When adjusting the input question, the first step is to eliminate colloquial expressions and provide a clearer description of the question. Next, the logical relationships between sentences are enhanced by adding logical connectors to each sentence, indicating the connections between different sentences. Finally, the actual problem is described in a more specialized manner by replacing general vocabulary with more professional terms whenever possible. It can be observed that increasing the number of prompts enhances both the length and professionalism of the output text, allowing the large language model's responses to be more focused on relevant vertical domains. Enhancing the logical coherence of the input text also makes the generated results more specific and significantly increases the length of the output text. The enhancement of specialized vocabulary makes the re-

sponses more professional and targeted, providing more detailed solutions to the problems and potentially offering additional solutions.

Additionally, in practical applications, enhancing the professionalism of the input question greatly tests the user's level of expertise. Therefore, for non-professional users in vertical domains, replacing general vocabulary with specialized terms can be challenging. In such cases, the use of logical connectors and prompts becomes crucial. However, both the use of prompts and the improvement of the input text's logical coherence significantly enhance the quality of the final generated results.

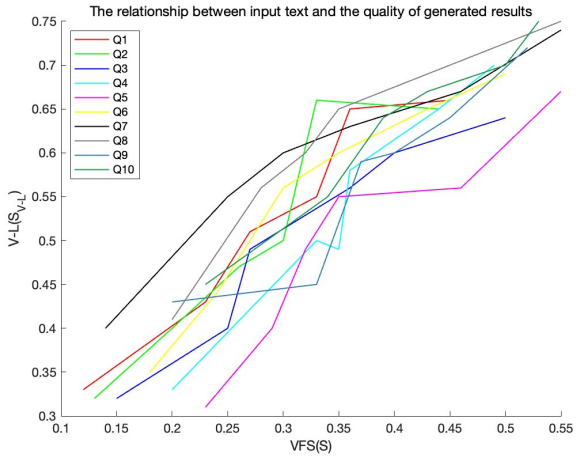


Figure 7. Input and output quality relationship.

Q&A(*S*=0.19)

Q:"我的服务器变得非常卡。登录页面也上不去，是受到什么攻击了么？"

Q:"My server has become very laggy, and I can't even access the login page. Is it under some kind of attack?"

A:"当你的服务器变得非常缓慢，并且连登陆页面都无法正常访问时，这可能是由于多种原因造成的。下面是一些可能的原因和对应的诊断方法:1.网络攻击:如分布式拒绝服务攻击(DDoS)或其他形式的网络攻击可能会导致服务器响应缓慢或不可用。你可以检查服务器的网络流量和日志,看是否有异常流量模式或大量来自特定IP地址的请求。2.资源过载:服务器可能因资源(如CPU、内存、磁盘空间或网络带宽)被大量占用而变慢。检查资源使用情况，确定是否有过度使用的情况。3.软件问题:服务器上运行的应用程序或服务可能存在配置问题或故障，导致性能下降。检查服务器上的应用程序和服务的状态和日志。4.硬件故障:服务器的硬件问题，如硬盘故障或内存问题，也可能导致性能下降。进行硬件诊断，以排除这种可能性。5.安全漏洞:如果服务器软件没有及时更新，可能会受到利用已知漏洞的攻击。检查所有系统和应用程序是否已更新到最新版本。对于非专业人士来说，处理这些问题可能比较复杂。如果你不确定如何进行这些检查或问题解决，可能需要寻求专业的I支持。同时，确保备份重要数据，以防万一需要恢复。"

A: "When your server becomes very slow, and even the login page cannot be accessed normally, this could be due to a variety of reasons. Here are some possible causes and corresponding diagnostic methods: 1. Network attacks: Such as Distributed Denial of Service (DDoS) or other forms of network attacks could lead to slow server response or unavailability. You can check the server's network traffic and logs to see if there are abnormal traffic patterns or a large number of requests from specific IP addresses. 2. Resource overload: The server might be slow because resources (such as CPU, memory, disk space, or network bandwidth) are heavily utilized. Check the resource usage to determine if there is excessive use. 3. Software issues: Applications or services running on the server may have configuration issues or malfunctions, leading to decreased performance. Check the status and logs of applications and services on the server. 4. Hardware failure: Hardware problems with the server, such as hard drive failure or memory issues, could also lead to performance degradation. Conduct hardware diagnostics to rule out this possibility. 5. Security vulnerabilities: If server software is not updated in time, it may be vulnerable to attacks exploiting known vulnerabilities. Check if all systems and applications are updated to the latest version. For non-professionals, dealing with these issues can be quite complex. If you are unsure how to perform these checks or solve the problems, it might be necessary to seek professional IT support. Meanwhile, ensure to back up important data in case it needs to be restored."

Figure 8. Question (S = 0.19).

Q&A(*S*=0.25)

Q:“你是一个优秀的网络安全工程师，我的服务器变得非常卡，登陆页面也上不去，我是受到什么攻击了么?”

Q:"As an excellent cybersecurity engineer, my server has become very laggy, and I can't even access the login page. Am I under some kind of attack?"

A:”作为网络安全工程师，我可以帮您分析一下可能的原因，服务器变或无法访问登录页面可能是由多种原因造成的，包括但不跟于网络攻击，以下是一些可能的原因和相应的诊断建议:1.DDoS攻击(分布式拒绝服务攻击):这是一种常见的网络攻击，攻击者利用多台计算机发起大量请求，导致服务器过载。可以检查服务器的流量日志，查看是否有异常的流量增加。2.服务器资源不足:可能是服务器的CPU、内存或碰盘空间不足。检查服务器的资源使用情况，看是否有资源过度使用的情况。3.网络问题:可能是网络连接问题，如带宽不足或网络硬件故障，可以检查网络连接和相关硬件设备。4.软件故障:服务器上运行的软件可能出现故障，如数据库崩清、Web服务异常等。检又查服务器上的软件日志，寻找可能的错误信息。5.恶意软件或病毒:可能服务器被恶意软件或病毒感染。进行病毒扫描，查看是否有恶意软件活动。6.配置问题:服务器的配置不当也可能导致性能问题，检查服务器的配置设置进行这些检查时，请确保你有足够的权眼，并小心操作，以免造成更多的问题，如果你不确定如何进行，最好寻求专业的网络安全专家的帮助，同时，建议定期各份重要数据，以昉数据丢失。”

A: "As a cybersecurity engineer, I can help you analyze the potential reasons why your server is lagging or the login page is inaccessible. This issue could be caused by various factors, including but not limited to network attacks. Here are some possible reasons and corresponding diagnostic suggestions:1.DDoS Attack (Distributed Denial of Service): This is a common type of network attack where attackers use multiple computers to send a large number of requests, causing the server to overload. You can check the server's traffic logs to see if there is an abnormal increase in traffic.2.Insufficient Server Resources: It might be that the server's CPU, memory, or disk space is insufficient. Check the server's resource usage to see if any resources are being overused.3.Network Issues: There could be problems with the network connection, such as insufficient bandwidth or hardware failures. Check the network connection and related hardware devices.4.Software Failures: Software running on the server might malfunction, such as database crashes or web service anomalies. Review the software logs on the server to find possible error messages.5.Malware or Virus: The server might be infected with malware or viruses. Conduct a virus scan to check for malicious software activity.6.Configuration Issues: Improper configuration of the server could also lead to performance issues. Review the server's configuration settings.When conducting these checks, ensure that you have sufficient permissions and proceed with caution to avoid causing further issues. If you are unsure how to proceed, it is best to seek help from a professional cybersecurity expert. Additionally, it is advised to regularly back up important data to prevent data loss."

Figure 9. Question (S = 0.25).

Q&A(*S*=0.35)

Q:“你是一个优秀的网络安全工程师，我的服务器变得卡顿且无法访问登录页面，这可能是受到攻击的迹象吗?”

Q:"As an excellent cybersecurity engineer, my server has become sluggish and I can't access the login page. Could this be a sign that it's under attack?"

A:”作为一个网络安全工程师，我可以帮您分析这种情况，服务器变得卡顿并且无法访问登录页面，确实有可能是受到了网络攻击，但也可能是由于其他技术原因，以下是一些可能的情况和相应的检查方法。1.分布式拒绝服务攻击(DDoS):这是一种常见的网络攻击方式。攻击者利用大量的系统发起海量的网络请求，导致服务器无法处理正常的请求。您可以检查网络流量日志，看是否有异常的流量增加。2.资源耗尽:服务器的CPU、内存或磁盘空间可能已经达到或接近其容量限制，检查服务器资源使用情况，看是否有资源过度使用的迹象。3.网络问题:可能是由于网络连接问题，例如带宽不足或网络硬件故障，检查服务器的网络连接和相关硬件设备。4.软件问题:服务器上的软件可能出现故障，例如数据库服务崩清或Web服务器软件故障，检查相关软件的日志，寻找可能的错误信息。5.恶意软件或病毒感染:服务器可能被恶意软件或病毒感染。进行病毒扫描，检查是否有恶意软件或病毒活动。6.错误:服务器配置错误也可能导致性能问题或服务中断。检查服务器的配置设置，确保一切设置正确。建议的操作步骤:1.立即备份:首先确保备份所有重要数据，以防数据丢失或损坏。2.检查日志文件:服务器日志通常是发现问题的第一步。检查Web服务器日志、数据库日志以及其他服务日志。3.资源监控:使用系统监控工具来查看CPU、内存、碰盘和网络的使用情况。4.网络流量分析:检查是否有异常的网络流量，这可能是DDoS攻击的迹象。5.安全扫描:使用安全扫描工具检查系统是否变到恶意软件或病毒的感染。如果你不确定如何进行这些检查，或者情况复杂，建议寻求专业的网络安全专家进行详细诊断和处理，同时确保定期更新和打补丁，以减少安全源洞的风险。”

A: " As a cybersecurity engineer, I can help you analyze this situation. Your server becoming sluggish and the inability to access the login page could indeed be indicators of a network attack, but it could also be due to other technical reasons. Here are some potential situations and corresponding methods of investigation:1.Distributed Denial of Service Attack (DDoS): This is a common type of network attack. Attackers use a massive number of systems to launch a flood of network requests, causing the server to be unable to process legitimate requests. You can check the network traffic logs to see if there's an abnormal increase in traffic.2.Resource Exhaustion: The server's CPU, memory, or disk space might have reached or be nearing its capacity limit. Check the server resource usage to see if there are signs of overuse.3.Network Issues: It could be due to network connection problems, such as insufficient bandwidth or network hardware failure. Inspect the server's network connections and related hardware devices.4.Software Problems: There could be a malfunction in the software on the server, such as database service crashes or web server software failures. Check the logs for these pieces of software for possible error messages.5.Malware or Virus Infection: The server might be infected with malware or a virus. Run a virus scan to check for malicious software or virus activity.6.Configuration Errors: Errors in the server configuration could also lead to performance issues or service interruptions. Review the server's configuration settings to ensure everything is set up correctly.Recommended Steps:1.Immediate Backup: First, ensure all important data is backed up to prevent loss or damage.2.Check Log Files: Server logs are often the first place to look for issues. Check web server logs, database logs, and logs for other services.3.Resource Monitoring: Use system monitoring tools to observe the usage of CPU, memory, disk, and network.4.Analyze Network Traffic: Look for abnormal network traffic, which could indicate a DDoS attack.5.Security Scanning: Use security scanning tools to check if the system has been infected with malware or viruses.If you are unsure about how to conduct these checks, or if the situation is complex, it's advisable to seek assistance from a professional cybersecurity expert for a detailed diagnosis and remedy. Also, ensure regular updates and patching to minimize the risk of security vulnerabilities."

Figure 10. Question (S = 0.35).

Q&A(*S*=0.51)

Q:"你是一个优秀的网络安全工程师，我的服务器目前表现出显著的卡顿现象，然后登录页面和目标地址都无法正常访问。所以这种现象是否可能是由于遭受了网络攻击所导致的？"

Q:" As an excellent cybersecurity engineer, My server is currently experiencing significant lag, and both the login page and the target address are inaccessible. Could this issue be due to a network attack?"

A:"是的，您的服务器出现显著的卡顿现象并且登录页面无法正常访问，可能是由于网络攻击导致的。以下是一些常见的网络攻击类型以及如何检测和处理这些攻击的方法：可能的攻击类型及检测方法：1.分布式拒绝服务攻击（DDoS）：症状：异常高的流量导致服务器过载，响应时间变长甚至无法访问。检测：检查服务器的网络流量日志，注意是否有大量的异常请求。使用流量分析工具（如Wireshark）分析网络流量。处理：配置防火墙规则限制流量，使用DDoS防护服务，如Cloudflare或AWS Shield，配置负载均衡器分散流量。2.暴力破解攻击：症状：大量的失败登录尝试导致服务器资源耗尽。检测：检查登录日志，注意是否有大量失败的登录尝试。处理：启用多因素认证（MFA），配置账户锁定策略，使用入侵检测系统（IDS）阻止恶意IP地址。3.恶意软件感染：症状：服务器性能下降，不明进程占用大量资源。检测：运行全面的反病毒和反恶意软件扫描，检查可疑进程和文件。处理：隔离受感染系统，清除恶意软件，更新所有软件和系统补丁。4.SQL注入攻击：症状：数据库查询异常，数据泄露或篡改。检测：检查Web服务器和数据库日志，注意异常的SQL查询。处理：修复应用中的SQL注入漏洞，使用参数化查询，部署Web应用防火墙（WAF）。5.跨站脚本攻击（XSS）：症状：用户数据被盗，页面行为异常。检测：检查应用的输入和输出，注意未被过滤的用户输入。处理：修复XSS漏洞，使用内容安全策略（CSP）防止脚本注入。基础检查项目资源使用情况：检查CPU、内存和磁盘使用情况，确保没有资源耗尽。网络配置：检查防火墙和路由配置，确保无未授权访问。系统和应用更新：确保所有软件和系统都是最新版本，修复已知安全漏洞。立即采取的措施隔离问题服务器：暂时从网络中移除受影响的服务器以防止进一步的攻击。分析日志和流量：快速分析日志文件和网络流量以识别攻击类型。启动应急响应计划：如果有网络安全应急响应计划，立即启动以减轻攻击影响。考虑联系专业的网络安全团队进行深入分析和防护。如果您需要进一步的详细指导或有特定日志和数据需要分析，请提供更多信息。"

A: "Yes, your server is experiencing significant lag and the login page is inaccessible, likely due to a network attack. Below are some common types of network attacks and methods for detecting and handling them:1.Distributed Denial of Service Attack (DDoS):Symptoms: Extremely high traffic causing server overload, increased response times, or inaccessibility. Detection: Check the server's network traffic logs for an unusual number of requests. Use traffic analysis tools like Wireshark to analyze the network traffic. Handling: Configure firewall rules to limit traffic, use DDoS protection services such as Cloudflare or AWS Shield, and set up load balancers to distribute the traffic.
2.Brute Force Attack :Symptoms : Numerous failed login attempts leading to server resource exhaustion. Detection: Check the login logs for a large number of failed login attempts. Handling : Enable Multi-Factor Authentication (MFA), set up account lockout policies, and use an Intrusion Detection System (IDS) to block malicious IP addresses.3.Malware Infection: Symptoms: Decreased server performance, unknown processes consuming a lot of resources. Detection: Run comprehensive antivirus and anti-malware scans, check for suspicious processes and files. Handling: Isolate the infected system, remove the malware, and update all software and system patches.4.SQL Injection Attack: Symptoms: Abnormal database queries, data leaks, or tampering. Detection: Check web server and database logs for unusual SQL queries. Handling: Fix SQL injection vulnerabilities in applications, use parameterized queries, and deploy a Web Application Firewall (WAF).5.Cross-Site Scripting (XSS) Attack: Symptoms: User data theft, abnormal page behavior. Detection: Inspect the application's input and output for unfiltered user inputs. Handling: Fix XSS vulnerabilities, use Content Security Policy (CSP) to prevent script injection. Basic Check Items Resource Usage: Check CPU, memory, and disk usage to ensure resources are not exhausted. Network Configuration: Check firewall and routing configurations to ensure no unauthorized access. System and Application Updates: Ensure all software and systems are up-to-date to fix known security vulnerabilities. Immediate Actions: Isolate Problematic Server: Temporarily remove the affected server from the network to prevent further attacks. Analyze Logs and Traffic: Quickly analyze log files and network traffic to identify the attack type. Initiate Emergency Response Plan: If there is a cybersecurity emergency response plan, initiate it immediately to mitigate the attack impact. Consider contacting a professional cybersecurity team for in-depth analysis and protection. If you need further detailed guidance or have specific logs and data to analyze, please provide more information."

Figure 11. Question (S = 0.51).

5. Conclusions

This study verified through experiments the significant impact of the three set scoring indicators D_1, D_2, and D_3 on the generation effect of the large language model. Experiments have shown that as the scores of these three indicators increase, the quality of the answers generated by the large language model improves accordingly. A particularly exciting finding is that as the comprehensive evaluation score S improves, there is a significant optimization in the generation results. This outcome suggests that the input text quality evaluation criteria designed in this paper are effective for assessing the generation effects of large language models in the field of cybersecurity, and this criterion has the potential to be applied to other vertical fields, such as medicine and law. The key difference in applying it across various fields lies in training the corresponding D_3 indicator model for the specialized vocabulary of each field.

The study of input text quality is of significant importance for improving the generative effects of large language models. While the Transformer architecture can comprehensively extract input information, variations in the way questions are posed in professional, vertical fields can lead to significant differences in the effectiveness and depth of generated results. This research, based on the three indicators proposed that affect generation quality, has constructed an input text quality standard S, which effectively evaluates input quality and thereby supports the application of large language models in fields such as cybersecurity, filling a gap in research on the quality of inputs for large language models. Additionally, this paper primarily focuses on the vertical field of cybersecurity, conducting validation experiments based on a cybersecurity knowledge base. In future research, knowledge from other fields will be gradually established to extend the applicability of this algorithm.

Future research directions and focuses will include:

1. Building more structured input models based on the existing foundation.

2. Continuously optimizing the details of the indicators so that the *S* can more accurately reflect input quality.
3. This will be applied in specialized large language models for specific vertical fields.
4. Building knowledge in other fields to promote the general applicability of this algorithm.
5. Conduct a more detailed analysis of input text research in the field of cybersecurity, including the examination of necessary hardware information and related parameters for specific issues. Additionally, integrate advanced algorithms to enhance the analysis results.

Author Contributions: Conceptualization, J.-K.Y.; Methodology, J.-K.Y.; Software, Y.-F.Y.; Validation, Y.-F.Y.; Formal analysis, Y.-F.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was supported by the National Key R&D Program of China under Grant No. 2024QY1703 and National Natural Science Foundation of China under Grant No. U1636208.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Ni, J.; Young, T.; Pandelea, V.; Xue, F.; Cambria, E. Recent advances in deep learning based dialogue systems: A systematic survey. *Artif. Intell. Rev.* **2023**, *56*, 3055–3155. [CrossRef]
2. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A survey of transformers. *AI Open* **2022**, *3*, 111–132. [CrossRef]
3. Johnson, R.; Zhang, T. Supervised and semi-supervised text categorization using LSTM for region embeddings. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 19–24 June 2016; pp. 526–534.
4. Liu, W.; Wang, Z.; Liu, X.; Zeng, N.; Liu, Y.; Alsaadi, F.E. A survey of deep neural network architectures and their applications. *Neurocomputing* **2017**, *234*, 11–26. [CrossRef]
5. Alshemali, B.; Kalita, J. Improving the Reliability of Deep Neural Networks in NLP: A Review. *Knowl.-Based Syst.* **2020**, *191*, 105210. [CrossRef]
6. Liu, G.; Guo, J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **2019**, *337*, 325–338. [CrossRef]
7. Lipton, Z.C.; Berkowitz, J.; Elkan, C. A critical review of recurrent neural networks for sequence learning. *arXiv* **2015**, arXiv:1506.00019.
8. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
9. Gillioz, A.; Casas, J.; Mugellini, E.; Abou Khaled, O. Overview of the Transformer-based Models for NLP Tasks. In Proceedings of the 2020 15th Conference on Computer Science and Information Systems (FedCSIS), Sofia, Bulgaria, 6–9 September 2020; IEEE: New York, NY, USA, 2020; pp. 179–183.
10. Rahman, W.; Hasan, M.K.; Lee, S.; Zadeh, A.; Mao, C.; Morency, L.P.; Hoque, E. Integrating multimodal information in large pretrained transformers. In Proceedings of the Conference. Association for Computational Linguistics. Meeting, Online, 5–10 July 2020; Volume 2020, p. 2359.
11. Ganesan, A.V.; Matero, M.; Ravula, A.R.; Vu, H.; Schwartz, H.A. Empirical evaluation of pre-trained transformers for human-level NLP: The role of sample size and dimensionality. *Proc. Conf. Assoc. Comput. Linguist. N. Am. Chapter Meet.* **2021**, *2021*, 4515.
12. Srivastava, A.; Rastogi, A.; Rao, A.; Shoeb, A.A.M.; Abid, A.; Fisch, A.; Wang, G. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv* **2022**, arXiv:2206.04615.
13. Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Lowe, R. Training language models to follow instructions with human feedback. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 27730–27744.
14. Dong, X.; Zhang, C.; Ge, Y.; Mao, Y.; Gao, Y.; Lin, J.; Lou, D. C3: Zero-shot Text-to-SQL with ChatGPT. *arXiv* **2023**, arXiv:2307.07306.
15. Xiao, L.; Chen, X. Enhancing llm with evolutionary fine tuning for news summary generation. *arXiv* **2023**, arXiv:2307.02839.
16. Wu, C.; Yin, S.; Qi, W.; Wang, X.; Tang, Z.; Duan, N. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv* **2023**, arXiv:2303.04671.
17. Dong, Y.; Jiang, X.; Jin, Z.; Li, G. Self-collaboration Code Generation via ChatGPT. *arXiv* **2023**, arXiv:2304.07590. [CrossRef]
18. Wei, X.; Cui, X.; Cheng, N.; Wang, X.; Zhang, X.; Huang, S.; Han, W. Zero-shot information extraction via chatting with chatgpt. *arXiv* **2023**, arXiv:2302.10205.

19. Azaria, A. *ChatGPT Usage and Limitations*; Ministère de L'enseignement Supérieur et de la Recherche: Paris, France, 2022.
20. Yu, Y.J.; Liu, Q. N-gram Chinese Characters Counting for Huge Text Corpora. *Comput. Sci.* **2014**, *41*, 263–268.
21. Xiong, H.; Wang, S.; Zhu, Y.; Zhao, Z.; Liu, Y.; Huang, L.; Shen, D. Doctorglm: Fine-tuning your Chinese doctor is not a herculean task. *arXiv* **2023**, arXiv:2304.01097.
22. Cui, J.; Li, Z.; Yan, Y.; Chen, B.; Yuan, L. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *arXiv* **2023**, arXiv:2306.16092.
23. Li, X.; Xie, H.; Li, L.J. Research on Sentence Semantic Similarity Calculation Based on Word2vec. *Comput. Sci.* **2017**, *44*, 256–260.
24. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on genetic algorithm: Past, present, and future. *Multimed. Tools Appl.* **2021**, *80*, 8091–8126. [CrossRef] [PubMed]

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