INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research

International Journal of Continuing Engineering Education and Life-Long Learning

ISSN online: 1741-5055 - ISSN print: 1560-4624
https://www.inderscience.com/ijceell

Optimisation of intelligent English grammar error correction based on multi-strategy Pinyin detection and hierarchical enhancement

Lingling Song

DOI: 10.1504/IJCEELL.2026.10077491

Article History:

Received:	13 November 2025
Last revised:	28 January 2026
Accepted:	29 January 2026
Published online:	20 April 2026

Optimisation of intelligent English grammar error correction based on multi-strategy Pinyin detection and hierarchical enhancement

Lingling Song

Open Education Academy,
Yantai Vocational College,
Yantai, 264003, China
Email: SOLL917@163.com

Abstract: This study proposes an intelligent grammar correction method integrating multi-strategy Pinyin detection and hierarchical data augmentation to address common errors in Chinese English learners' writing. A dual-strategy Pinyin detection algorithm combines syllable tree matching and linguistic rules to accurately identify and preserve Pinyin segments. A hierarchical data augmentation approach employs rule-based and model-based back-translation to build diverse parallel corpora targeting typical learner errors. Based on the Transformer architecture, the grammar correction model treats error correction as a sequence-to-sequence task. Results show the Pinyin detector achieves 99.95% accuracy, processing 5,386 words/second with 13.02 MB memory usage. The correction model attains a 40.58 F_{0.5} score and 49.56% accuracy on CoNLL-2014. On CLEC subsets, it achieves 87.5%, 90.2%, and 85.9% accuracy for article, subject-verb agreement, and verb tense errors, respectively. Pinyin false corrections dropped from 65% to 1.8%, demonstrating significant improvement in handling Chinese learners' English writing.

Keywords: grammar error correction; GEC; Pinyin detection; data augmentation; Transformer; English.

Reference to this paper should be made as follows: Song, L. (2026) 'Optimisation of intelligent English grammar error correction based on multi-strategy Pinyin detection and hierarchical enhancement', *Int. J. Continuing Engineering Education and Life-Long Learning*, Vol. 36, No. 8, pp.21–48.

Biographical notes: Lingling Song obtained her MA in Literature from Ocean University of China in 2010. Presently, she is working as an Associate Professor in the Open Education Academy, Yantai Vocational College, China. She is qualified as a thesis defence chairperson for the English major at the Open University of China, and Key Lecturer of the English teaching team at Shandong Open University. She has delivered numerous academic lectures on English syntax, grammar, pragmatics and other related fields. She has published her research findings in more than ten well-known domestic journals. Her areas of interest include English linguistics and British and American literature.

1 Introduction

Against the backdrop of rapid globalisation and the swift development of information technology, English, as an international lingua franca, plays a crucial role in academic exchanges, business communications, and cross-cultural collaborations, where the accuracy and standardisation of its written expression are of paramount importance. However, for the vast majority of non-native English learners, due to differences in language habits, thinking patterns, and grammatical rules, making grammatical errors in English writing has become a prevalent and challenging issue (Jackson et al., 2025; Aravind et al., 2025). These errors not only compromise the readability and professionalism of the text but may also lead to ambiguities and misunderstandings in communication. In recent times, with breakthroughs in deep learning technology, particularly the application of large-scale pre-trained language models based on the Transformer architecture, the performance of grammar error correction (GEC) systems has seen significant improvement. These models can learn complex linguistic patterns from massive corpora, thereby effectively detecting and correcting various types of grammatical errors (Wang et al., 2023; Shao and Liu, 2024). Nevertheless, most existing models are designed with a general-purpose approach, and their performance still leaves considerable room for improvement when dealing with specific errors made by Chinese English learners. Influenced by negative transfer from their native language, Chinese learners often make errors such as misuse of articles, subject-verb disagreement, incorrect preposition choices, and spelling mistakes directly resulting from Chinese Pinyin thinking. These errors exhibit distinct regional and group characteristics. Such errors have unique forms and often trigger a chain reaction of grammatical anomalies. However, existing systems tend to treat them merely as spelling issues, failing to conduct in-depth detection and collaborative correction from the perspective of phonetic-form associations (Qing, 2024).

For example, when Chinese learners input sentences containing Pinyin proper nouns (such as “My friend Zhangsan lives in Beijing”), the Pinyin strings ‘Zhangsan’ and ‘Beijing’ have a high probability of being misjudged as English spelling errors. The system may erroneously ‘correct’ it as meaningless character fragments (such as’ Zhangsan’) or completely unrelated English words (such as’ Changing sun’). This misjudgement not only fails to correct actual errors, but also introduces new errors, completely distorting the original meaning of the sentence, seriously damaging the integrity of the text, user experience, and trust in the automatic correction system. Statistics show that about 38% of Chinese learners’ English compositions contain such Pinyin proprietary terms, of which 72% of Pinyin sequences are misjudged as spelling errors by existing mainstream GEC systems and undergo invalid corrections (Song et al., 2023). This not only leads to distorted text expression, but also may mislead learners into forming incorrect language cognition, seriously affecting the practicality of GEC systems in Chinese learner scenarios. Therefore, the development of efficient and precise automatic English GEC technology has emerged as a highly valuable and promising research direction in the fields of computational linguistics and educational technology.

In terms of Chinese spelling error detection, Wang et al. (2021) proposed a fused qualified LSTM model for spelling errors caused by Pinyin input methods. This model end-to-end fuses characters, words and Pinyin features, achieving better detection results than the baseline method on the dataset, demonstrating the importance of Pinyin information in the discrimination of Chinese spelling errors. In the field of word

representation learning, Zhang et al. (2021) proposed a probabilistic representation model based on feature subsequences. This model integrated multiple internal features of Chinese characters such as strokes, structure, and Pinyin, and captured the multi-dimensional semantics of words through feature subsequences and probability distributions. It performed excellently in tasks such as word class ratio and text classification. It provided new ideas for Chinese semantic modelling based on multi-feature fusion.

To address the issues of homophonic typos evading detection and data sparsity in Chinese malicious long-tail keyword detection, Sun and Zhang (2024) introduced a detection model that integrates radical and Pinyin attention. This method trained embedding vectors for Chinese characters, radicals, and Pinyin separately using Word2Vec and employed a parallel co-attention network to fuse multimodal information. Experiments on multiple short-text datasets demonstrated that the model improved the recognition accuracy of malicious long-tail keywords. To tackle the problem that existing Chinese spelling correction models failed to effectively utilise multimodal information for handling different types of errors, Li et al. (2025) proposed a dynamically reweighted multimodal spelling correction model. This model introduced a dynamic reweighting module to optimise inter-modal interactions and adopted an independent modal masking strategy to enhance model robustness during the pre-training phase. Experimental results showed that the model could effectively model inter-modal interactions and remain robust against erroneous modal information. To enhance the modelling effectiveness of Chinese characters as target units in end-to-end Mandarin speech recognition systems, Li et al. (2023) proposed enhancing character-based automatic speech recognition systems by incorporating Chinese Pinyin representations. They constructed four new frameworks that fused Pinyin and character information and trained them under a multi-task framework. Experiments indicated that this method significantly reduced the character error rate, with a relative reduction of up to 6.4%. To address the issue of spelling errors affecting sentiment polarity prediction in Chinese sentiment analysis, Li et al. (2024) fused glyph and Pinyin information for feature matrix learning. By iteratively mining character, glyph, and Pinyin features and dynamically weakening the influence of interfering words through a combination of soft attention and matrix composition modules, the model's robustness to noise was enhanced. Experiments showed that the model effectively utilised multisource information and improved the performance of Chinese sentiment analysis. To achieve multi-speaker Chinese news broadcasting and real-time voice timbre switching, Zhao et al. (2023) proposed a speech synthesis system based on an improved Tacotron2. This system accurately converted text into Pinyin phonemes and employed a timbre encoder to construct speaker embeddings. Experiments demonstrated that the system could synthesise highly natural speech with target timbres and styles under limited data conditions, making it suitable for real-time news broadcasting.

To address the issues of complex rules, data sparsity, and insufficient context utilisation in traditional GEC systems, Jiang (2025) proposed a transformer-based artificial intelligence model for English GEC. This model leveraged the powerful context understanding and feature extraction capabilities of pre-trained language models, combined with large-scale datasets and data augmentation techniques to enhance generalisation ability. Experimental results demonstrated that the model exhibited excellent GEC capabilities and practical potential. To tackle the problems of over-correction, poor multilingual processing, and weak domain adaptability in existing

GEC systems, Alapati et al. (2025) proposed a Transformer-based hybrid error detection and correction framework. This framework integrated a Seq2Seq structure, attention mechanisms (AMs), and error-specific layers, and employed data augmentation techniques such as back-translation to enhance system robustness. The results showed that the model achieved an accuracy rate of 99% and performed excellently in multilingual and domain-specific scenarios. Addressing the characteristics of local and cross-segment distribution of grammatical errors, Zhao (2024) proposed an English GEC method based on AM-driven machine translation, treating the error correction process as a monolingual translation task. Transformer was adopted as the encoder-decoder framework to simultaneously capture local context and long-range dependencies. Experiments indicated that the Transformer model significantly outperformed baseline approaches grounded in recurrent neural networks or convolutional neural networks. To achieve precise multilingual GEC, Kumar et al. (2024) proposed a graph convolutional model that combined context awareness with adversarial learning. This model integrated the multilingual embedding generation capabilities of multilingual T5, modelled dynamic dependencies between words using temporal graph convolutional networks, and introduced adversarial learning to enhance cross-language generalisation performance. Experiments showed that the model achieved significant improvements over existing methods, particularly in resource-scarce languages and complex syntactic environments. To address the issue of verb form error detection in English articles, Hu and Zhang (2024) employed a bidirectional long short-term memory network for verb error identification. By utilising textual context information, the model demonstrated high accuracy and stability in part-of-speech tagging and verb form error detection. Simulation experiments revealed that this method outperformed support vector machines and rule-based approaches in error type identification.

In summary, when dealing with Pinyin proper nouns, the traditional name-library matching method fails to adapt to domestic learner scenarios due to contradictions in lexicon scale, excessively high time and space complexity, and risks of misjudgement and omission. Although single Pinyin detection algorithms eliminate reliance on lexicons, the former suffers from insufficient accuracy in zero-initial consonant detection, while the latter exhibits slow execution speed. Traditional GEC models mostly adopt end-to-end architectures, constrained by the low match between publicly available datasets and error types in domestic exams, and they lack hierarchical optimisation for different error granularities. To address these issues, the study innovatively constructs an intelligent English GEC framework that integrates multi-strategy Pinyin detection and hierarchical data augmentation. In the Pinyin detection phase, it combines syllable tree matching and linguistic rule matching, paired with a hierarchical decision-making mechanism and a length filtering mechanism, to achieve precise identification of Pinyin vocabulary in texts. Data augmentation employs a hierarchical strategy, generating parallel corpora through rule-driven insertion, deletion, and substitution operations, as well as model-driven constrained back-translation, while ensuring corpus quality through filtering and hierarchical sampling. The core GEC component adopts a Transformer encoder-decoder architecture and also integrates a feedback filtering mechanism.

2 Methods

The study first constructs a multi-strategy Pinyin detection algorithm to address the issue of misjudgement of Pinyin vocabulary in texts produced by Chinese learners. Secondly, the hierarchical data augmentation strategy tackles the challenge of scarce training data by automatically generating high-quality and diverse parallel error-correction corpora through a combination of rule-driven and model-driven approaches. Finally, the GEC model architecture is based on an advanced transformer encoder-decoder structure, responsible for performing sequence-to-sequence conversion tasks from erroneous sentences to correct sentences.

2.1 Multi-strategy Pinyin detection algorithm

As the preprocessing module of the entire grammar correction system, the core objective of the multi-strategy Pinyin detection algorithm is to solve the confusion problem between Pinyin proper nouns and English spelling errors in the texts of Chinese learners. This module accurately identifies and protects Pinyin fragments, preventing them from being wrongly modified into English words in the subsequent general error correction process. The algorithm in this section will provide a noise-free basic text input for the subsequent error correction model, thereby ensuring the pertinence and practicality of the system.

In the writing scenarios of Chinese English learners, proper nouns such as personal names and place names often need to be presented in Pinyin form. While such Pinyin spellings are not inherently incorrect, traditional GEC systems lack the targeted recognition ability and often misjudge them as English spelling errors, making inappropriate corrections. This significantly affects user experience and system credibility (Liu et al., 2023; Sun et al., 2024). However, this method has inherent drawbacks, including limited lexicon coverage and inefficient storage and retrieval. To overcome these limitations, the study proposes a dual-strategy detection algorithm that integrates rapid syllable tree matching with Pinyin rule lookup.

The total number of Chinese Pinyin syllables is limited, but their combinations are complex. If traditional dictionary matching methods are employed, an extremely large lexicon would need to be constructed, resulting in inefficient storage and retrieval (Arundathi and Satyanarayana, 2024). To improve detection efficiency, this study constructs a syllable tree structure with initial consonants as the root node, for efficient matching of Pinyin strings. The construction of the syllable tree is based on the standard syllable system stipulated in the ‘scheme for the Chinese phonetic alphabet’, covering 23 initial consonants (including zero initial consonants), 35 final vowels (including single final vowels, compound final vowels, and nasal final vowels), and legal syllable combinations, without tone information – because the Pinyin proper nouns used by Chinese learners in English writing are usually not marked with tones. Moreover, the tone does not affect the recognition and judgment at the spelling level. The construction process is as follows:

1 Root node initialisation: 23 initial consonants (b, p, m, f, d, t, n, l, g, k, h, j, q, x, zh, ch, sh, r, z, c, s, y, w) and zero initial consonants (marked with ‘∅‘) are used as root nodes to construct the first-level node layer

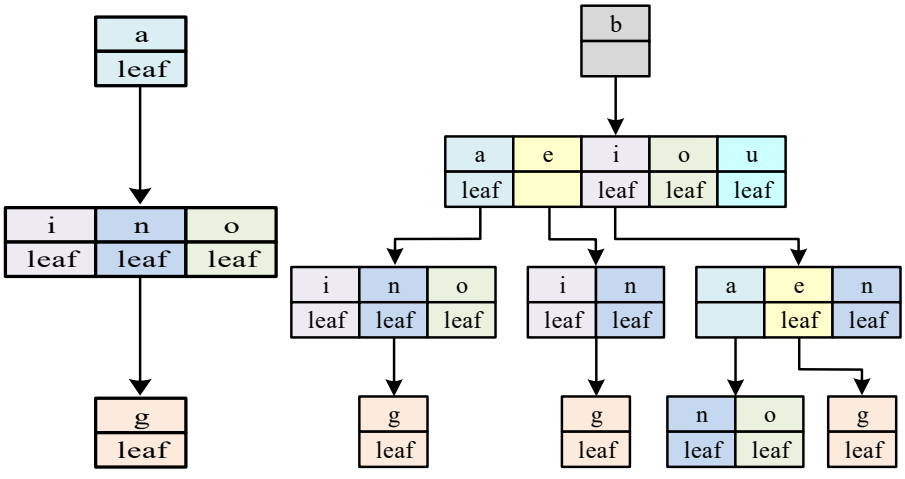
2 Expansion of vowel nodes: For each root node of an initial consonant, based on the legal vowel combinations of the initial consonant in the 'Hanyu Pinyin Scheme', the corresponding child nodes of the vowel are added in sequence (for example, expand the child nodes 'a', 'o', 'i', 'u', etc. under the initial consonant 'b' to form syllable paths such as 'ba', 'bo', 'bi', 'bu', etc.)

3 Terminal node marking: when a certain path fully corresponds to a standard Pinyin syllable (for example, 'b' → 'e' → 'i' corresponds to 'bei'), the final node of this path is marked as the terminal node.

4 Zero-initial syllable processing: for zero-initial syllables (such as 'a', 'ai', 'an', etc.), the process directly uses the vowel as the core path, extends from the zero-initial root node 'ø' to construct child nodes and mark the terminal.

Its structure is illustrated in Figure 1. According to Figure 1, during detection, the algorithm performs a breadth-first traversal of the string under test. If the character sequence can fully match a path from the root node to a certain terminal node, and the end marker of that terminal node is true, the string is judged to be Pinyin.

Figure 1 Syllable tree structure, (a) the vowel a is the root node, (b) the vowel b is the root node (see online version for colours)



(a) (b)

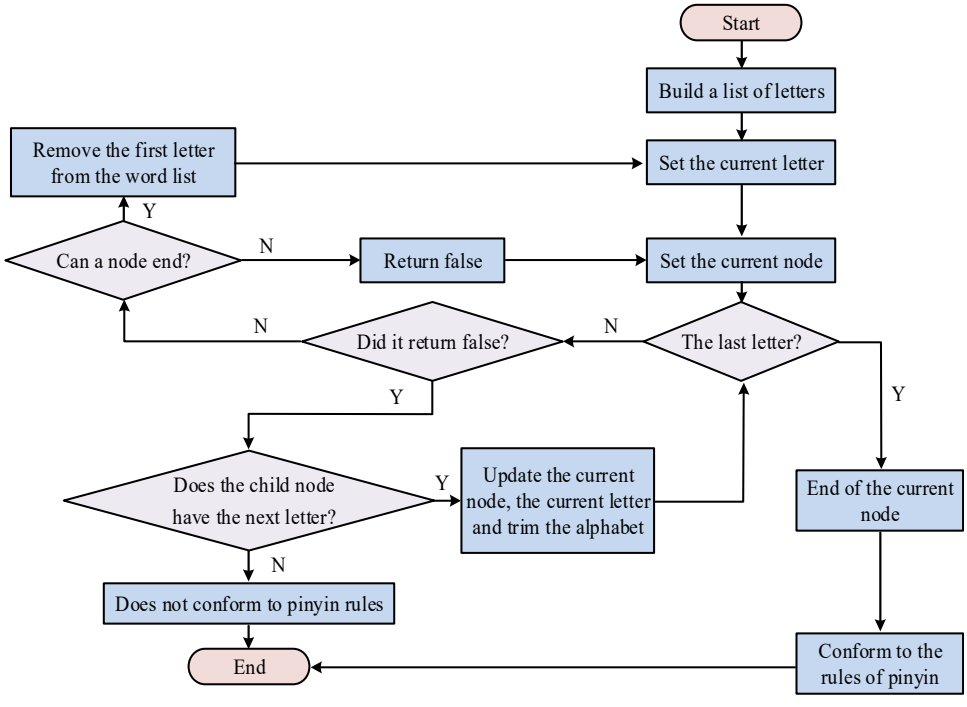
The core concept of the syllable tree-based Pinyin detection algorithm is as follows: Based on the syllable structure of Mandarin Chinese, a tree-shaped data structure with initial consonants as root nodes is constructed. Through breadth-first search (BFS) traversal of this tree structure, the algorithm verifies whether the word under detection conforms to syllable combination rules (Das et al., 2024). This algorithm does not rely on any external lexicon; it accomplishes Pinyin determination solely through predefined syllable logic. Figure 2 illustrates the detection process of the syllable tree-based Pinyin detection algorithm.

According to Figure 2, firstly, the syllable tree structure is defined as $T = (V, E)$, where V represents the set of nodes and E represents the set of edges. Each node $v \in V$ contains three attributes: the current character *v.char*, the child node mapping *v.children*,

and the end flag *v.is_end* (a Boolean value indicating whether the path from the root node to the current node constitutes complete Pinyin).

The initial consonant set is denoted as $R = \{r_1, r_2, \ldots, r_3\}$, encompassing 23 standard initial consonants (including the zero initial consonant). For each initial consonant r_i, an independent syllable subtree T_i is established. The syllable insertion process adheres to the following rules: for a given Pinyin syllable $s = c_1, c_2, \ldots, c_k$, the process commences from the root node $r = c_1$, and each character c_j ($j = 2, 3, \ldots, k$) is processed sequentially. If the character c_j does not exist among the child nodes of the current node, a new node is created. After all characters have been processed, the *is_end* attribute of the terminal node is set to true.

Figure 2 Flowchart of syllable tree pinyin detection (see online version for colours)



The detection algorithm adopts the BFS strategy. For the string $S = c_1, c_2 \ldots c_n$ to be detected, the corresponding mother child tree is first determined based on the first character c_1. Then, the character sequence is recursively checked: at the current node v, the next character c_j is processed. If $c_j = v.children$, it is transferred to the child node $v.children[c_j]$; otherwise, the matching fails. When all characters are processed, check the *is_end* attribute of the final node. If it is true, the match is successful (Kavros and Tzitzikas, 2023; Mangotra et al., 2024).

To compensate for the deficiency of the syllable tree method in recognising unlisted Pinyin forms, the study introduces a matching algorithm based on linguistic rules. This algorithm, grounded in the phonological rules of Chinese Pinyin, employs a recursive descent parsing strategy, which can be formally represented as equation (1).

$$Syllable = [Initial\ Consonant] + [Medial\ Glide] + Final \tag{1}$$

The definition of this rule set strictly adheres to the phonetic norms of the 'Hanyu Pinyin Scheme' and the modern Hanyu Pinyin spelling rules. The syllable structure only supports three types: 'initial consonant + intermediate consonant + final', 'initial consonant + final', and 'zero initial consonant + final'. Illegal combinations of initial consonants and final are not allowed. The initial consonant set is C_{init} = {b, p, m, f, d, t, n, l, g, k, h, j, q, x, zh, ch, sh, z, c, s, y, w, r}, the intermediate consonant set is $C_{med} = \{i, u, ü\}$ (in practical processing, v is used instead of $ü$), and the vowel set includes a, o, e, ai, ei, ao, ou, an, en, ang, eng, ong, ia, ian, iang, uang waiting for 39 standard vowels. The spelling rules follow the basic norms of Pinyin spelling. For example, 'j, q, x' should not be followed by the base form of 'u', but should be written as 'u'. After 'n' and 'l', a 'u' prototype can be followed, etc.

The rule matching algorithm adopts a recursive descent parsing strategy. The detection process of the matching function $M(S)$ for the input string S is defined as follows: First, all possible prefix partitions $P = \{p_1, p_2, .., p_m\}$ are generated, where each prefix p_i must be a valid Pinyin syllable. For each candidate prefix p_i, $M(S')$ is recursively called on the remaining substring $S' = S[|p_i|:]$. If there is a partition that completely divides the entire string and each part is a valid Pinyin, then the match is successful. To boost the efficacy of the algorithm, the study adopted memory technology to avoid repeatedly calculating the matching results of the same substring. Based on the normative requirements of Hanyu Pinyin, the letter expression length of a complete syllable (including initial consonants, intermediate sounds, and final vowels) is limited. The longest syllable form (such as 'Zhuang', 'Jiang') has 6 letters (Liu et al., 2021). Therefore, to eliminate meaningless matching of extremely long strings and improve the efficiency and robustness of the algorithm, this study sets the maximum recursion depth limit to $D_{max} = 6$.

The syllable tree matching algorithm and the rule-based matching algorithm exhibit significant complementarity in terms of performance and applicable scenarios. The syllable tree algorithm offers high speed and a low rate of word misjudgements, but its matching of zero-initial syllables relies on the tree structure. The rule-based matching algorithm has low memory consumption and precise detection of zero-initial syllables, yet its recursive process results in slower speed. To fully leverage the advantages of both algorithms, the study devises a hierarchical decision-making mechanism to integrate the two detection methods. Let $D(S)$ denote the final detection result for string S, and its decision-making process is formally represented as equation (2).

$$D(S) = TreeMatch(S) \vee RuleMatch(S)\} \tag{2}$$

In equation (2), $TreeMatch(S)$ represents the syllable tree matching result, and $RuleMatch(S)$ represents the rule matching result. The specific implementation process is as follows: For the input string, the efficient syllable tree algorithm is employed for matching first. If the matching fails, the rule-based matching algorithm with stronger generalisation capability is then activated. This collaborative strategy ensures high detection efficiency while significantly improving the recognition accuracy for various Pinyin variants.

To enhance the system's practicality, the study also introduces a length filtering mechanism. For strings exceeding L_{max} = 8 characters in length, a negative result is directly returned, as this surpasses the reasonable length range for Chinese Pinyin. Meanwhile, for single-character strings, a special processing procedure is adopted, where

the character is only recognised as Pinyin if it belongs to the zero-initial syllable set C_{zero} = {a, o, e, ai, ei, ao, ou, an, en, ang, er}. This multi-strategy integrated approach ensures both detection efficiency and the ability to recognise various Pinyin variants, providing reliable pre-processing functionality for the subsequent GEC module.

The core process of the syllable tree matching algorithm is to perform BFS traversal on the input string. Let the length of the input string be L, and the maximum branching factor of the syllable tree be a constant b (determined by the number of legal vowels). In the worst-case scenario, the algorithm needs to check all possible paths starting from the root node, but the matching of each path is a character-by-character linear process. Therefore, its worst-case time complexity is $O(b \cdot L)$. Since b is a fixed constant, the actual time complexity can be simplified to $O(L)$, that is, it is linearly related to the input length. The rule matching algorithm attempts to perform all possible splits on the string to match the Pinyin rules. In the worst-case scenario (such as when the string is composed entirely of single letters and all may become zero initial syllables), its time complexity grows exponentially and theoretically can reach up to $O(k^L)$ (where k is the average number of possible syllable divisions at each position). However, the setting of the maximum recursion depth D_{max} = 8 strictly limits the search space within a constant range, reducing the actual worst-case time complexity to $O\left(k^{D_{max}}\right)$.

2.2 Hierarchical data augmentation strategy

After effectively controlling the input noise through the Pinyin detection module, how to enable the error correction model to accurately learn and correct the typical grammatical errors of Chinese learners has become the key. However, high-quality and large-scale parallel error correction corpora for this specific group are relatively scarce. To this end, this section proposes a hierarchical data augmentation strategy. This strategy aims to automatically generate training data that conforms to the error patterns of Chinese learners, providing sufficient and high-quality training resources for subsequent error correction models. It is the core for enhancing the domain adaptability of the model.

The rule-driven reinforcement layer is grounded in linguistic theory and error analysis, and it devises generative rules targeting common error types among Chinese English learners. Suppose the original set of correct sentences is denoted as $D_{clean} = \{s_1, s_2, \ldots, s_N\}$, where s_i represents the i^{th} grammatically correct sentence. Here, three fundamental editing operations are introduced for the purpose of generating incorrect sentences. The first type is the random insertion operation, which inserts interference words at random positions within the sentence with a probability of p_{ins} = 0.2. Assuming a sentence $s = w_1 w_2, \ldots, w_m$, a word w_{noise} is randomly selected from the pre-constructed interference word list V_{noise} and inserted at position k, as illustrated in equation (3).

$$s_{err} = w_1 \ldots w_k \oplus w_{noise} \oplus w_{k+1} \ldots w_m \tag{3}$$

In equation (3), $\oplus$ represents the concatenation operation, and $k \sim U(1, m)$ denotes a randomly selected insertion position from a uniform distribution. The second operation is the random deletion operation, which randomly deletes words from the sentence with a probability of p_{del} = 0.2, as shown in equation (4).

$$s_{err} = w_1 \ldots w_{k-1} \oplus w_{k+1} \ldots w_m \tag{4}$$

The third type is the random replacement operation, which is carried out with a probability of p_{sub} = 0.6 and encompasses two strategies: simple replacement and syntax error replacement. For simple replacement, replacement words are randomly selected from the set V_{noise}. Grammar error replacement is generated based on part-of-speech tagging and morphological variations. In the case of grammar error replacement, a part-of-speech tagging tool is first utilised to obtain the sentence's part-of-speech sequence $POS(s) = [pos_1, pos_2, \ldots, pos_m]$. Corresponding replacement rules are then designed for specific error types t (including article errors, preposition errors, noun count errors, and verb form errors). For noun count errors, the positions of all noun labels marked as $pos_i = NN$ or $pos_i = NNS$ are detected, and singular-to-plural or plural-to-singular conversions are performed with a probability of p_{noun} = 0.3, as demonstrated in equation (5).

$$w_i' = \begin{cases} \text{singularize}(w_i) & \text{if } pos_i = NNS \\ \text{pluralize}(w_i) & \text{if } pos_i = NN \end{cases} \tag{5}$$

For verb form errors, the verb tags labelled as $pos_i \in = \{VB, VBD, VBG, VBN, VBP, VBZ\}$ are detected, and erroneous verb forms are generated with a probability of p_{verb} = 0.4, as shown in equation (5).

$$w_i' = \text{generate wrong verbform}(w_i, pos_i) \tag{5}$$

The setting of the probability parameters for the above three basic editing operations aims to simulate the random distribution of errors in learners' writing, while avoiding excessive distortion of the original sentence to prevent it from losing readability. Firstly, the study referred to previous research on the distribution of grammatical errors, which indicated that at the sentence level, the occurrence density of unintentional spelling or grammatical errors was usually low; Secondly, through preliminary experiments on a small-scale validation set, the research found that when the comprehensive editing probability of a single sentence (i.e., the proportion of words expected to be modified) was controlled at 15%–20%, a good balance could be achieved between introducing sufficient diversity of errors and maintaining the basic structure and semantic integrity of the sentence.

The model-driven augmentation layer employs neural machine translation models to generate more natural and diverse erroneous sentences. Based on the back-translation approach, correct sentences are first translated into an intermediate language and then translated back into English, thereby introducing naturally occurring grammatical errors (Lee et al., 2025). Let the English-to-Chinese translation model be denoted as $M_{en \to zh}$, and the Chinese-to-English translation model as $M_{zh \to en}$. For a correct sentence s, the process of generating erroneous sentences is represented as shown in equation (6).

$$s_{err} = M_{zh \to en}\left(M_{en \to zh}(s)\right) \tag{6}$$

To enhance the specificity and diversity of error generation, the study adopts a constrained back-translation strategy. First, the distribution of grammatical errors is analysed to identify a set of key error types, denoted as T_{focus}. During the backtranslation process, the study adjusted the sampling temperature parameter τ to 1.2. A temperature slightly above 1 would smooth the probability distribution of the model output, encouraging the decoder to explore more diverse but slightly less likely primifiers,

thereby avoiding the generation of overly conservative or repetitive error patterns and increasing the diversity of errors. To guide the model to generate specific types of errors, a mapping table from error types to inducible prefixes was constructed. For example, to encourage. The generation of article errors, 'A' or 'the' can be used as a soft prefix during decoding. To encourage subject-verb inconsistency, a wrong prefix can be specified where a third-person singular subject is followed by the base form of a verb. These prefixes, as initial decoding conditions, are integrated into the generation process through beam search, significantly increasing the occurrence frequency of the target error type. The mathematical expression of the above process is shown in equation (7).

$$s_{err} = \arg\max_{s'} P\left(s' \mid M_{en\to zh}(s)\right) \cdot \mathbb{I}\left(s' \text{contains} t \in T_{focus}\right) \tag{7}$$

To ensure the quality of the augmented data, a multi-tiered filtering mechanism is established. First, the edit distance between the generated sentence s_{err} and the original sentence s is calculated, as shown in equation (8).

$$d_{edit} = \text{Levenshtein}\left(s, s_{err}\right) \tag{8}$$

Samples that meet the condition of $d_{min} \le d_{edit} \le d_{max}$ are retained, while language model perplexity is employed for filtering, as shown in equation (9).

$$PP\left(s_{err}\right) = \exp\left(-\frac{1}{n}\sum_{i=1}^{n} \log P\left(w_i \mid w_{<i}\right)\right) \tag{9}$$

Samples satisfying $PP(s_{err}) \le \alpha\ PP(s)$ are retained, where $\alpha = 0.2$ represents the perplexity threshold coefficient. To maintain a balanced distribution of error types, a stratified sampling strategy is employed. Let the target distribution of error types be $P_t(t)$ and the actual distribution be $Q_t(t)$. The calculation method for the sampling weight $w(t)$ is presented as shown in equation (10).

$$w(t) = \frac{P_t(t)}{Q_t(t) + \in} \tag{10}$$

In equation (10), $\in = 10^{-6}$ is used to prevent zero division errors. During the generation of training data, the rule-driven and model-driven methods are not used in isolation but are integrated through a weighted mixed sampling strategy to balance the coverage of error types and the naturalness of the generated sentences.

Let the set of sentences generated by rule-driven be $\mathcal{D}_{rule}$ and the set of sentences generated by model-driven be $\mathcal{D}_{model}$. In each round of training data construction, the ratio of the two is controlled at λ: $(1-\lambda)$ ($\lambda = 0.7$ in this study), that is, more emphasis is placed on rule-generating data that can precisely target specific error types (Zhang et al., 2025). When sampling from $\mathcal{D}_{rule}$, the weight $w(t)$ calculated based on equation (10) is used to ensure that the actual distribution $Q_t(t)$ of the error type approximates the target distribution $P_t(t)$. Sampling from $\mathcal{D}_{model}$ is equally valued, aiming to increase the diversity of the overall data.

In the early stage of model training, the proportion of mixed data $\left(\mathcal{D}_{rule} \cup \mathcal{D}_{model}\right)$ in the training set is relatively high, aiming to enable the model to quickly grasp the key error patterns of Chinese learners. As the training progresses, it gradually reduces the

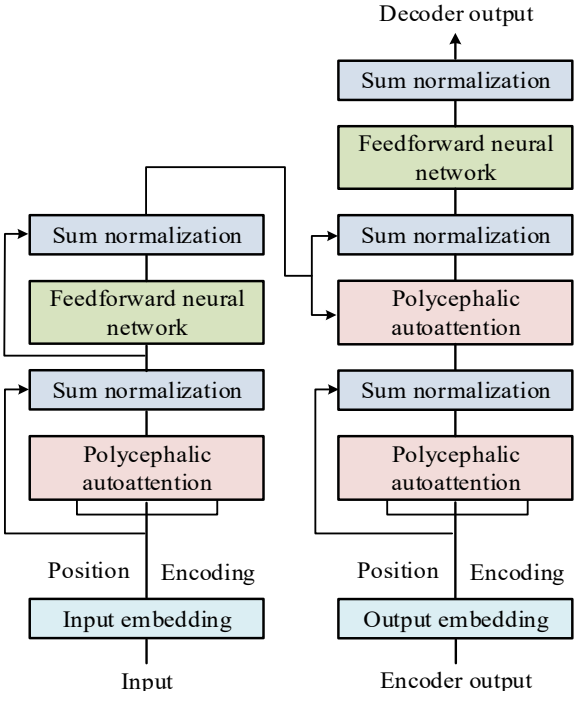
proportion of augmented data mixture in the later stage and increases the weight of the original correct sentences to promote the model's consolidation of correct grammatical patterns and prevent overfitting.

2.3 Architecture of GEC model

Based on the aforementioned Pinyin detection and data augmentation work, this section constructs the core GEC model of the system. This model formalises the grammar correction task as a sequence-to-sequence transformation task. Through the attention mechanism and autoregressive generation strategy, it achieves precise correction of grammar errors in the text after Pinyin protection and serves as the core carrier for connecting the pre-order modules and output the final error correction result.

The research adopts the encoder-decoder architecture based on Transformer as the core model for GEC. The choice of this mature architecture over designing a novel network structure is mainly due to the fact that the GEC task is essentially a sequence-to-sequence generation problem of 'translating' erroneous sentences into correct ones. Its self-attention mechanism can simultaneously capture local grammatical dependencies and long-distance semantic associations, which precisely aligns with the multi-granularity distribution characteristics of grammatical errors among Chinese learners. This architecture formalises the grammar correction task as a sequence-to-sequence text generation problem. The Transformer architecture is shown in Figure 3, mainly consisting of an encoder and a decoder. The encoder is responsible for converting the input sequence into context representations, while the decoder uses these representations to generate the output sequence (Zhong et al., 2023).

Figure 3 Transformer model structure (see online version for colours)



The model receives a source text sequence that may contain grammatical errors and outputs the corresponding corrected text sequence. Let the input sequence be denoted as $X = (x_1, x_2, \ldots, x_n)$, where n represents the sequence length, and the output sequence be denoted as $Y = (y_1, y_2, \ldots, y_m)$, where m represents the output length. The model establishes a mapping relationship between the input and output based on conditional probability, as shown in equation (11).

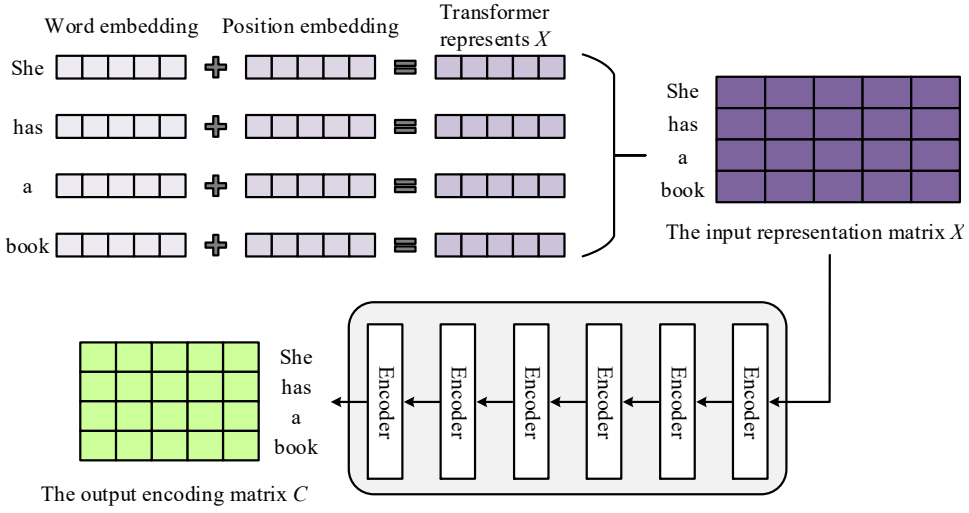
$$P(Y \mid X) = \prod_{t=1}^{m} P\left(y_t \mid y_{<t}, X\right) \tag{11}$$

In equation (11), $y_{<t} = (y_1, \ldots, y_{t-1})$ represents the generated prefix sequence. The model adopts a multi-layer Transformer structure, including $L = 12$ encoder layers and $L = 12$ decoder layers. The hidden dimension of each layer is $d_{model} = 768$, the dimension of the feedforward network is $d_{ff} = 3{,}072$, and the number of attention heads is $H = 12$. Attention calculation adopts the scaled dot product AM, as shown in equation (12).

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{12}$$

In equation (12), Q, K, and V represent the query, key, and value matrices, and $d_k = d_{model}/H = 64$ is the dimension of each attention head. The encoder is responsible for deep semantic encoding of the input sentence, as shown in Figure 4.

Figure 4 Transformer encoder encoding sentence information (see online version for colours)

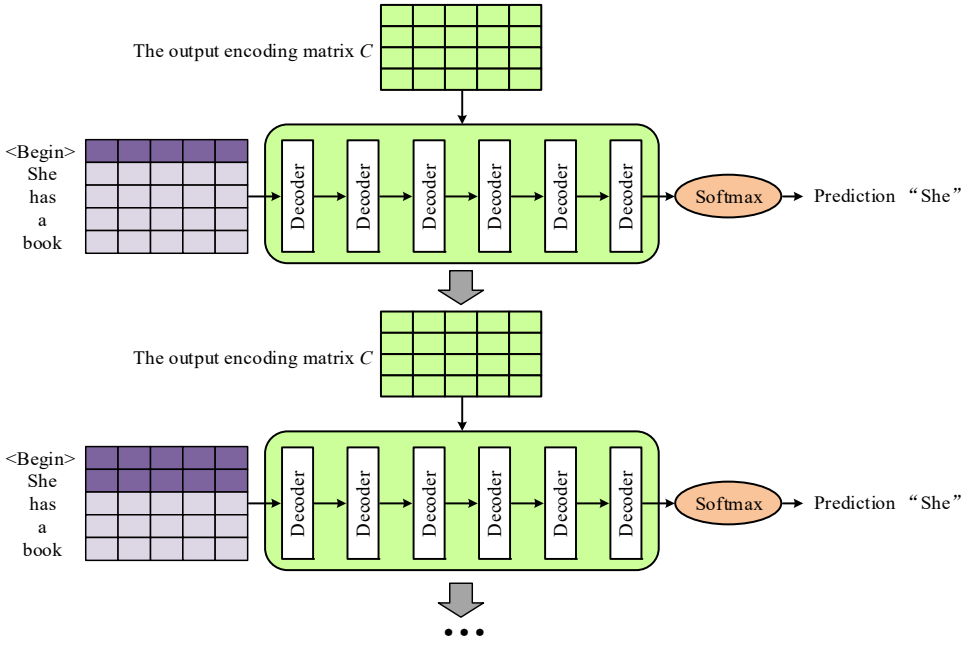


In Figure 4, the text sequence to be corrected is first transformed into a vector representation through an embedding layer, and then input into an encoder composed of multiple layers. Each encoder layer consists of two sub layers, namely a multi head self AM and a feedforward neural network. Each sub layer adopts residual connections and layer normalisation, as shown in equation (13).

$$\text{LayerNorm}(x + \text{Sublayer}(x)) \tag{13}$$

In equation (13), Sublayer(x) represents the output of the sublayer. The output of the encoder is a context aware hidden state sequence $H_{enc} = (h_1, h_2, \ldots, h_n)$, where each hidden state $h_i \in R^{d_{model}}$ contains contextual information of the entire input sequence. The decoder uses autoregression to generate the target sequence. As shown in Figure 5, the Transformer model completes the sequence prediction task of syntax correction through autoregression.

Figure 5 Transformer prediction (see online version for colours)



The decoder gradually predicts the next word based on the context encoding matrix C output by the encoder and combined with the generated partial result sequence (Jiang et al., 2023; Khabutdinov et al., 2024). In the initial state, masked attention ensures that the decoder can only access information from the first t–1 positions when generating the t^{th} word, as shown in equation (14).

$$\text{MaskedAttention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V \tag{14}$$

In equation (14), M is the mask matrix. When $i < j$, $M_{ij} = -\infty$; otherwise, $M_{ij} = 0$. The encoder decoder AM enables the decoder to focus on the relevant parts of the input sequence, as shown in equation (15).

$$\text{CrossAttention}(Q, H_{enc}) = \text{Attention}\left(Q, KH_{enc}, VH_{enc}\right) \tag{15}$$

The output layer uses the softmax function to calculate the probability distribution on the vocabulary, as shown in equation (16) (Shanthamallappa and Ravi, 2023).

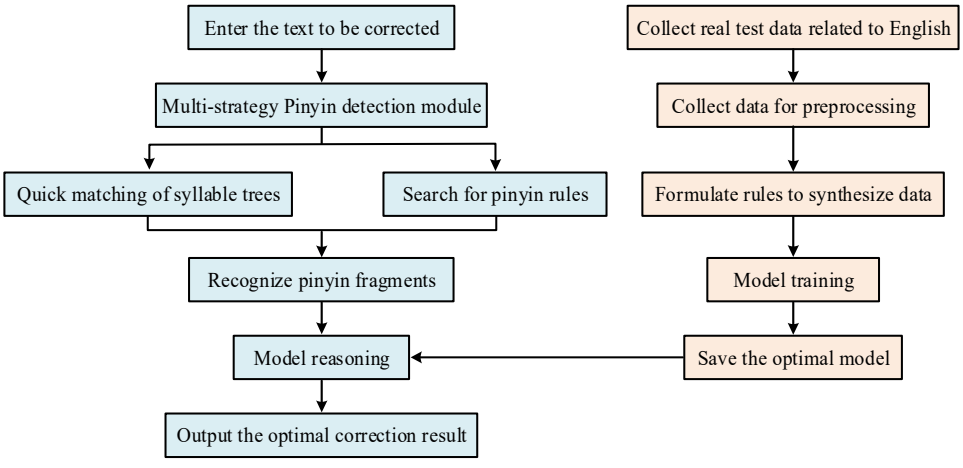
$$P\left(y_t \mid y_{<t}, X\right) = \text{softmax}\left(W_o h_t + b_o\right) \tag{16}$$

In equation (16), W_o is the output weight matrix, and h_t is the hidden state of the decoder at time step t. The goal of training is to reduce the negative log likelihood loss, as shown in equation (17).

$$\mathcal{L} = -\sum_{t=0}^{m} \log P\left(y_t^* \mid y_{<t}^*, X\right) \tag{17}$$

In equation (17), y_t^* is the true value of the t^{th} word in the target sequence. Figure 6 shows the two core stages of the grammar correction system proposed in the study.

Figure 6 Process of the grammar correction model (see online version for colours)



The overall workflow of the system commences with the user inputting the text to be corrected. Initially, the text is processed by a multi-strategy Pinyin detection module. This module employs two algorithms in coordination: rapid matching using a syllable tree and Pinyin rule lookup, to accurately identify Pinyin segments within the text and protect them from being misclassified as spelling errors in subsequent stages. After the Pinyin protection is completed, the text proceeds to a transformer-based GEC model for inference, ultimately yielding the optimal corrected result.

3 Results and analysis

A comparative analysis was conducted between the multi-strategy algorithm proposed in the study and traditional methods, focusing on key metrics such as accuracy, recall, and efficiency, to demonstrate its comprehensive advantages in identifying Pinyin vocabulary. Furthermore, the final error correction system was compared with various baseline models, and detailed ablation experiments were carried out to thoroughly investigate the contribution of each component, including the data augmentation strategy and the Pinyin detection module, to the overall system performance.

3.1 Dataset details and experimental setup

To comprehensively evaluate the performance of the multi-strategy Pinyin detection algorithm proposed in the study, a test set comprising 10,000 samples was constructed. This set included 5,000 common Pinyin strings (such as 'Zhangsan,' 'Beijing,' etc.) and 5,000 English words (such as 'apple,' 'the,' etc.), covering inputs of varying lengths (1–10 characters). For the training and performance validation of the GEC model, the Chinese learners English corpus (CLEC) and the CoNLL-2014 test sets were primarily utilised. The CLEC, jointly constructed by domestic universities, encompasses compositions from Chinese college students taking CET-4, CET-6, specialised English exams, and high school students. With a total word count exceeding 1 million, it features annotation information covering error types, part-of-speech tags, and error frequencies, accurately reflecting typical errors made by Chinese English learners (such as Pinyin confusion and article misuse). The CoNLL-2014 test set, serving as the official evaluation data for an international competition in the field of GEC, includes 28 types of grammatical errors. Its evaluation rules and scripts are publicly available and transparent, often used for cross-model performance comparisons to assess the efficacy of the suggested approach in complex error scenarios. The experimental environment and parameter settings are shown in Table 1.

Table 1 Experimental environment and parameter settings

Category	*Configuration item*	*Value/model*
Hardware environment	CPU	Intel(R) Xeon(R) Platinum 8358P CPU @ 2.60GHz
	GPU	NVIDIA A40 (48GB) * 1
	Memory	80GB
Software environment	Operating system	Ubuntu 20.04
	Deep learning framework	PyTorch 1.11.0
	Programming language	Python 3.8
	CUDA toolkit	CUDA 11.3
Training hyperparameters	Iterations (Epochs)	10
	Optimiser	AdamW
	Learning rate	1e-4
	batch size	100

3.2 Comparison of Pinyin detection performance

Table 2 compares the comprehensive performance of the traditional name library matching method, the single-syllable tree matching algorithm, the Pinyin rule matching algorithm, and the multi-strategy Pinyin detection algorithm proposed in the research. As can be seen from Table 2, the multi-strategy Pinyin detection algorithm performs evenly and excellently in multiple indicators. The fundamental reason for this lied in its integration of the dual strategies of rapid matching of syllable trees and parsing of linguistic rules, supplemented by hierarchical decision-making and length filtering mechanisms. Compared with the high time consumption of 116.76 seconds and large memory usage of 94.12 MB caused by the traditional name library matching method due

to its reliance on a huge static word library, the multi-strategy algorithm, by combining dynamic tree structure with rule parsing, controlled the memory usage at 13.02 MB, increasing the processing speed to 5,386 words per second while maintaining a low storage overhead. Although the execution time of the multi-strategy algorithm (0.1856 s) was much better than that of the name library matching (116.76 s) and rule matching (0.3401 s), it was still slightly higher than that of the single-syllable tree algorithm (0.0729 s). This was because the syllable tree failed to cover complex or non-standard Pinyin strings. The activated rule matching algorithm adopted recursive descent parsing, and its time complexity increased with the string length in the worst case. This was the inherent cost it incurred in exchange for high generalisation ability. However, in terms of detection accuracy, this algorithm achieved a Pinyin detection accuracy rate of 99.95%, with false alarm rates and missed alarm rates of only 0.48% and 0.06% respectively, significantly outperforming single-strategy syllable tree or rule matching methods. This was mainly attributed to its hierarchical decision-making process: Priority was given to using an efficient syllable tree for initial matching. If it failed, a rule matching with a wider coverage was initiated, and a length filtering mechanism was combined to eliminate illegal Pinyin combinations. Thus, under the premise of ensuring high-speed processing, high coverage and high discrimination for various Pinyin variants were achieved, effectively solving the problems of misjudgement and missed judgment caused by the limitations of the word bank or the single rule in traditional methods.

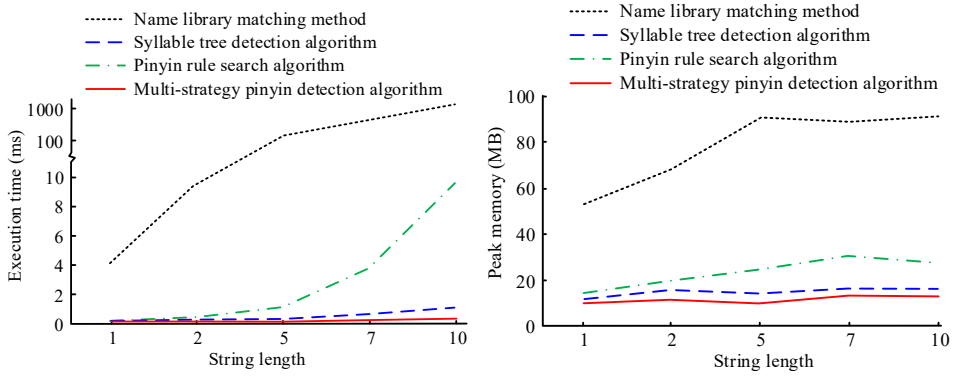
Table 2 Comparison of Pinyin detection results

Detection strategy	*Name library matching method*	*Syllable tree detection algorithm*	*Pinyin rule search algorithm*	*Multi-strategy Pinyin detection algorithm*
Execution time (s)	116.7580	0.0729	0.3401	0.1856
Memory usage (MB)	94.12	15.53	10.51	13.02
Word detection accuracy rate	0.9912	0.9462	0.8984	0.9521
Accuracy rate of Pinyin detection	0.9850	0.9980	0.9925	0.9995
Word detection recall rate	0.9821	0.9385	0.8847	0.9483
F1	0.9865	0.9423	0.8915	0.9502
Processing speed (words/s)	8.56	13,720.00	2,940.00	5,386.00

To further analyse the impact of varying input lengths on detection performance, the study generated Figure 7, illustrating the changes in execution time and memory usage for the four strategies across different string lengths. Figure 7(a) depicts the trend of execution time as string length increases. The name library matching method exhibited a sharp rise in processing time with growing length, reaching over 116 seconds when the length reached 10 characters, indicating its difficulty in adapting to long-text processing. The syllable-tree matching algorithm performed best, with execution times consistently below 0.1 seconds and the flattest growth curve, demonstrating its high efficiency. The rule matching algorithm had moderate processing times but a noticeable upward slope in

its growth curve, indicating increased computational complexity with longer strings. The multi-strategy method's time consumption closely resembled that of the syllable-tree algorithm, with only a slight increase for long texts, remaining under 0.2 seconds overall and reflecting its superior real-time performance. Figure 7(b) shows that the name library matching method had the highest memory usage, continuously increasing with length and peaking near 100 MB, indicating substantial resource consumption. The rule matching algorithm had the lowest memory usage, stabilising around 10.5 MB and demonstrating its lightweight advantage. The syllable-tree algorithm had moderate memory usage at 15.5 MB, maintaining stability. The multi-strategy method had slightly higher memory usage than the rule matching algorithm, stabilising at 13 MB, indicating a well-balanced trade-off between memory efficiency and processing capability achieved through strategy fusion.

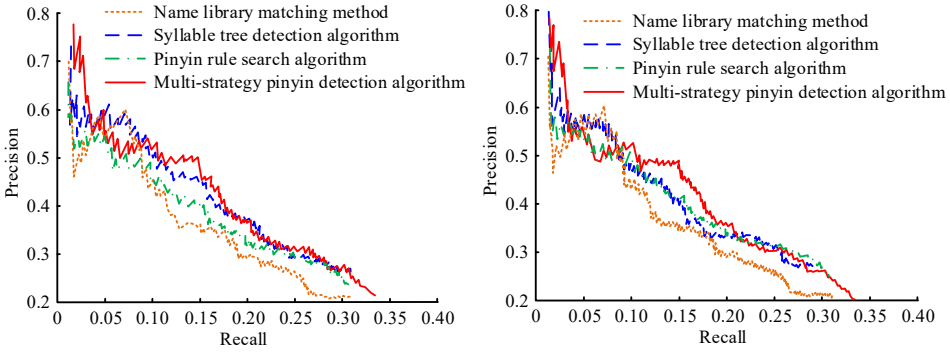
Figure 7 Comparison of efficiency under different input lengths, (a) curve of execution time varying with string length, (b) curve of memory usage varying with string length (see online version for colours)



To further evaluate the balance between precision and recall for each Pinyin detection strategy, the study generated Figure 8, which presents the comparative results of the precision-recall (P-R) curves. As can be seen from the figure, when the precision rate of the multi-strategy Pinyin detection algorithm reaches 99.95%, the recall rate can still maintain 94.83%, demonstrating the optimal performance balance. The curve of the name library matching method has a certain accuracy rate in the medium and low recall rate range, but when the recall rate increases to above 98%, the accuracy rate drops significantly. The curve position of the syllable tree matching algorithm is lower than that of the multi-strategy algorithm. When its recall rate reaches 93.85%, the precision rate is 94.62%. The P-R curve of the Pinyin rule matching algorithm is the lowest overall. When the precision rate is 89.84%, the recall rate is only 88.47%. From the overall performance of the curve, the P-R curve of the multi-strategy Pinyin detection algorithm was always at the top. The fundamental reason lied in that the hierarchical design of this algorithm achieved differentiated optimal processing for Pinyin samples of different natures and difficulties. For high-confidence standard Pinyin (easy sample), the syllable tree matching layer could quickly identify it with extremely high accuracy, laying the foundation for the curve to be in the high-accuracy range. For complex samples such as zero-initial syllables (difficult samples), the rule matching layer, as a fallback

mechanism, significantly enhanced the recall capability, making the curve decay more smoothly in the medium to high recall rate range. For the obviously non-Pinyin interference terms (negative samples), the length filtering and fast negation mechanism effectively controlled false positives, thereby stabilising the high precision performance.

Figure 8 The comparison results of P-R curves, (a) training set, (b) test set (see online version for colours)



(a) (b)

To further analyse error types, Table 3 presents the false positive and false negative statistics for each strategy on the test set. Table 3 reveals that the name library matching method exhibited a relatively high false negative rate (1.50%) due to incomplete lexicon coverage. The rule matching algorithm, while demonstrating strong rule generalisation capability, suffered from insufficient precision, resulting in high rates of both false positives and false negatives. In contrast, the multi-strategy Pinyin detection algorithm effectively controlled both types of errors through its two-stage verification mechanism, achieving false positive and false negative rates of 0.48% and 0.06%, respectively. This performance significantly outperformed other methods, particularly excelling in avoiding missed detections of Pinyin sequences.

Table 3 Analysis of error type distribution

Detection strategy	*False alarm number (English words are judged as Pinyin)*	*Number of underreported cases (Pinyin not recognised)*	*False alarm rate (%)*	*Underreporting rate (%)*
Name library matching method	9	75	0.18	1.5
Syllable tree detection algorithm	28	10	0.56	0.2
Pinyin rule search algorithm	52	38	1.04	0.76
Multi-strategy Pinyin detection algorithm	24	3	0.48	0.06

3.3 Performance of the GEC model

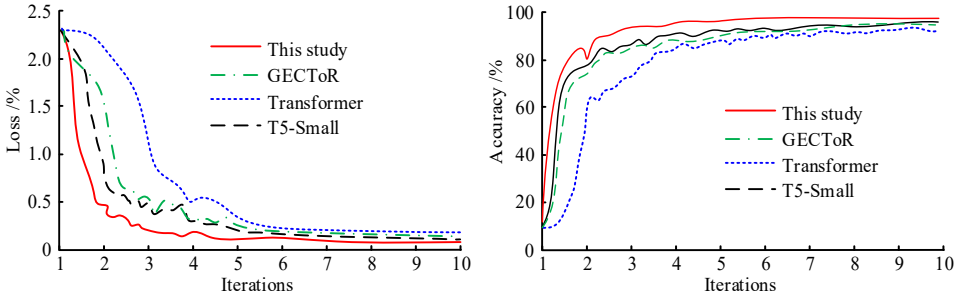
The study compared the proposed GEC model (denoted as this study) with the Sequence-to-sequence bidirectional long short-term memory model based on bidirectional LSTM (Seq2Seq-BiLSTM), the transformer model, the T5-small model, and the grammatical error correction: transformer-based sequence labelling (GECToR). As shown in Table 4, the grammar correction model proposed in this study achieved an F_{0.5} value of 40.58, an accuracy rate of 49.56%, and a recall rate of 34.72% on the CoNLL-2014 test set. Its comprehensive performance was significantly better than that of the listed baseline models. This was because the model integrated a high-precision multi-strategy Pinyin detection module at the front end, effectively protecting the Pinyin fragments in the text and avoiding misjudging them as English spelling errors for improper correction, thereby reducing over-correction from the source. Secondly, the model generated high-quality and diverse training corpora through a combination of rule-driven and model-driven approaches, enabling the model to learn more deeply the correction rules for specific errors such as article misuse and subject-verb inconsistency, thereby enhancing the error capture capability and making the recall rate better than the baseline. Furthermore, the model still maintained an inference speed of 175 sentences per second under the condition of only 68 M parameters, demonstrating a good balance between model lightweight and efficiency. Therefore, this model achieved a significant improvement in overall performance through the organic combination of error targeting enhancement, Pinyin protection mechanism and efficient architecture design.

Table 4 Overall performance comparison on the CoNLL-2014 test set

Model	*F_{0.5}*	*Precision (%)*	*Recall (%)*	*Model parameter quantity (M)*	*Reasoning speed (sentence/s)*
Seq2Seq-BiLSTM	25.73	31.45	21.82	90	85
Transformer	33.45	38.92	29.67	65	210
GECToR	39.12	47.83	33.41	110	180
T5-small	38.76	46.95	32.88	60	190
This study	40.58	49.56	34.72	68	175

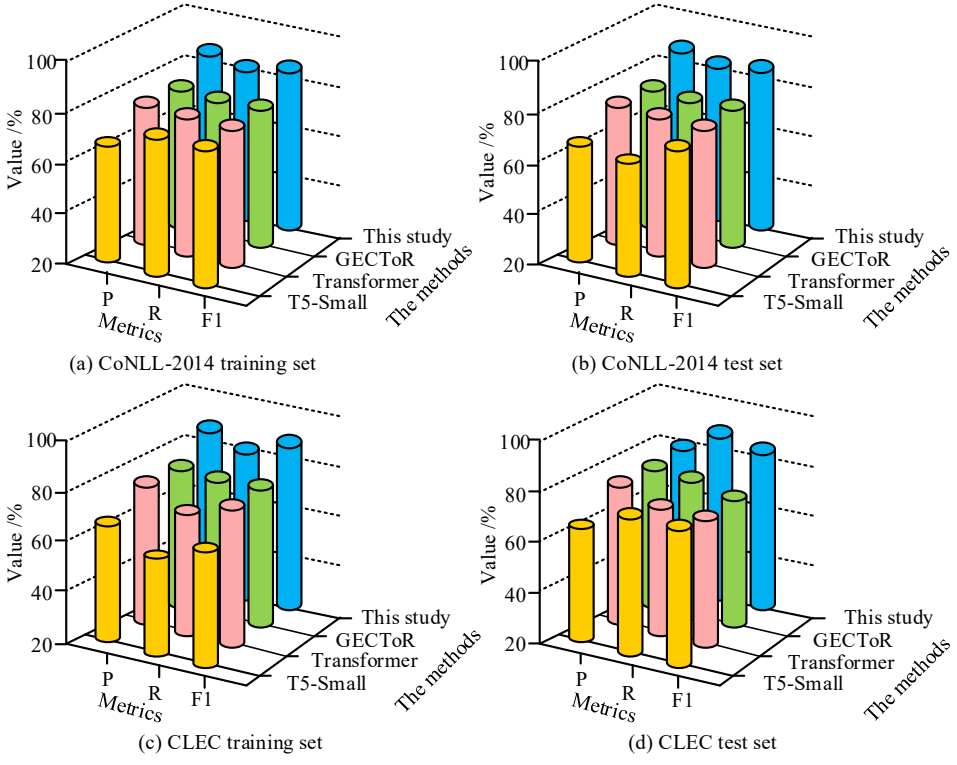
Figure 9 compares the performance of different models using dual indicators of training loss and test accuracy. Figure 9(a) illustrates the changes in training loss, revealing that the loss value of the research model continuously declined from an initial 2.45% to 0.52% by the tenth iteration, demonstrating the fastest convergence speed and a smooth curve. The GECToR model achieved a final loss of 1.15%, the transformer model 1.68%, and the T5-small model 1.92%. All three models exhibited higher curves than the research model and converged more slowly. Figure 9(b) reflects the changes in test accuracy, showing that the research model's accuracy steadily increased from 62.3% in the first round to 80.1% in the tenth round, consistently maintaining a significant lead. The GECToR model achieved a final accuracy of 69.8%, the transformer model 57.5%, and the T5-small model 55.2%. Together, these two sets of data indicated that the research model possessed significant advantages in both training efficiency and generalisation capability.

Figure 9 Comparison and analysis of training convergence and generalisation performance, (a) training time losses of different schemes, (b) test accuracy rates of different schemes (see online version for colours)



(a) (b)

Figure 10 Performance comparison of different models, (a) CoNLL-2014 training set, (b) CoNLL-2014 test set, (c) CLEC training set, (d) CLEC test set (see online version for colours)



(a) CoNLL-2014 training set (b) CoNLL-2014 test set

(c) CLEC training set (d) CLEC test set

To compare the performance of different GEC models, the study generated Figure 10. From Figure 10(a), on the CoNLL-2014 training set, the F1 score of the proposed model reached 90.36%, with precision and recall rates significantly higher than those of models such as GECToR and T5-Small. From Figure 10(b), the CoNLL-2014 test set, the bar chart representing the proposed model's metrics still ranked first, demonstrating

outstanding generalisation capability. From Figure 10(c), under the CLEC training set, the proposed model exhibited strong performance in terms of F1 score and recall, with bar heights surpassing those of other models. From Figure 10(d), on the CLEC test set, the proposed model had the highest F1 score bar, while also maintaining advantages in precision and recall metrics. Overall, across different datasets during both training and testing phases, the proposed model outperformed models such as GECToR and T5-small in terms of precision, recall, and F1 score, demonstrating more stable and superior performance.

To verify the model's ability to correct typical errors made by Chinese learners, the study extracted a test subset (CLEC-GEC-Test) comprising 2,000 sentences from the CLEC corpus. This subset focused on typical errors such as article misuse, subject-verb disagreement, verb tense errors, and Pinyin-character confusion. The results are shown in Table 5. The findings in Table 5 indicated that, benefiting from the stratified data augmentation strategy, which specifically generated error types prone to being made by Chinese learners, the proposed model significantly outperformed the baseline models in terms of detection and correction accuracy for specific errors such as articles, subject-verb agreement, and verb tenses. Particularly noteworthy was the 'Pinyin miscorrection' category, where the proposed model, leveraging its pre-positioned multi-strategy Pinyin detection module, successfully reduced the rate of Pinyin being misclassified as spelling errors and subsequently corrected from approximately 65% in the baseline model to just 1.8%. This reduction held decisive significance for enhancing the system's practicality in Chinese educational contexts.

Table 5 Error type segmentation performance on the CLEC-GEC-test subset

Error type	*Model*	*Detection accuracy rate (%)*	*Error correction accuracy rate (%)*	*Recall rate of this error type (%)*
Article misuse	Transformer	78.3	75.1	70.5
	GECToR	85.6	83.2	82.1
	This study	89.4	87.5	86.8
Subject-verb inconsistency	Transformer	82.5	80.3	76.4
	GECToR	88.9	86.7	85
	This study	92.1	90.2	89.5
Verb tense error	Transformer	75.6	72.8	68.9
	GECToR	83.4	80.5	79.2
	This study	87.8	85.9	84.3
Pinyin error correction	Transformer	30.5	-	-
	GECToR	35.2	-	-
	This study	98.2	-	-

To comprehensively evaluate the performance of the system in the actual deployment environment, detailed engineering performance tests were conducted on the system under various typical configurations. Condition A: INT8 quantisation model, batch size = 32, average sentence length = 15 words, hardware: NVIDIA A40. Condition B: FP16 precision model, batch size = 16, average sentence length = 25 words, hardware: NVIDIA V100. Condition C: INT8 quantisation model, batch size = 64, average sentence

length =10 words, hardware: NVIDIA A40. The test results are summarised in Table 6. The system demonstrated excellent engineering applicability and stability under different sentence lengths, batch sizes, and quantisation strategies. Under the condition of using INT8 quantisation and appropriate batch processing, the system could achieve a high throughput of over 300 sentences per second, while keeping the single-sentence inference latency within 33 milliseconds. In terms of resource usage, after quantification, the model's video memory usage was reduced to 0.61GB, and the disk model volume was only 87 MB, significantly reducing the deployment burden. The Pinyin detection module maintained a processing speed of over 5,000 words per second and an accuracy rate of over 99.9% under various conditions, demonstrating the high efficiency and stability of the pre-processing stage. In terms of concurrent support, the system could stably serve over 150 to 200 users, with a request success rate exceeding 99%, indicating its excellent scalability and service reliability. These key performance indicators jointly confirmed that the system could balance response speed, resource efficiency and concurrent capability in actual deployment, meeting the requirements of real-time performance and stability in educational application scenarios.

Table 6 Detailed evaluation of system engineering performance

Test dimension	*Indicator*	*Condition A*	*Condition B*	*Condition C*
Reasoning efficiency	Throughput (sentence/s)	218	165	302
	Average delay (ms/sentence)	46	61	33
	Peak delay (ms/sentence)	89	112	75
Resource occupation	GPU memory (GB)	0.61	2.1	0.61
	System memory peak (MB)	680	720	650
	Disk model size (MB)	87	315	87
Concurrent capability	Support concurrent user numbers	150+	100+	200+
	Request success rate (%)	99.3	98.7	99.5
Pinyin detection module	Detection speed (words per second)	5386	5200	5500
	Accuracy rate (%)	99.95	99.92	99.96

4 Discussion

The study proposed an intelligent GEC system that integrated multi-strategy Pinyin detection with stratified data augmentation to address the challenge of GEC caused by the mixing of Pinyin and English in texts produced by Chinese English learners. Experimental results demonstrated that the system exhibited significant advantages in terms of detection efficiency, correction accuracy, and practicality. In the design of the Pinyin detection module, the study employed a dual-strategy algorithm combining syllable-tree matching and rule matching, which substantially enhanced detection efficiency and accuracy. Experiments revealed that the multi-strategy Pinyin detection algorithm achieved an execution time of merely 0.1856 seconds and a memory footprint of 13.02 MB, far outperforming the traditional name library matching method, which

required 116.76 seconds and consumed 94.12 MB. This study has similarities with Kumar et al.'s deep first search method in terms of efficiency improvement. Both refine and optimise the graph structure search path through adjacency metrics, thereby enhancing the efficiency of pattern recognition in complex networks (Kumar, 2025). In contrast, the study optimised syllable matching paths using a BFS strategy, avoiding the recursive overhead associated with depth-first search and demonstrating the general advantages of graph structure algorithms in efficient pattern matching. Additionally, the improved depth-first search method proposed by Nikolaev et al. (2023) for constructing Gray codes underscored the importance of search strategies in structured data matching. The study further optimised the real-time performance and stability of the search process through a hierarchical decision-making mechanism.

In terms of Pinyin recognition accuracy, the proposed method achieved a Pinyin detection accuracy of 99.95%, with a false positive rate of merely 0.48% and a false negative rate as low as 0.06%. Regarding the architecture of the GEC model, the study was based on a Transformer encoder-decoder structure, incorporating AMs and autoregressive generation strategies to achieve end-to-end error detection and correction. Experimental results showed that the model attained an F_{0.5} score of 40.58, an accuracy of 49.56%, and a recall of 34.72% on the CoNLL-2014 test set, all outperforming the baseline models. Despite having a parameter count of only 68M, the model maintained a high inference speed, highlighting the importance of efficient model design.

Compared with the current general models based on large-scale pre-training, the model proposed in this study demonstrated more targeted advantages in the error correction task of texts for Chinese learners. On the general benchmark CoNLL-2014, the F_{0.5} value of this model reached 40.58, which was superior to 39.12 of GECToR and 38.76 of T5-Small, and the accuracy rate also led with 49.56%. This was mainly attributed to the targeted training data generated by the hierarchical data augmentation strategy, which enabled the model to learn the unique error patterns of Chinese learners more accurately. More importantly, in the CLEC subset test that reflected regional characteristics, the correction accuracy rates of this model for articles, subject-verb agreement, and verb tense errors reached 87.5%, 90.2%, and 85.9% respectively, which were significantly higher than those of the general model that was not specifically optimised. In addition, this model reduced the Pinyin error correction rate from approximately 65% of the baseline model to 1.8% through a pre-Pinyin protection module. This key improvement was not available in general pre-trained models. Although it was comparable to the lightweight pre-trained model in terms of efficiency metrics such as 68M parameters and an inference speed of 175 sentences per second, this model achieved a better balance between accuracy and practicality in error handling in specific domains through customised design at the algorithm level.

In summary, the study constructed an efficient, accurate, and practical intelligent English GEC system through the organic integration of multi-strategy Pinyin detection and stratified data augmentation. Optimisation strategies consistent with the latest research were evident across multiple stages, including Pinyin recognition, error generation, model training, and semantic validation, resulting in significant performance improvements in experimental validations. However, this study still has certain limitations. Firstly, the model faced challenges when dealing with long texts: on the one hand, the self-attention mechanism of the Transformer was prone to distraction in long sequences, which affected the capture of cross-paragraph grammatical dependencies; On

the other hand, although the existing length filtering strategy of the Pinyin detection module could eliminate unreasonable Pinyin, it was difficult to effectively identify the context association of multiple Pinyin combinations in long texts, which may lead to local misjudgement. Secondly, the current model only performed independent error correction for individual sentences and failed to model the historical information of multi-round dialogues. Therefore, it was unable to effectively identify and correct dependent grammatical errors caused by context omission or improper tense connection. In response to these issues, future work will focus on optimising the long text processing mechanism, introducing techniques such as sparse attention to improve long-range dependency modelling, and dynamically adjusting the Pinyin detection strategy. Meanwhile, a correction framework capable of integrating historical dialogue information is constructed. Historical dialogue information is incorporated into the model input, and context-dependent errors are captured through the cross-sentence attention mechanism to enhance the applicability of the system in real interaction scenarios.

5 Conclusions

The study addressed the core issue of collaborative correction of Pinyin misjudgements and grammatical errors in texts produced by Chinese English learners by constructing an intelligent English GEC system that integrates multi-strategy Pinyin detection, stratified data augmentation, and a Transformer architecture. The effectiveness and superiority of the proposed method were systematically validated through experiments. The results showed that, in the Pinyin detection phase, the multi-strategy algorithm significantly outperformed traditional methods. Its execution time was only 0.1856 seconds, far lower than the 116.76 seconds required by the traditional name library matching method, while its memory footprint was 13.02 MB, representing an 86% reduction compared to the 94.12 MB of the name library matching method. The Pinyin detection accuracy reached 99.95%, with a false positive rate of 0.48% and a false negative rate of 0.06%, completely resolving misjudgements and missed detections caused by insufficient lexicon coverage. The processing speed of 5,386 words per second fully met real-time processing requirements. Regarding GEC performance, the system achieved an F_{0.5} score of 40.58 on the CoNLL-2014 test set, representing a 1.46-point improvement over the baseline model GECToR. Its accuracy was 49.56%, recall 34.72%, with a parameter count of 68M and an inference speed of 175 sentences per second, achieving a balance between precision and efficiency. Testing on the CLEC-GEC-Test subset, which focused on typical errors made by Chinese learners, revealed that the system achieved correction accuracies of 87.5%, 90.2%, and 85.9% for article misuse, subject-verb disagreement, and verb tense errors, respectively. Moreover, it reduced the rate of Pinyin being misclassified as spelling errors from 65% to 1.8%. In summary, through hierarchical optimisation of Pinyin and grammatical error processing, the study effectively enhanced the adaptability of the error correction system to texts produced by Chinese learners and provided new insights for regionalised error correction research in the field of computational linguistics.

Disclaimer of interests

The author declares no conflicts of interest.

References

Alapati, P.R., Swathi, A., Madhuri, J.N., Burugari, V.K., Pagidipati, B. and El-Ebiary, Y.A.B. (2025) 'Improving English writing skills through NLP-driven error detection and correction systems', *Int. J. Adv. Comput. Sci. Appl*., September, Vol. 16, No. 2, pp.1098–1110.

Aravind, A., Durairaj, M., Chitkara, P., El-Ebiary, Y.A.B., Muniyandy, E., Ushasree, L. and Ammar, M.B. (2025) 'Adaptive AI-based personalized learning for accelerated vocabulary and syntax mastery in young English learners', *Int. J. Adv. Comput. Sci. Appl.*, November, Vol. 16, No. 4, pp.678–687.

Arundathi, A.J.V.S. and Satyanarayana, K.V.V. (2024) 'A secure and efficient framework for multi-user encrypted cloud databases supporting single and multiple keyword searches', *Int. J. Adv. Comput. Sci. Appl.*, June, Vol. 15, No. 9, pp.537–546.

Das, S., Dereniowski, D. and Uznański, P. (2024) 'Energy constrained depth first search', *Algorithmica*, October, Vol. 86, No. 12, pp.3759–3782, DOI: 10.1007/s00453-024-01275-8.

Hu, P. and Zhang, H. (2024) 'Verb form recognition and error detection in English articles using long short-term memory and grammar checks', *J. Adv. Comput. Intell. Inform.*, September, Vol. 28, No. 5, pp.1164–1168, DOI: 10.20965/jaciii.2024.p1164.

Jackson, S., Beekhuizen, B., Zhao, Z. and McEwen, R. (2025) 'GPT-4-Trinis: assessing GPT-4's communicative competence in the English-speaking majority world', *AI Soc.*, May, Vol. 40, No. 3, pp.1785–1801, DOI: 10.1007/s00146-024-01945-9.

Jiang, H. (2025) 'Developing an artificial intelligence model for English grammar correction: a computational linguistics approach', *J. Comput. Methods Sci. Eng.*, May Vol. 25, No. 4, pp.2992–3006, DOI: 10.1177/14727978251318799.

Jiang, P., Pan, W., Zhang, J., Wang, T. and Huang, J. (2023) 'A robust conformer-based speech recognition model for Mandarin air traffic control', *Comput. Mater. Contin*., Vol. 77, No. 10, pp.911–940, DOI: 10.32604/cmc.2023.041772.

Kavros, A. and Tzitzikas, Y. (2023) 'SoundexGR: an algorithm for phonetic matching for the Greek language', *Nat. Lang. Eng.*, February, Vol. 29, No. 5, pp.1305–1340, DOI: 10.1017/S1351324922000018.

Khabutdinov, I.A., Chashchin, A.V., Grabovoy, A.V., Kildyakov, A.S. and Chekhovich, U.V. (2024) 'RuGECToR: rule-based neural network model for Russian language grammatical error correction', *Program. Comput. Softw.*, July, Vol. 50, No. 4, pp.315–321, DOI: 10.1134/S0361768824700129.

Kumar, N., Kumar, P., Tripathy, S., Samal, N., Gountia, D. and Singh, T. (2024) 'Context-aware adversarial graph-based learning for multilingual grammatical error correction', *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, November, Vol. 23, No. 12, pp.1–15, DOI: 10.1145/3696106.

Kumar, P. (2025) 'A depth-first search approach to detect the community structure of weighted networks using the neighbourhood proximity measure', *Int. J. Data Sci. Anal.*, September, Vol. 20, No. 3, pp.2833–2850, DOI: 10.1007/s41060-024-00631-9.

Lee, S., Yun, G., Nguyen, X.T. and Lee, H.J. (2025) 'FACET: on-the-fly activation compression for efficient Transformer training', *IEEE Trans. Circuits Syst. I, Reg. Papers*, October, Vol. 72, No. 8, pp.4091–4102, DOI: 10.1109/TCSI.2024.3506999.

Li, J., Zhang, D., Xie, Y., Wulamu, A. and Zhang, Y. (2024) 'GP-FMLNet: a feature matrix learning network enhanced by glyph and phonetic information for Chinese sentiment analysis', *CAAI Trans. Intell. Technol.*, March, Vol. 9, No. 4, pp.960–972, DOI: 10.1049/cit2.12300.

Li, L., Long, Y., Xu, D. and Li, Y. (2023) 'Boosting character-based Mandarin ASR via Chinese Pinyin representation', *Int. J. Speech Technol*., November, Vol. 26, No. 4, pp.895–902, DOI: 10.1007/s10772-023-10050-z.

Li, Y., Huang, H., Wang, B. and Gao, Y. (2025) 'DRMSpell: dynamically reweighting multimodality for Chinese spelling correction', *Front. Inf. Technol. Electron. Eng.*, April, Vol. 26, No. 3, pp.354–366, DOI: 10.1631/FITEE.2300816.

Liu, J., Xie, Z., Zhang, C. and Shi, G. (2021) 'A novel method for Mandarin speech synthesis by inserting prosodic structure prediction into Tacotron2', *Int. J. Mach. Learn. Cybern.*, July, Vol. 12, No. 10, pp.2809–2823, DOI: 10.1007/s13042-021-01365-x.

Liu, Q., Peng, Y., Tang, Z., Jiang, H., Wu, J., Peng, T. and Wang, G. (2023) 'VeffChain: enabling freshness authentication of rich queries over blockchain databases', *IEEE Trans. Knowl. Data Eng.*, May, Vol. 36, No. 5, pp.2285–2300, DOI: 10.1109/TKDE.2023.3316127.

Mangotra, H., Dabas, V., Khetharpal, B., Verma, A., Singhal, S. and Mohapatra, A.K. (2024) 'University auto reply FAQ chatbot using NLP and neural networks', *Artif. Intell. Appl.*, June, Vol. 2, No. 2, pp.126–134, DOI: 10.47852/bonviewAIA3202631.

Nikolaev, S., Romanov, O. and Nyshchuk, A. (2023) 'Method of modified depth-first search in a graph for constructing all possible Gray codes of a specified length', *Cybern. Syst. Anal.*, May, Vol. 59, No. 3, pp.359–364, DOI: 10.1007/s10559-023-00570-6.

Qing, Y. (2024) 'Design and application of automatic English translation grammar error detection system based on BERT machine vision', *Scalable Comput. Pract. Exp.*, April, Vol. 25, No. 3, pp.2088–2102, DOI: 10.12694/scpe.v25i3.2770.

Shanthamallappa, M. and Ravi, D.J. (2023) 'Robust perceptual wavelet packet features for the recognition of spontaneous Kannada sentences', *Wireless Pers. Commun.*, December, Vol. 133, No. 2, pp.1011–1030, DOI: 10.1007/s11277-023-10802-9.

Shao, H. and Liu, Z. (2024) 'Mobile smart app and its application in improving the efficiency of English homework correction', *Scalable Comput. Pract. Exp.*, April, Vol. 25, No. 3, pp.1631–1646, DOI: 10.12694/scpe.v25i3.2791.

Song, J., Liu, Y. and Qu, Y. (2023) 'N-gram language model for Chinese function-word-centered patterns', *J. Comput. Inf. Technol*., July, Vol. 31, No. 1, pp.39–55, DOI: 10.20532/cit.2023.1005719.

Sun, G. and Zhang, Z. (2024) 'Radical-attended and pinyin-attended malicious long-tail keywords detection', *Neural Comput. Appl.*, May, Vol. 36, No. 24, pp.14757–14773, DOI: 10.1007/s00521-024-09871-z.

Sun, H., Kong, D., Jiang, S., Yue, Y. and Qin, X. (2024) 'TrieKV: a high-performance key-value store design with memory as its first-class citizen', *IEEE Trans. Parallel Distrib. Syst.*, December, Vol. 35, No. 12, pp.2479–2496, DOI: 10.1109/TPDS.2024.3473013.

Wang, H., Wang, B., Duan, J. and Zhang, J. (2021) 'Chinese spelling error detection using a fusion lattice LSTM', *ACM Trans. Asian Low-Resour. Lang. Inf. Process*., May, Vol. 20, No. 2, pp.1–11, DOI: 10.1145/3426882.

Wang, J., Yue, K. and Duan, L. (2023) 'Models and techniques for domain relation extraction: a survey', *J. Data Sci. Intell. Syst.*, May, Vol. 1, No. 2, pp.65–82, DOI: 10.47852/bonview JDSIS3202973.

Zhang, Y., Liu, Y., Zhu, J. and Wu, X. (2021) 'FSPRM: a feature subsequence based probability representation model for Chinese word embedding', *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, July, Vol. 29, No. 1, pp.1702–1716, DOI: 10.1109/TASLP.2021.3073868.

Zhang, Y., Liu, Y., Zhu, J., Chen, Z., Zhai, S. and Wu, X. (2025) 'Inner-character and inner-word features based representation learning for Chainese word embedding', *ACM Trans. Asian Low-Resour. Lang. Inf. Process*., September, Vol. 24, No. 9, pp.1–33, DOI: 10.1145/3748316.

Zhao, K. (2024) 'English grammar correction based on attention mechanism machine translation', *Int. J. Comput. Intell. Stud.*, January, Vol. 13, Nos. 1–2, pp.78–94, DOI: 10.1,504/IJCISTUDIES.2024.144049.

Zhao, W., Lian, Y., Chai, J. and Tu, Z. (2023) 'Multi-speaker Chinese news broadcasting system based on improved Tacotron2', *Multimed. Tools Appl.*, May, Vol. 82, No. 30, pp.46905–46937, DOI: 10.1007/s11042-023-15279-z.

Zhong, Q., Ding, L., Liu, J., Du, B. and Tao, D. (2023) 'E2S2: encoding-enhanced sequence-to-sequence pretraining for language understanding and generation', *IEEE Trans. Knowl. Data Eng*., December, Vol. 36, No. 12, pp.8037–8050, DOI: 10.1109/TKDE.2023.3341917.