ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230627

MIATS:A Chinese Spelling Error Correction Algorithm Based on Multimodal Information Alignment of Three-Towers Structure

Guochao Zhao[a], Yan Guo[b;*], Jia Tang[a], Zongwei Zhu[b] and Guixing Wu[b]

[a]University of Science and Technology of China, Hefei 230026, China
[b] Suzhou Institute for Advanced Research, University of Science and Technology of China, Suzhou 215123, China
Guochao Zhao iyao@mail.ustc.edu.cn, Yan Guo guoyan@ustc.edu.cn, Jia Tang sa21225401@mail.ustc.edu.cn, Zongwei Zhu zzw1988@ustc.edu.cn, Guixing Wu gxwu@ustc.edu.cn

Abstract. Chinese spelling correction (CSC) is a crucial task in natural language processing, aiming to detect and correct spelling errors in Chinese text. The improved performance of Chinese spelling errors correction algorithms can enhance the efficiency and accuracy of upstream and downstream Chinese natural language processing tasks, such as OCR, ASR, and translation.However, current methods based on neural networks are mostly limited to either using only contextual information to correct misspelled words or failing to fully utilize glyph and pinyin information. Therefore, we propose a multimodal approach to address the above issues. Specifically, a three-tower multimodal structure is used to extract glyph, pinyin, and semantic information, and a decoder composing of an error probability prediction network and a transformer network is employed to achieve cross-modal information interaction. Besides, an additional training task is used to achieve cross-modal information alignment. Experiments demonstrate that proposed network outperform most existing motheds.

1 Introduction

In recent years, with the rapid development of information technology, various natural language applications, such as machine translation [26], optical character recognition (OCR) [1], and automatic speech recognition (ASR) [9] are becoming widespread. However, due to many reasons, like low quality of training corpus and poor performance of the neural networks, the resulted outcome of such applications might generate spelling errors, which will further affect downstream applications. Therefore, it is necessary to do text correction to effectively improve the performance and efficiency of various Chinese natural language processing tasks. For example, the query string directly affects the results of search search engines [11], but due to reasons like typo errors as well as lack of necessary background knowledge, the query strings input by users are prone to be with errors. With corrected query string, better search results and user experience could be achieved. On the other hand, it is widely acknowledged that high quality training corpus plays a vital role in the training of neural networks. The presence of spelling errors would bring damage to the language understanding ability of the generated model. Thus, training corpus whose spelling errors have been corrected will lead to better training result.

Undoubtedly, it is obvious that Chinese spelling errors correction is quite important in Chinese natural language processing tasks.Research on Chinese spelling errors correction has a long history, dated back to as early as 1990s, with the target to detect and correct errors in Chinese text [2, 20]. The former research in Chinese spelling errors correction field can be roughly categorized into two classes: traditional machine learning methods [24, 19, 17, 25] and deep learning methods [23, 14, 15, 7, 5]. Traditional machine learning methods usually generate candidate words based on a dictionary and confusion set, calculate perplexity [8] based on language models to rank and select the optimal candidate words. Deep learning methods typically learn better information representation through large corpus training and then correct spelling errors with the learned language understanding ability. Generally speaking, deep learning methods take into consideration of contextual information, and their effectiveness is superior to traditional machine learning methods.

Chinese characters are special in the sense that characters sometimes contain semantic information. Phonetically similar characters are also semantically similar. And it is observed that most errors are caused by visual and phonological similarities [10]. So such features are important in Chinese processing. And in English, edit distance is available as a measure to indicate spelling similarity between words. Phonetical information is also be used in English spelling correction.

Table 1. Two spelling errors examples:errors caused respectively by similar character glyph and by similar character pinyin.

Type	Percentage	Example	Correction
Glyph Erros	48%	我爱白然语言处理。	白→自
Pinyin Errors	83%	我的眼镜有点红肿。	镜→睛

For Chinese text errors correction tasks, there are mainly two spelling error sources: errors caused by character glyph similarity and errors caused by character pinyin similarity. In Liu et al.'s study [10], 48% of errors were caused by errors in similar character glyphs, such as the similarity between " 白 (which means white) " and " 自 (which means natural together with 然) " in " 我爱白然语言处理 " (I love white language processing). 83% of Chinese spelling errors

* Corresponding Author.

were caused by errors in similar character pronunciations, such as the similarity between " 镜 (which means glasses) " and " 睛 (which means eyes) " in " 我的眼镜有点红肿 " (My glasses are a little swollen and red).

From the above observation, it is obvious that character glyph and pinyin features will provide effective information for correction. Although recent years have seen the emergence of many Chinese spelling errors correction algorithms, such methods mainly focus on fully utilizing contextual and semantic information, but fail to fully utilize the glyph and pinyin information of characters, resulting in insufficient feature processing.

In this paper, we investigate multimodal techniques in Chinese spelling errors correction and introduce a multimodal model for the correction tasks. We propose a Chinese spelling errors correction algorithm **MIATS**, that is, **M**ultimodal **I**nformation **A**lignment of **T**hree-towers **S**tructure. Our model uses a three-tower structure, in which BERT acts as an encoder to model glyph, pinyin, and semantic information. Besides, through training on a task to learn multimodal information alignment, better multimodal information representation is obtained. As for decoder, we use transformer to achieve cross-modal information interaction. Besides, an error detection network is trained to calculate the probability of errors occurring at each position, based on which multimodal information and contextual information is further fused. Finally, a simple linear layer is used as a classifier to predict the final result and select proper word from the vocabulary. The proposed method is tested on the publicly available Chinese spelling errors correction datasets SIGHAN2013, SIGHAN2014, and SIGHAN2015, achieving F1-scores of 0.78, 0.67, and 0.77, respectively, surpassing most of the existing Chinese spelling error correction algorithms. The contributions of this paper are as follows:

1. We propose a Chinese spelling errors correction algorithm based on the alignment of multimodal information using a three-tower structure.
2. We design an error probability prediction network with transformer to achieve cross-modal information interaction.
3. We design specific training tasks, including error correction task, error detection task, and cross-modal information alignment task to better train the network.

2 Related work

2.1 Chinese spelling errors correction based on traditional machine learning

Traditional machine learning methods for Chinese spelling errors correction mainly utilize basic word frequency statistics, preconstructed confusion sets, and language models. Candidate sets for correction are formed from confusion sets, and the optimal candidate is selected with the help of language models. For example, Yeh et al. proposed a method using n-gram ranked inverted index list and a confusion set of homophonic characters [19]. Zhang et al. proposed a correction framework HANSpeller++ [24], which uses a hidden Markov Model to generate candidate sets and a filter to rank them. Xie et al. proposed a method employing a joint n-gram grammar and Chinese segmentation language model [17] to enumerates all candidate words. Zhao et al. proposed a hybrid model for CSC, comprising three models dealing with different correction tasks [25].

Traditional machine learning methods are advantageous in processing speed and computational resource consumption, as the language models are generally simple. However, since they do not have the ability to extract contextual semantic information, their performance is limited. What is more, the performance is directly affected by confusion sets, in the case of new words or out-of-vocabulary words, the confusion sets and dictionaries need to be constantly updated, resulting in increasing maintenance cost.

2.2 Chinese spelling errors correction based on deep learning

Compared with traditional methods, large deep learning-based methods are more powerful in contextual semantic information extraction. Prior to BERT, most commonly used neural networks are recurrent neural networks (RNNs) for Chinese spelling correction. Wang et al. proposed a sequence labeling model based on LSTM [6] for error detection [14], while Yang et al. proposed a method [18] based on LSTM with CRF for error position detection. Wang et al. utilizes both LSTM and confusion sets [15].

After the introduction of BERT [5] by Devlin et al. in 2018, pre-trained language models are widely used in various natural language processing tasks, including error correction. Hong et al. proposed FASPell [7] using a BERT-based new paradigm which consists of a denoising autoencoder (DAE) and a decoder. FASPell eliminates the need of confusion sets, but instead utilizes a Confidence-Similarity Decoder to rank candidates based on confidence and similarity (in terms of character glyph and pinyin). Ming proposes MacBERT4CSC [12] based on MacBERT [4], which integrates the error detection and correction tasks, and extends the correction task to include the detection task. Zhang et al. proposed Soft-Masked BERT [23], which divides the entire correction task into two parts, corresponding to two networks.

For the Chinese spelling errors correction task, the character glyph and pinyin can provide valuable information for the entire correction system. However, existing BERT-based correction algorithms have not fully utilized such information. Zhang et al. proposes ERNIE4CSC [22], combining character pronunciation with ERNIE, which combines the error detection probability, character pronunciation as well as ERNIE output, and then uses a transformer for the final prediction. Cheng et al. proposes SpellGCN [3], which uses BERT to extract semantic knowledge and GCN to extract character glyph and pinyin features. The GCN is able to capture the pronunciation/shape similarity and explore the prior dependencies between characters.

As for utilizing glyph and pinyin information, existing algorithms still have some problems. MacBERT4CSC [12] and Soft-Masked BERT [23] use the powerful representation ability of BERT for error correction, but do not consider the impact of Chinese character glyph and pinyin information on the task. ERNIE4CSC [22] combines pinyin knowledge for error correction, but ignores character glyph information. Based on this observation, this paper conducts in-depth research on multimodal Chinese spelling errors correction and proposes MIATS, a Chinese spelling errors correction algorithm based on three-tower multimodal information alignment. MIATS achieves better performance than most existing CSC algorithms on the SIGHAN dataset.

3 Method

Firstly, MIATS uses a three-tower multi-modal structure as an encoder to extract information from multiple data modalities. Secondly, it employs a decoder to achieve cross-modal information interaction and to better utilize context knowledge. The decoder consists of a transformer network to extract context information and a linear

network to predict the probability of a character is incorrect, thus to explicitly distinguish correct and incorrect characters processing. Therefore, the decoder is able to more effectively utilize token feature information with context information. Specific training tasks are designed to train the model, and correction loss, detection loss, and contrastive loss are used as loss functions for correction task, detection task, and cross-modal information alignment task, respectively. As a result, MIATS is able to fully extract and utilize multi-modal information and context information, explicitly distinguish processing of correct and incorrect characters, thus achieve better correction result on the SIGHAN datasets.

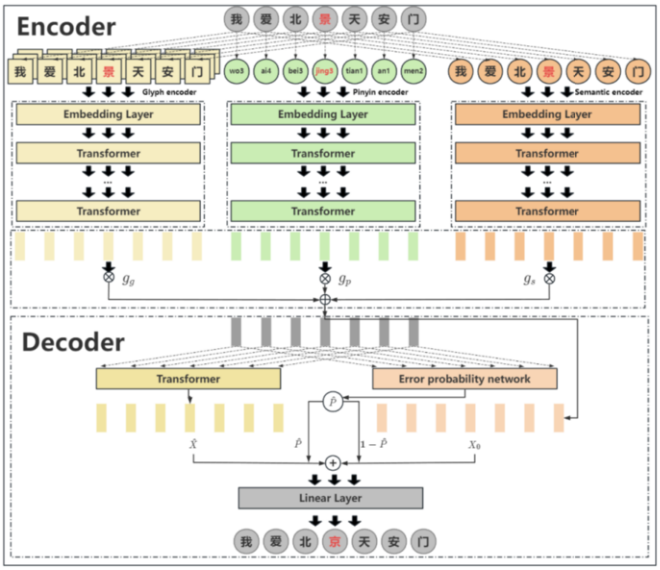

Figure 1. The network architecture of MIATS:It includes an encoder with a three-tower multi-modal structure and an encoder composed of a transformer network and an error probability network.

Figure 1 illustrates the network architecture of MIATS. The three-tower multi-modal structure is used to extract glyph, pinyin, and semantic information, and the adaptive algorithm adjusts the weights of different modalities for multi-modal information fusion. The transformer network of the decoder aims to interact cross-modal information among different tokens. The error probability network, consisting of a linear layer and a sigmoid function, estimates the probability of error occurrence at each position. Based on the estimation, the vector representation of the input is computed with both context information as well as the character itself. Finally, a linear layer maps the result to the vocabulary space and calculates the probability distribution of each word to obtain the final correction result.

3.1 Encoder

3.1.1 Glyph encoder

The structure of the glyph encoder is illustrated in Figure 2. In this section, we use the open-source package Pillow[1] to generate Chinese character images for each character in the input text, resulting in a

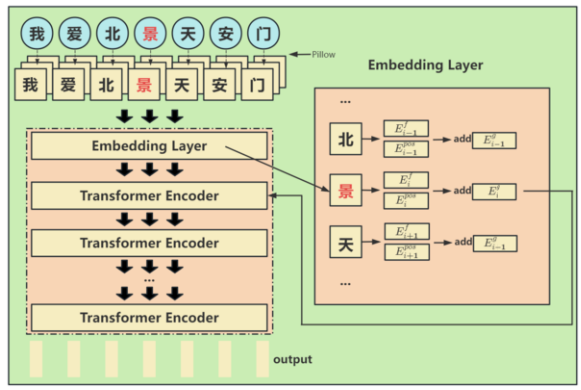

Figure 2. Glyph encoder:It uses transformer as the backbone network, whose embedding includes both glyph information of different fonts and positional information.

sequence of Chinese character images. For each character, we generate images in three fonts, namely Songti, Kaiti, and Lishu, each of size $16 * 16$. Each image is flattened into a vector of dimension 256, and the three vectors are concatenated to obtain a 768-dimensional vector as the embedding of the glyph image, denoted as E^f in Figure 2. To avoid losing positional information among tokens, we also use positional embeddings E^p, and the two embeddings are added to obtain the glyph embedding E^g, as shown in Equation (1):

$$E^g = E^f + E^p \tag{1}$$

The embeddings are then fed into L layers of a transformer network. As shown in Equation (2), $Transformer_l$ represents the l-th layer of transformer, and $H_l^g \in R^{n \times d}$, where n is the sequence length, d is the hidden size, H_l^g and H_{l-1}^g represents the output of the l-th and (l-1)-th layer of transformer. The total number of layers in the transformer is L.

$$H_l^g = Transformer_l(H_{l-1}^g), l \in [1, L] \tag{2}$$

When $l = L$, the output of the glyph encoder, $H^g = H_L^g = \{h_0^g, ..., h_{n-1}^g\}$, is the vector representation of the glyph information of the input text sequence. Here, h_i^g represents the glyph feature of the i-th token in the input text sequence.

3.1.2 Pinyin encoder

The structure of the pinyin encoder is illustrated in Figure 3. The embedding of the pinyin encoder consists of four parts: initial syllable embedding, final syllable embedding, tonetic embedding, and positional embedding, as shown in the Figure 3. Given a Chinese character such as " 景 ", it is converted into a pinyin sequence "jing3" using the open-source toolkit PyPinyin[2]. The initial consonant "j", final compound vowel "ing", tonetic symbol "3", and the position index of " 景 " in the text sequence (here is "4") are extracted, and then their corresponding embeddings, denoted as E^i,E^f,E^t, and E^{pos}, respectively, are summed up to obtain the embedding of the pinyin encoder, denoted as E^p, as shown in Equation (3).

$$E^p = \frac{E^i + E^f + E^t + E^{pos}}{4} \tag{3}$$

Thus obtained pinyin embedding is then fed into the L layers of the transformer network, as shown in Equation (4), where

[1] https://github.com/python-pillow/Pillow

[2] https://github.com/mozillazg/python-pinyin

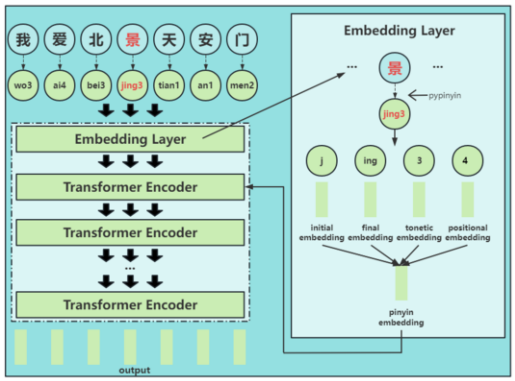

Figure 3. Pinyin encoder: Transformer acts as the backbone network, and the embedding consists of four parts, that is, tone information, initial syllable information, last syllable information and positional information.

$Transformer_l$ represents the l-th layer of the transformer network, $H_l^p \in R^{n\times d}$ denotes the output of the l-th layer of the transformer, H_{l-1}^p denotes the output of the previous layer, and L is the total number of layers in the transformer network. The pinyin information of the input is the output of the Lth layers of the transformer network of the pinyin encoder.

$$H_l^p = Transformer_l(H_{l-1}^p), l \in [1, L] \tag{4}$$

When $l = L$, the output of the transformer network, $H^p = H_L^p = \{h_0^p, ..., h_{n-1}^p\}$, is the vector representation of the pinyin information of the input text sequence, where h_i^p represents the phonetic feature of the i-th token in the input text sequence.

3.1.3 Semantic encoder

The semantic encoder employs MacBERT, which distinguishes itself with powerful information extraction capability. Its embedding is consistent with that of BERT, including token embedding and positional embedding. The calculation process of the transformer layer is shown in Equation (5), and the output of the last layer is $H^s = H_L^s = \{h_0^s, ..., h_{n-1}^s\}$,as the vector representation of the semantic information of the input text sequence. Here, h_i^s represents the semantic feature of the i-th token in the input text sequence.

$$H_l^s = Transformer_l(H_{l-1}^s), l \in [1, L] \tag{5}$$

3.1.4 Fusion

The vectors output by the three encoders, namely, glyph, pinyin, and semantic encoders, are fused as multimodal information using gate function, where the weights for each modality are obtained. Consequently, the importance of each encoder is adaptively adjusted, and the multimodal knowledge is fully fused. Equations (6), (7), and (8) describe the weight calculation process based on gate function for the three modalities.

$$g^g = \sigma(W^g(H^g + H^p + H^s) + b^g) \tag{6}$$
$$g^p = \sigma(W^p(H^g + H^p + H^s) + b^p) \tag{7}$$
$$g^s = \sigma(W^s(H^g + H^p + H^s) + b^s) \tag{8}$$

Here, $W^g, W^p, W^s \in R^{d\times 1}$, which are used to reduce the sum of H^g,H^p and H^s to a vector of size 1, respectively. The sigmoid function σ is applied to calculate the weights g^g,g^p,and g^s for each modality. The fused multimodal information result H is obtained using the weighted sum of the information from each modality, as shown in Equation (9).

$$H = g^g * H^g + g^p * H^p + g^s * H^s \tag{9}$$

3.2 Decoder

The decoder module of the MIATS aims to achieve cross-modal information interaction among different tokens. The network structure of decoder is illustrated in Figure 4.

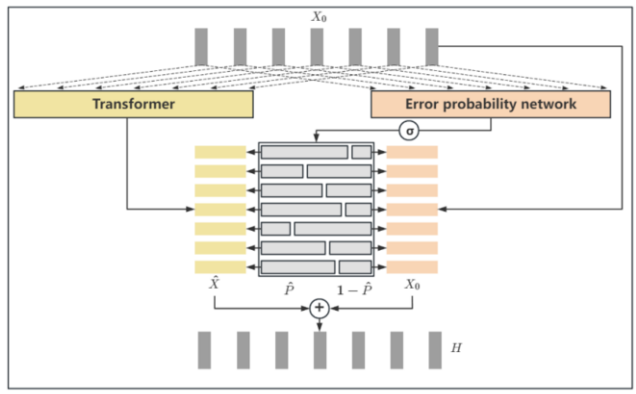

Figure 4. The decoder network consists of a transformer, an error probability network, and a fusion layer. The output of the fusion layer can be used to predict the correction result.

3.2.1 Transformer layers

The output of the encoder, denoted as X_0, represents the fused multimodal information, where each token's corresponding vector contains its glyph, pinyin, and semantics information. However, different modalities of information among tokens are separate and do not interact with each other. To address this issue, we introduce a transformer network after X_0, where the self-attention mechanism enables cross-modal interactions between different tokens. Specifically, as shown in Equation (10), X_0 is fed through L layers of transformer network, where $Transformer_l$ represents the l-th layer of the transformer, $X_l \in R^{n\times d}$ is the output of the l-th layer, and H_{l-1} is the output of the (l-1)-th layer. Here, L is the total number of layers in the transformer. The input text sequence's i-th token feature is represented as x_i.

$$X_l^g = Transformer_l(X_{l-1}^g), l \in [1, L] \tag{10}$$

When $l = L$, the final output of the transformer network is denoted as $\hat{X} = X_L = \{x_0, ..., x_{n-1}\}$. Compared with X_0, each vector in $\hat{X}$ contains more contextual information.

3.2.2 Error probability prediction network

In addition, an error probability prediction network is introduced in the decoder of MIATS, which consists of a linear layer and a sigmoid function, with input X_0, as shown in Figure 1. The purpose of this network is to detect the probability of error at each position in the input sequence. The linear layer is represented as Equation (11),

where $W_P \in R^{d\times 1}$, and the final error probability is obtained with the sigmoid function.

$$\hat{P} = \sigma(W_P X_0 + b_P) \tag{11}$$

With this probability, the correct and incorrect characters can be explicitly distinguished and be processed differenctly.

3.2.3 Fusion

The transformer layer and error probability prediction network separately generate $\hat{X}$ and $\hat{P}$ for each token. X_0 is the input of transformer,representing the multimodal feature information of each token itself, including its glyph, pinyin, and semantic information;$\hat{x}$ incorporates the contextual information; and $\hat{P}$ represents the probability that each token is a misspelling. The intuition is that, if the character itself is correct, then its knowledge is important for the understanding of the input; however, if the character is incorrect, then its feature information is less useful or even misleading. Thus, different processing should be adopted for correct and incorrect characters. If a token is regarded as incorrect, for the correction task,$\hat{X}$, as the context information, should be incorporated other than the feature information X_0 of this character itself. If the token is not regarded as misspelling, then the network would directly copy the original information of this token X_0, which is more important than contextual knowledge $\hat{X}$. Therefore, H, as the fusing result of X_0 and $\hat{X}$ is introduced, as shown in Equation (12), it is based on the computed error probability to dynamically adjust the weights of X_0 and $\hat{X}$.

$$H = \hat{P} * \hat{X} + (1 - \hat{P}) * X_0 \tag{12}$$

When $\hat{P}$ is relatively large, indicating that the token is highly likely to be a misspelling,$\hat{X}$ has a greater weight, thus to reduce the influence of the incorrect word itself while utilizing the correct context. On the other hand, it also takes into consideration of the original X_0, since the pinyin and glyph information might be helpful. When $\hat{P}$ is relatively small, X_0 has a greater weight, indicating that the word is likely to be the correct word and its original information is useful. Through this fusion step, it can be ensured that correct and useful information could be obtained.

3.2.4 Prediction

As shown in Equation (13), (14)and (15),after the fusion of X_0 and $\hat{X}$, H is obtained, and H is mapped to the vocabulary space denoted as $\hat{H}$ by a linear layer. The softmax function is then applied to obtain the probability distribution of each word , denoted as $\hat{Y}$, and the word with the highest probability is selected as the final prediction result by the argmax function, denoted as Y.

$$\hat{H} = WH + b \tag{13}$$

$$\hat{Y} = Softmax(\hat{H}) \tag{14}$$

$$Y = Argmax(\hat{Y}) \tag{15}$$

3.3 Training task

The training task of the MIATS consists of three parts: error correction task, error detection task, and cross-modal information alignment task. These tasks correspond to the correction loss, detection loss, and contrastive loss, respectively.The reasons for our choice are as follows: The correction loss is used to represent the difference between the correction result and the ground truth; the detection loss is used to measure the performance of the error probability network; and the contrastive loss is used to constrain the interaction of multimodal information, achieving cross-modal information alignment.The weight ratios of the three loss functions are evaluated through extensive experiments.

3.3.1 Error Correction Task

The error correction task uses cross-entropy loss, which is calculated by the last linear layer of the network to obtain the probability distribution of each word in the vocabulary $\hat{Y} = \{\hat{y}_0, ..., \hat{y}_{l-1}\}$, and the difference between the predicted labels $Y = \{y_0, ..., y_{l-1}\}$ and the true labels is represented by the loss ℓ_1 in Equations (16) and (17):

$$\ell_1 = \frac{1}{B}\sum_{i=1}^{B} \tilde{\ell}_1(Y^i, \hat{Y}^i) \tag{16}$$

$$\tilde{\ell}_1(Y^i, \hat{Y}^i) = -\sum_{j=1}^{l}\sum_{k=1}^{n} y^i_{j,k} \log \hat{y}^i_{j,k} \tag{17}$$

Where B represents the batch size, $Y^i = \{y^i_1, ..., y^i_l\}$ stands for the ground truth label of the i-th text sequence in the batch,$\hat{Y}^i = \{\hat{y}^i_1, ..., \hat{y}^i_l\}$ represents the predicted probability of the i-th text sequence in the batch, and n represents the size of the vocabulary. $\tilde{\ell}_1(Y^i, \hat{Y}^i)$ represents the correction loss of the i-th text sequence in the batch, and ℓ_1 is the average correction loss of the batch.

3.3.2 Error Detection Task

The error detection task uses binary cross-entropy loss to measure the performance of the error probability network. For each position in the text sequence, label p stands for the probability that the character in the position is incorrect, where value of 0 indicates that the position is not a misspelling while 1 denotes that the position is a misspelling. The error detection network outputs the probability of the character in the position being incorrect.

$$\ell_2 = \frac{1}{B}\sum_{i=1}^{B} \tilde{\ell}_2(P^i, \hat{P}^i) \tag{18}$$

$$\tilde{\ell}_2(P^i, \hat{P}^i) = -\sum_{j=1}^{l}\left[p^i_j \log \hat{p}^i_j + (1 - p^i_j)\log(1 - \hat{p}^i_j)\right] \tag{19}$$

As shown in Equation (18)and (19), the detection loss is calculated by the binary cross-entropy loss function to evaluate the difference between the ground truth labels $P^i = \{p^i_1, ..., p^i_l\}$ of each token in the i-th text sequence in the batch and the predicted probability $\hat{P}^i = \{\hat{p}^i_1, ..., \hat{p}^i_l\}$ of the error probability network in the batch. $\tilde{\ell}_1(Y^i, \hat{Y}^i)$ represents the detection loss of the i-th text sequence in the batch, and ℓ_1 denotes the average detection loss of the batch.

3.3.3 Cross-Modal Information Alignment Task

For cross-modal information alignment task, we proposes a character-level multi-modal information alignment approach using contrastive loss. Three different modality encoders are employed to map the input into three distinct vector spaces. Character-level contrastive loss is applied to spaticnarrow the distance between three modality vectors corresponding to the same Chinese character, achieving character-level multi-modal information alignment. The calculation of character-level contrastive loss is formulated in Equations (20),s (21) and (22):

$$\ell_3 = \frac{1}{4B}\sum_{i=1}^{B} \tilde{\ell}^i_3 \tag{20}$$

$$\tilde{\ell}_3^i = \tilde{\ell}(H_i^g, H_i^s) + \tilde{\ell}(H_i^s, H_i^g) + \tilde{\ell}(H_i^p, H_i^s) + \tilde{\ell}(H_i^s, H_i^p) \quad (21)$$

$$\tilde{\ell}(H_i^1, H_i^2) = -\frac{1}{l}\sum_{j=1}^{l} \log \frac{exp(D(h_{i,j}^1, h_{i,j}^2)/\tau)}{\sum_{k=1}^{l} exp(D(h_{i,j}^1, h_{i,k}^2)/\tau)} \quad (22)$$

Where B represents batch size, l represents the length of the text sequence, H^g, H^p, and H^s denote the outputs of the encoders for character glyph, pinyin, and semantics, respectively. H_i^g, H_i^p, and H_i^s represent the feature vectors of character glyph, pinyin, and semantics of the i-th text sequence in a batch. The temperature constant τ in Equation (22) is used to adjust the attention to difficult samples.

Contrastive loss ℓ_3 consists of four parts, $\tilde{\ell}(H_i^g, H_i^s)$, $\tilde{\ell}(H_i^s, H_i^g)$,$\tilde{\ell}(H_i^p, H_i^s)$,$\tilde{\ell}(H_i^s, H_i^p)$, representing the calculation of contrastive loss between character glyph information and semantic information, semantic information and glyph information, pinyin information and semantic information, and semantic information and pinyin information, respectively. $H_i^1 = \{h_{i,1}^1, ..., h_{i,l}^1\}$ and $H_i^2 = \{h_{i,1}^2, ..., h_{i,l}^2\}$ represent the feature vector outputs of text sequences for modality 1 and modality 2, respectively. $\tilde{\ell}(H_i^1, H_i^2)$ calculates the character-level contrastive loss between a feature vector of a modality and another feature vector of a different modality in a text sequence. ℓ_3 denotes the average contrastive loss of a batch.exp() represents the exponential function with base e.

$$D(a,b) == \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}} \quad (23)$$

Equation (23) shows the calculation of the distance function $D(a,b)$ between vectors a and b using the cosine similarity function.

3.3.4 Overall Loss

The overall loss of the algorithm is formulated in Equation (24),

$$\ell = 0.8 * \ell_1 + 0.1 * \ell_2 + 0.1 * \ell_3 \quad (24)$$

As shown in Table 4, after testing various combinations of different weights, weights of 0.8, 0.1, and 0.1 are set for the correction loss, detection loss, and contrastive loss, respectively, to form the overall loss function.

4 Experimental design and analysis

4.1 Dataset and evaluation metrics

The datasets used for training in this paper include the Wikipedia corpus[3], Wang271K [14][4], SIGHAN2013 [16], SIGHAN2014 [21], SIGHAN2015 [13], and a Chinese character image library generated using Pillow.The number of sentences, average sentence length, and number of errors in the Chinese text correction dataset are shown in Table 2. The evaluation metrics of the algorithm are precision, recall, and F1 score at sentence level.

4.2 Training

Training consists of two phases: pre-training and fine-tuning. Pre-training is with the purpose to provide the three different modality encoders with powerful information representation capabilities. Chinese Wikipedia corpus is used for pre-training.The glyph encoder is initialized with the default weights of BERT, and then trained on the task of restoring the input text.The training of the pinyin encoder is to predict the Chinese character corresponding to the input pinyin.The semantic encoder directly loads the public weights of MacBERT, therefore does not undergo additional pre-training. For fine-tuning, the three encoders together with the subsequent decoder network, error probability network are trained on SIGHAN and Wang271K dataset for error correction tasks to adjust the network parameters, allowing the network to perform better on error correction tasks.

[3] http://download.wikipedia.com/zhwiki

[4] https://github.com/wdimmy/Automatic-Corpus-Generation/blob/master

Table 2. The Wang271K is an automatically generated dataset for CSC tasks, with more than 270,000 pairs of sentences. The SIGHAN13, SIGHAN14, and SIGHAN15 datasets are benchmarks of spelling error correction tasks. Table 2 lists basic statistics of these datasets.

Training Data	#Lines	Avg.Length	#Errors
Wang271K	271329	42.6	381962
SIGHAN13	350	49.3	339
SIGHAN14	3437	49.6	5136
SIGHAN15	2339	31.3	3048
Testing Data	**#Lines**	**Avg.Length**	**#Errors**
SIGHAN13	1000	74.3	1221
SIGHAN14	1062	50.0	771
SIGHAN15	1100	30.7	705

4.3 Analysis of experimental results

Table 3 presents the comparison of the MIATS with other baseline methods.

Table 3. On the three datasets of SIGHAN13, SIGHAN14, and SIGHAN15, the F1 score of MIATS is better than other baselines.

Testing Data	Baseline	Sentence-Level		
		pre	rec	F1
SIGHAN13	NRI [19](Character Level)	70.3	62.5	66.2
	LMC [17](Character Level)	77.6	22.7	35.1
	SL [14](Character Level)	(-)	(-)	52.1
	PN [15](Character Level)	79.7	59.4	68.1
	BERT [5]	97	44	60.6
	FASpell [7]	73.1	60.5	66.2
	SpellGCN [3]	78.3	**72.7**	75.4
	MIATS	**99.3**	65.4	**78.9**
SIGHAN14	HM [25]	55.5	39.1	45.9
	SL [14](Character Level)	(-)	(-)	56.1
	BERT [5]	64.8	49.4	56.1
	FASpell [7]	59.4	52	55.4
	SpellGCN [3]	63.1	**67.2**	65.3
	MIATS	**69.5**	65.4	**67.3**
SIGHAN15	LMC [17](Character Level)	71.1	50.2	58.8
	SL [14](Character Level)	(-)	(-)	57.1
	PN [15](Character Level)	71.5	59.5	69.9
	HanSpeller++ [24]	79.7	51.5	62.5
	FASpell [7]	66.6	59.1	62.6
	Soft-Masked BERT [23]	66.7	66.2	66.4
	BERT [5]	80.3	62.7	70.4
	MacBERT4CSC [12]	82.6	73.6	77.8
	SpellGCN [3]	72.1	**77.7**	75.9
	MIATS	**84.2**	73.4	**78.4**

It can be observed that overall, MIATS outperforms most other

methods on the SIGHAN dataset and achieves very high precision on all three datasets, indicating strong error correction abilities. More information should be given on baseline SpellGCN, its approach to spell correction and why it achieved better recall values than the proposed model.SpellGCN utilizes GCN (Graph Convolutional Network) to learn the embedding of characters, which might better capture the similarity and correlation between characters, thus is able to locate the misspelled characters more accurately. Therefore, the average recall rate of SpellGCN on the SIGHAN dataset is nearly 4% higher than ours. Nonetheless, the precision of our MIATS is nearly 10% higher than that of SpellGCN on average. Overall, the F1 value of MIATS is also better than that of SpellGCN. This has fully verified that the multi-modal encoder possesses stronger representation ability, and the consideration of detection loss as well as contrastive loss allows MITAS to learn additional knowledge and improve its ability to detect and correct errors.

To verify the effectiveness of detection loss and contrastive loss, and the impact of the weight parameters of each loss on error correction results, we have conducted the following comparative experiments:

1. Set the weights of error correction loss and contrastive loss to be constant, and change the weight of detection loss. We then test the algorithm on the SIGHAN13 test set.
2. Set the weights of error correction loss and detection loss to be constant, and change the weight of contrastive loss. We then test the algorithm on the SIGHAN13 test set.

Table 4. Effect of different training weights combination, tested on the SIGHAN13 dataset.

correction	detection	contrastive	pre	rec	F1
0.8	0	0.1	**99.5**	61.3	75.8
0.8	0.05	0.1	98.8	63.7	77.4
0.8	0.2	0.1	98.2	65.5	78.5
0.8	0.8	0.1	94.4	**67.1**	78.2
0.8	0.1	0.1	99.3	65.4	**78.9**
0.8	0.1	0	98.7	64.5	78
0.8	0.1	0.05	99.2	64.9	78.4
0.8	0.1	0.2	98	61.7	75.7
0.8	0.1	0.8	97.8	59.2	73.7
0.8	0.1	0.1	**99.3**	**65.4**	**78.9**

The experimental results are presented in Table 4. It is obvious that varying the weights of detection loss and contrastive loss affects error correction results. When the weights of error correction loss and contrastive loss are constant, increasing the weight of detection loss improves the ability to detect errors, resulting in a higher recall rate, but the precision rate correspondingly decreases. When the weights of error correction loss and detection loss are constant, increasing the weight of contrastive loss leads to a greater drop in performance. The intuitive explanation is that, a large contrastive loss makes the three modalities spatially close, resulting in weakened robustness and poorer performance. Properly balancing the weight proportions of error correction loss, detection loss, and contrastive loss yields an optimal ratio of 8:1:1.

To verify the effectiveness of the cross-modal information alignment, we have constructed the similarity matrix between different modal vectors. As an example, Figure 5 shows the similarity between the semantic vector and pinyin vector of the text sequence " 自然语言处理 " before and after the alignment. Before conducting the alignment process, the similarity values on the main diagonal were not always the maximum in the corresponding row or column. After the alignment, the maximum values of the rows and columns are on the main diagonal, indicating that the network has aligned the cross-modal information and have stronger representation ability. Similar phenomena have also been observed in the similarity matrix of semantic vector and glyph vector, as well as glyph vector and pinyin vector.

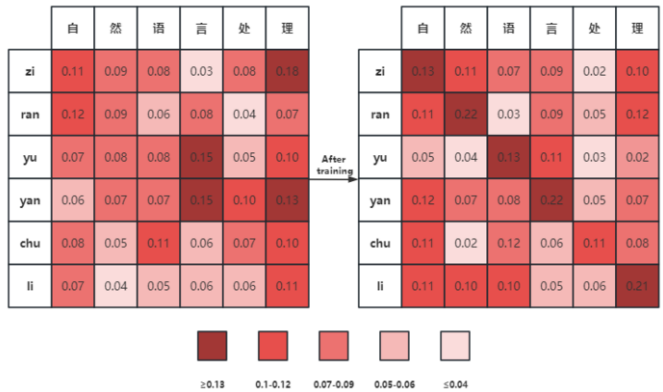

Figure 5. Similarity matrix before and after cross-modal information alignment of the semantic vector and pinyin vector of the text sequence.

5 Conclusion

This paper proposes a Chinese text correction algorithm MIATS, based on tri-tower multi-modal information alignment. A tri-tower multi-modal structure is used as the encoder to extract glyph, pinyin, and semantic information. The decoder part explores the cross-modal information interaction and facilitates better context information utilization with an error probability network and transformer network. During the training process, special training tasks are designed, including correction task, detection task, and cross-modal information alignment task. The effectiveness of the proposed algorithm is fully verified on the SIGHAN datasets.

References

[1] Haithem Afli, Zhengwei Qiu, Andy Way, and Páraic Sheridan, 'Using SMT for OCR error correction of historical texts', in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 962–966, Portorož, Slovenia, (May 2016). European Language Resources Association (ELRA).

[2] Chao-Huang Chang, 'A new approach for automatic chinese spelling correction', in *Proceedings of Natural Language Processing Pacific Rim Symposium*, volume 95, pp. 278–283. Citeseer, (1995).

[3] Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi, 'SpellGCN: Incorporating phonological and visual similarities into language models for Chinese spelling check', in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 871–881, Online, (July 2020). Association for Computational Linguistics.

[4] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu, 'Revisiting pre-trained models for Chinese natural language processing', in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pp. 657–668, Online, (November 2020). Association for Computational Linguistics.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, 'BERT: Pre-training of deep bidirectional transformers for language understanding', in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*,

pp. 4171–4186, Minneapolis, Minnesota, (June 2019). Association for Computational Linguistics.

[6] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural computation*, **9**(8), 1735–1780, (1997).

[7] Yuzhong Hong, Xianguo Yu, Neng He, Nan Liu, and Junhui Liu, 'Faspell: A fast, adaptable, simple, powerful chinese spell checker based on dae-decoder paradigm', in *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pp. 160–169, (2019).

[8] F. Jelinek and R. L. Mercer, 'Interpolated estimation of markov source parameters from sparse data', in *Proceedings of the workshop on pattern recognition in practice*, pp. 381–397. Elsevier Science Publishers B.V., (1980).

[9] Yichong Leng, Xu Tan, Linchen Zhu, Jin Xu, Renqian Luo, Linquan Liu, Tao Qin, Xiangyang Li, Edward Lin, and Tie-Yan Liu, 'Fastcorrect: Fast error correction with edit alignment for automatic speech recognition', *Advances in Neural Information Processing Systems*, **34**, 21708–21719, (2021).

[10] Chao-Lin Liu, Min-Hua Lai, Yi-Hsuan Chuang, and Chia-Ying Lee, 'Visually and phonologically similar characters in incorrect simplified chinese words', in *Coling 2010: Posters*, pp. 739–747, (2010).

[11] Bruno Martins and Mário J Silva, 'Spelling correction for search engine queries', in *Advances in Natural Language Processing: 4th International Conference, EsTAL 2004, Alicante, Spain, October 20-22, 2004. Proceedings 4*, pp. 372–383. Springer, (2004).

[12] Xu Ming. Pycorrector: Text error correction tool. https://github.com/shibing624/pycorrector, 2021.

[13] Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen, 'Introduction to sighan 2015 bake-off for chinese spelling check', in *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pp. 32–37, (2015).

[14] Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang, 'A hybrid approach to automatic corpus generation for chinese spelling check', in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2517–2527, (2018).

[15] Dingmin Wang, Yi Tay, and Li Zhong, 'Confusionset-guided pointer networks for chinese spelling check', in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5780–5785, (2019).

[16] Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee, 'Chinese spelling check evaluation at sighan bake-off 2013', in *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pp. 35–42, (2013).

[17] Weijian Xie, Peijie Huang, Xinrui Zhang, Kaiduo Hong, Qiang Huang, Bingzhou Chen, and Lei Huang, 'Chinese spelling check system based on n-gram model', in *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pp. 128–136, (2015).

[18] Yi Yang, Pengjun Xie, Jun Tao, Guangwei Xu, Linlin Li, and Luo Si, 'Alibaba at ijcnlp-2017 task 1: Embedding grammatical features into lstms for chinese grammatical error diagnosis task', in *Proceedings of the IJCNLP 2017, Shared Tasks*, pp. 41–46, (2017).

[19] Jui-Feng Yeh, Sheng-Feng Li, Mei-Rong Wu, Wen-Yi Chen, and Mao-Chuan Su, 'Chinese word spelling correction based on n-gram ranked inverted index list', in *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pp. 43–48, (2013).

[20] Junjie Yu and Zhenghua Li, 'Chinese spelling error detection and correction based on language model, pronunciation, and shape', pp. 220 – 223, Wuhan, China, (2014). Character level;Delimiters;Error detection and correction;Language model;N-gram language models;Spelling checks;Spelling errors;User-generated;Word level;Written texts;.

[21] Liang-Chih Yu, Lung-Hao Lee, Yuen-Hsien Tseng, and Hsin-Hsi Chen, 'Overview of sighan 2014 bake-off for chinese spelling check', in *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pp. 126–132, (2014).

[22] Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang, 'Correcting chinese spelling errors with phonetic pre-training', in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 2250–2261, (2021).

[23] Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li, 'Spelling error correction with soft-masked BERT', in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 882–890, Online, (July 2020). Association for Computational Linguistics.

[24] Shuiyuan Zhang, Jinhua Xiong, Jianpeng Hou, Qiao Zhang, and Xueqi Cheng, 'Hanspeller++: A unified framework for chinese spelling correction', pp. 38 – 45, Beijing, China, (2015). Decisions makings;Foreign language;Non-native speakers;Performance;Reranking;Spelling checks;Spelling correction;Test data;Unified framework;.

[25] Hai Zhao, Deng Cai, Yang Xin, Yuzhu Wang, and Zhongye Jia, 'A hybrid model for chinese spelling check', *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, **16**(3), 1–22, (2017).

[26] Renjie Zheng, Mingbo Ma, Baigong Zheng, Kaibo Liu, and Liang Huang, 'Opportunistic decoding with timely correction for simultaneous translation', in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 437–442, Online, (July 2020). Association for Computational Linguistics.