Languages, Literatures and Cultures
2026, Vol. 2, No. 2, pp. 91–98
https://doi.org/10.11648/j.llc.20260202.12

SciencePG
Science Publishing Group

Research Article

A Korean Handwritten Text Recognition Method Based on CSWin Transformer and GPT Language Model

Il-Nam Pak, Myong-Chol Kim, Hyon-Gwang O, HakHo Hong*

Institute of Mathematics, State Academy of Sciences, Pyongyang, Democratic People's Republic of Korea

Abstract

HTR is a challenging problem in the field of pattern recognition, as there are many variations in writing style, slope, size, noise, etc., depending on the individual writter's habits, unlike PTR(Print Text Recognition). In this paper, we design a CSWin transformer-based encoder and a GPT-2 language model based decoder to improve the per-formance of Korean handwritten text recognition. We have improved CSWin transformer based on the self-attention in horizontal and vertical stripes to suit the characteristics of handwritten text recognition, used it as an encoder. The main features of CSWin transformer is the self-attention modules in the horizontal and vertical stripes and local enhancement positional encoding module. The local enhancement positional encoding module can support any image resolution, and is especially advantageous for HTR with line width varying frequently. In this paper, we use the local enhance positional encoding module. We also applied a projection transform to the encoder's features to transform to language-like tokens, and designed a decoder based on the Korean language model. we constructed a Korean subword vocabulary based on the byte pair encoding algorithm. The text corpus used for subword vocabulary construction contains 16,181,377 lines extracted from newspapers, common sense, novels, dialogues, and dictionaries. Since in Korean, too long sequence can be selected into a subword unlike English, so the maximum length of a subword is set to 5 to prevent the length of a subword from being too long when constructing a subword vocabulary. The final subword vocabulary contains 15477 subwords, and we use these subwords as the language tokens. Consequently, we have proposed a Korean handwritten text recognition method consisting of a CSWin transformer-based encoder and a GPT language model-based decoder. Experimental results show that our method is an efficient and effective method for Korean handwriting text recognition.

Keywords

CSWin Transformer, GPT-2, Language Model, Encoder, Decoder

1. Introduction

HTR(Handwritten Text Recognition) has important applications, including converting historical documents into digital ones. However, HTR is a challenging problem in the field of pattern recognition, as there are many variations in writing style, slope, size, noise, etc., depending on the individual writter's habits, unlike PTR(Print Text Recognition). Although many methods have been developed to detect and recognize individual characters or words, these methods have the low recognition rate due to the low accuracy of character and word

*Correspondence: Hakho Hong (hhhong@star-co.net.kp)

Received: 14 December 2025; **Accepted:** 15 January 2026; **Published:** 10 June 2026

detection and the lack of learning the language context. To solve this problem, many approaches have been developed to recognize text images at the line, paragraph, or page level [12]. Line-level HTR methods can be divided into full-convolution networks+CTC [7], and methods using Transformer [1, 2, 4, 5, 11], depending on the structure of the deep learning model. Recently, the use of transformer has been widely studied and applied to text recognition, with many advances in recognition performance. In particular, the combination of visual transformer models and language models has led to the great sucesses in HTR [1, 2, 4-6, 8, 11]. Just as a person has to use the language contextual information as well as individual characters to recognize the texts written by another person, so the language model in HTR is very important.

In Li's study [4], a TrOCR model using DeiT as visual encoder and RoBERTa language model as decoder is proposed by Liu [10]. This method achieved 2.89% CER in the IAM database. In [6], they reached 1.5% CER in the handwriting lemma recognition of the medieval atin dictionary using the Swin Transformer and GPT2 language model. In [2, 5], a DTrOCR model, decoder-only transformer model was proposed. This method achieved 1.7% CER in the IAM database using GPT-2 decoder model pretrained in a large text corpus [3].

Figure 1 shows the models of text recognition methods using the transformers studied to date. In this figure, (a)-(c) is the encoder-decoder model, and (d) is the decoder-only model. (c) is a model that uses an external language model like ABINet, which uses the language model as a spell checker. (b) and (d) are models that use the internal language model, which has recently been recognized as the best performance method.

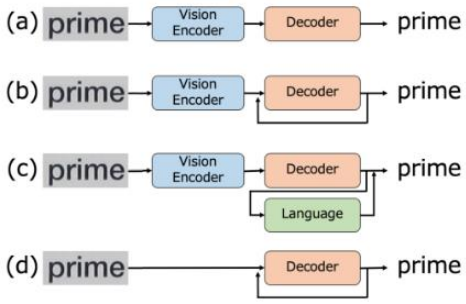

Figure 1. *Encoder-decoder(decoder-only) models for text recognition.*

It is not clear which of these two models is better. Indeed, for English, the decoder-only model has been shown to have better recognition performance than the encoder + decoder model, but slightly less for Chinese [5]. We have analyzed some advantages and disadvantages of TrOCR and DTrOCR methods, and proposed to improve the recognition performance and speed of the Korean HTR using CSWin-based vision encoder and decoder-only transformer. The paper is organized as follows. Section 2 briefly reviews the TrOCR and DTrOCR and analyzes them from the viewpoint of recognition performance and efficiency. Section 3 presents the encoder design method for HTR based on CSWin transformer. We also describe a decoder design method based on the language- like token features and the GPT language model. Section 4 demonstrates the effectiveness of our method through the experiments, and Section 5 present the conclusions and future work.

2. Related Work

First, we briefly review the TrOCR method studied in [4]. Figure 2 shows the overall architecture of the TrOCR.

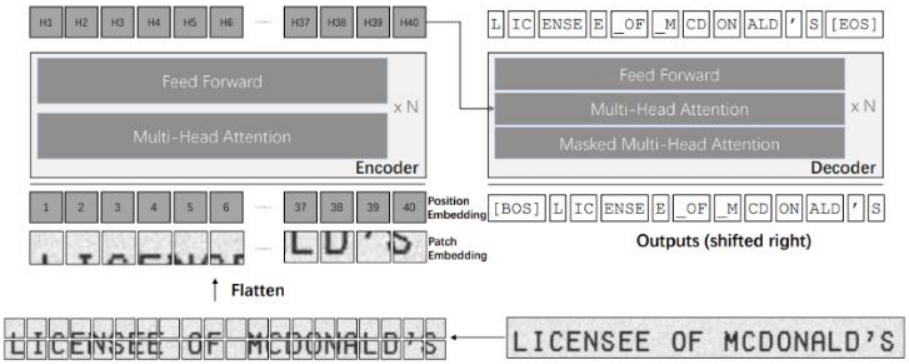

Figure 2. *The overall architecture of TrOCR.*

This architecture is a typical encoder + decoder model, consisting of a DeiT-based encoder and a RoBERTa-based decoder. However, this method requires the cross-attention between the encoder and the decoder, and thus has the disadvantage that the language model cannot be used as it is. In the TrOCR method, the cross-attention between the text image features and the decoder is used many times, so the decoder is

significantly different from the pre-trained language model. To solve this problem, TrOCR proposes to keep the weights of pre-trained language model, randomly initializes the weights corresponding to the cross-attentions, and then train the whole model. Also, DeiT, the encoder used in TrOCR, has high computational complexity and is not friendly to varying input resolutions.

Unlike TrOCR, DTrOCR has the advantage of better utilizing the capability of the language model than TrOCR because the decoder contains no cross-attention by using a decoder-only transformer [5]. Figure 3 shows architecture of the DTrOCR method.

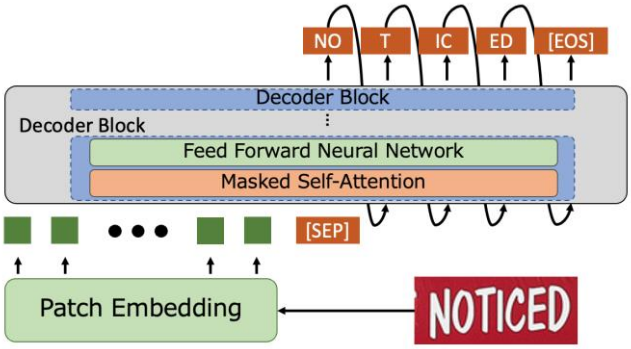

Figure 3. *The architecture of DTrOCR.*

First, we decompose the input image $x \in R^{3\times H\times W}$ into non-overlapping image patches of size $P_H \times P_W$.

Then, the input image is decomposed into $N = H/P_H \times W/P_W$ patches.

Each patch is transformed into a column vector and mapped into a D-dimensional vector. Adding the position encoding vector to this vector, we finally obtain N D-dimensional vectors. This image feature sequence is inputed to decoder-only transformer and autoregressively outputs the text. This method has the advantage of constructing a decoder using the language model, not requiring cross-attention, unlike the TrOCR method. The disadvantages of this method are analyzed as follows.

First, patch embedding features and language token features are very heterogeneous features, so we can't directly use the pre-trained GPT language model. To solve this problem in DTrOCR, the decoder was trained using artificially generated 4 billion text-image pairs.

Second, to input patch embedding feature to the decoder directly is not efficient in the aspect of computational complexity. In [2, 5], the text image is decomposed into 4×8 non-overlapping blocks, so that $N = H/4 \times W/8$ patch features are inputed to the decoder. In DTrOCR, 12 transformer blocks were used, such as GPT-2. Since the attention module, the main module of the transformer block, requires computational complexity, $O(N^2)$, it can be said that reducing the number of patches N, is essential to reduce the computational complexity of the decoder.

To solve these problems, we propose to convert the patch embedding features into N = W/8 features using our encoder. If H = 48, the computational complexity of the decoder can be greatly reduced since a sequence of feature vectors 12 times shorter than the DTrOCR method is inputed to the decoder.

In particular, we used the projection transformation to transform the encoder features into language-like token features without inputing to the decoder directly, so that we can perform handwritten text recognition by simply finetuning the decoder based on the language model without much effort.

3. Proposed Method

3.1. CSWin Transformer Based Encoder

In [13], a CSWin transformer, which is efficient and effective and naturally supports arbitrary input resolutions, has been proposed. We design an encoder based on the CSWin transformer, considering that this transformer is very light and the width of the line image is frequently variable. The CSWin transformer has surpassed the DeiT and Swin based vision transformers in the recognition performance and speed on several databases including IMAGENET and MSCOCO [13].

Figure 4 shows the overall structure of the CSWin transformer.

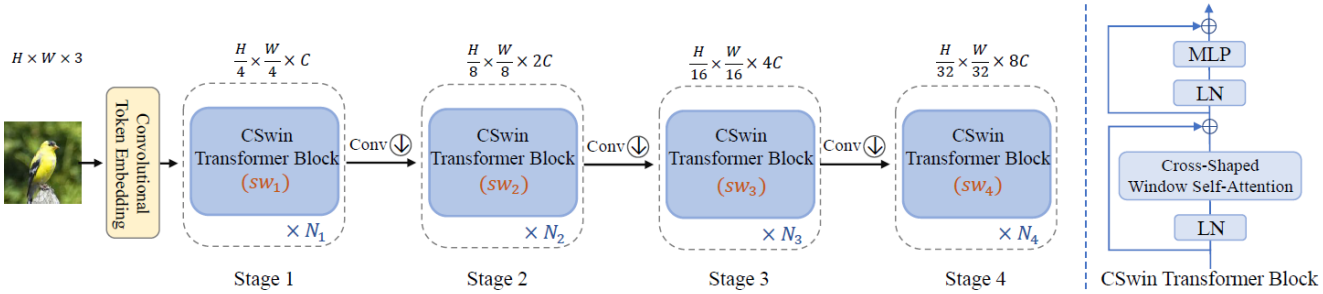

Figure 4. *The overall architecture of CSWin transformer and CSWin transformer block.*

The main features of CSWin transformer is the self-attention modules in the horizontal and vertical stripes and local enhancement positional encoding module. The local enhancement positional encoding module can support any image resolution, and is especially advantageous for HTR with line width varying frequently.

In this paper, we use the local enhance positional encoding module as it is, so we do not discuss about it anymore, but rather consider the self attention in horizontal and vertical stripes in more detail.

For the self-attention in horizontal stripes, X is evenly partitioned into non-overlapping horizontal stripes $[X^1, \dots, X^M]$ of equal height sH, and each of them contains sH × W tokens. Here, sH is the stripe height and can be adjusted to balance the learning capacity and computation complexity.

Formally, suppose the projected queries, keys and values of the k^{th} head all have dimension d_k, then the output of the self-attention in horizontal stripes for k^{th} head is defined as:

$$Y_k^i = Attention(X^i W_k^Q, X^i W_k^K, X^i W_k^V)$$

$$H - Attention_k(X) = [Y_k^1, \dots, Y_k^M],$$

where $X^i \in R^{(sH \times W) \times C}, M = \frac{H}{sH}, i = 1, \dots, M$, and $W_k^Q \in R^{C \times d_k}, W_k^K \in R^{C \times d_k}, W_k^V \in R^{C \times d_k}$ are projecti-on matrices of queries, keys and values for the k^{th} head respectively, and $d_k = C/K$.

The self-attention in vertical stripe for the k^{th} head is defined similarly, which is denoted by $V - Attention_k(X)$. Finally the output of these two parallel groups will be concatenated back together.

$$CSWin - Attention(X) = Concat(head_1, \dots, head_K)W^O,$$

$$head_k = \begin{cases} H - Attention_k(X), k = 1, \dots, K/2 \\ V - Attention_k(X), k = \frac{K}{2} + 1, \dots, K' \end{cases} W^O \in R^{C \times C}$$

In [13], they set sW = sH and set sW differently for each stage. However, for HTR, so if we set sW = sH, the computational complexity of the horizontal strip attention is much larger than that of vertical strip attention.

On the other hand, in CSWin computation, both types of attentions are executed in parallel, so to perform the operation efficiently, we are going to balance the operations of two parallel groups of attentions.

Therefore, we set sH ≪ sW. For example, we set sH = 1, sW = 4. Comparing this with sH = 1, sW = 1, we can better model the relationship between local character strokes without requiring additional computational by reflecting the relationship between features along a vertical strip of width 4 rather than simply the relationship between features along a vertical line.

Meanwhile, we modify the structure of the CSWin transformer shown in Figure 4 to suit the characteristics of HTR.

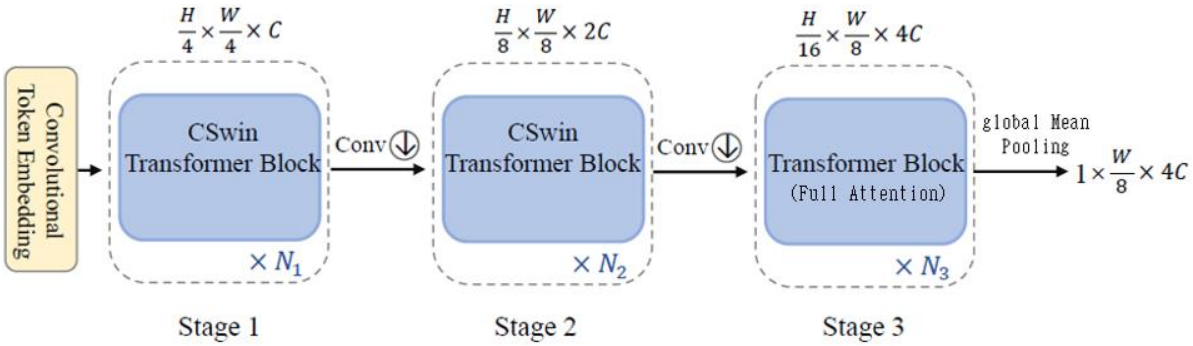

Figure 5. *The overall structure of our encoder.*

First, line's height, H is set to 48, and taking this into account, the number of stages of CSWin transformer is set to 3. In addition, the CSWin transformer blocks are used only in stages 1 and 2. To prevent too much reduction in the horizontal direction, we set the horizontal stride to 1 and the vertical stride to 2 for the convoultion after stage 2 is finished. In stage 3, unlike stages 1, 2, we design to capture the global context characteristics of the handwritten text image by using the full attention. In fact, the spatial size of feature map in the stage 3 is 3 ×W/8, which is much reduced compared to the original input feature map, so that the full attention does not increase the computational complexity greatly. Finally, passing through the above three stages, we can capture the global context characteristics as well as the relationships between local strokes in handwritten text image. After stage 3, we apply global average pooling in the vertical direction, and text image feature sequence of 1 ×W/8 ×4C is finally obtained. We set C = 64, so eventually we obtain the feature sequence of 1 ×W/8 ×256. Figure 5 shows the overall structure of our encoder.

3.2. A Language Model Based Decoder

We describe the design of our proposed decoder. When recognizing using the encoder-decoder model, the input is a text image feature sequence and the output is a token sequence, which is essentially an image feature sequence-to-token sequence modeling problem. Thus, the encoder-decoder model predicts the subsequent token in an autoregressive manner based on the preceeding tokens and the image features.

$$P(\mathrm{w} = \{\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_m\}|\mathrm{x}) = \prod_{k=1}^{m} \mathrm{P}(\mathrm{w}_k|\mathrm{w}_1, \ldots, \mathrm{w}_{k-1}, \vec{x}),$$

where $\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_{k-1}$ are token sequence up to time k-1 steps, w_k is a token a time step k, $\vec{x}$ represents encoder feature sequence, namely text image feature sequence.

As can be seen from the above expression, the problem of estimating the corresponding token sequence given an image feature sequence $\vec{x}$ is closely related to the language model P that estimates the token at time k, knowing the preceeding tokens. In particular, language modeling is an essential problem because it is difficult to recognize by interpreting only individual characters because the number of classes and ambiguous characters are relatively large for Korean HTR. We modify the above expression as follows.

$$P(\mathrm{w} = \{\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_m\}|\mathrm{x}) = \prod_{k=1}^{m} \mathrm{P}(\mathrm{w}_k|\vec{x}, \mathrm{w}_1, \ldots, \mathrm{w}_{k-1})$$

If we can treat text image feature sequence as same as the language token sequence, this problem can be considered as a problem of predicting the next text based on the already pretrained language model P, given the prompt $\vec{x}$. Based on this observation, we construct a language token projector E that maps a text image feature sequence into a language token space.

Language token projector

The purpose of the language token projector is to map a text image feature sequence $\vec{x} = \left\{x_i \in R^{256}, i = 1,2, \ldots, \frac{W}{8}\right\}$ to the language token space, and obtain the language-like token feature sequence $\vec{y} = \left\{y_i \in R^{256}, i = 1,2, \ldots, \frac{W}{8}\right\}$.

We simply design the language token projector E as a FFN-like structure used in the transformer block, i.e., two fully connected layers.

Language models

First, we constructed a Korean subword vocabulary based on the byte pair encoding algorithm.

The text corpus used for subword vocabulary construction contains 16,181,377 lines extracted from newspapers, common sense, novels, dialogues, and dictionaries. Since in Korean, too long sequence can be selected into a subword unlike English, so the maximum length of a subword is set to 5 to prevent the subword' length from being too long when constructing a subword vocabulary. The final subword vocabulary contains 15477 subwords, and we use these subwords as the language tokens.

We use the GPT-2 model as a language model, as well as the DTrOCR method. Considering the scale of the text corpus used for training is too small, we reduced the number of transformer blocks to 6. Although the ability of the language model may be not high due to the too small number of transformer blocks, we perform the recognition using the text image features obtained from the encoder as a main tool and the language model as an auxiliary tool, so we set the number of transformer blocks to small. The maximum length of the input sequence is set to 512, and the dimension of the transformer is set to 256 as well as the encoder. The positional encoding module uses the rotary encoding method. The text corpus used for constructing the subword dictionary is also used for training the GPT-2 language model. By training using the method described above, a language model L that predicts the next token given a token sequence is obtained.

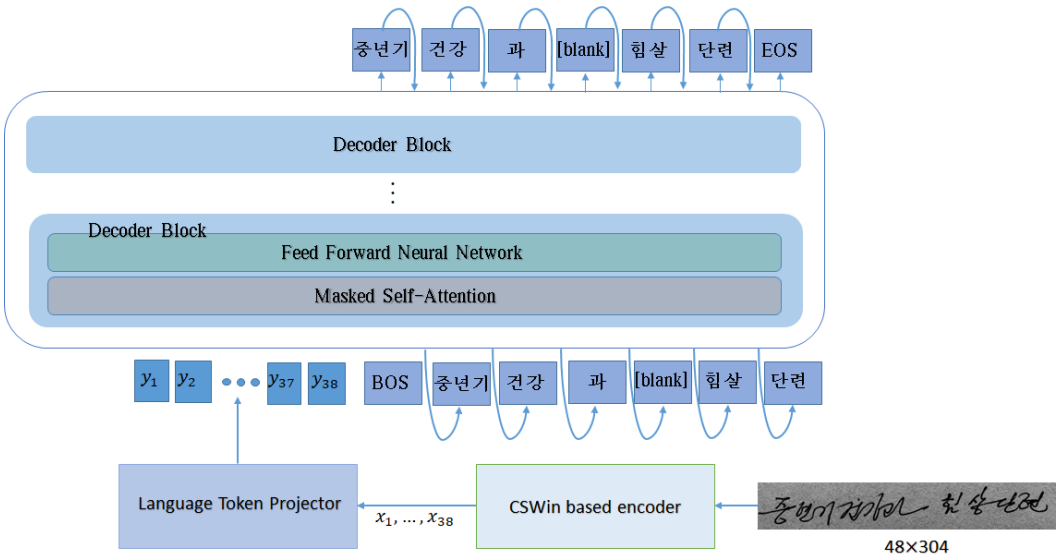

Figure 6. The overall structure of our proposed method.

Decoder design

The decoder has two stages. First, we transform the text image feature sequence $\vec{x}$ into the language-like token feature sequence $\vec{y}$ using the language token projector E. Then the language-like token feature sequence $\vec{y}$ is entered into the language model L. The token embedding in the language model does not work for $\vec{y}$ but only for real language tokens. The $\vec{y}$ and the start token [BOS] are inputed to L to predict the first token and the text corresponding to the text image feature sequence is predicted in an autoregressive manner until the end token [EOS] is obtained. Our decoder uses no any cross attention between encoder features and decoder unlike the typical encoder-decoder-based transformer. Therefore, if the language-like token features are really mapped into language token space, we can almost literally use the language model trained only in the text corpus. Figure 5 shows the overall architecture of our proposed method.

4. Experiment

4.1. Database

We perform training and evaluation using "RanPil", a training and evaluation database for Korean handwriting recognition [9]. This database consists of a total of 8600 pages containing 18,2000 lines written by 1,804 people. Of these, the training database contains 7600 pages written by 1696 writters, and the test database contains 1000 pages written by 108 writters. Because of the small size of the training database, we manually created 50 handwritten fonts and added 5 million line images to the training database using Text Render from the text corpus(https://github.com/oh-my-ocr/text_renderer). Together, 396 print fonts were used by Text Render to generate 5 million print lines and added to the training database. On the other hand, “RanPil” test database is used as the test database. In this database, we perform the performance evaluation on only the whole test database, not categorical evaluation.

4.2. Training Methods

The training is largely divided into encoder model training, language model training, and encoder-decoder model training.

Encoder model training

For encoder training, we use CTC loss. For the training data, the transformations of zoom, perspective, motion blur, Gaussian blur and Gaussian noise are applied with a probability of 0.3. We use the optimization algorithm as AdamW, the number of mini-batch 128, the learning rate 5e-4, the learning rate schedule method cosine annealing, the weight decay coefficient 0.05, and the epoch 50.

Language model training

As we have described earlier, the text database used for language model training contains very small lines.

The loss function uses the cross-entropy loss. We use the optimization algorithm as AdamW, the number of mini-batch 512, the learning rate 2e-4, the learning rate schedule method cosine annealing, the weight decay coefficient 0.05, and the epoch 50. Because of the small number of training database, the language model has not high ability, but this model contributes significantly to the performance.

Encoder-decoder model training

After the encoder and language model training has been completed, the encoder-decoder model training is performed. As the training of the encoder and the language model has already been done, we focus on the training of the language token projector. Therefore, we set the learning rate for the language token projector to be much greater than that of the encoder and language models, so that training can be performed in a finetuning manner, so that the encoder and language models can be maintained. We use the optimization algorithm as AdamW, the number of mini-batch 128, the learning rate schedule method cosine annealing, the weight decay coefficient 0.05, and the epoch 50. The initial learning rate for the language token projector is set to 5e-4, and for the encoder and language model to 1e-5.

4.3. Comparison with Previous Methods

First, we evaluate the performance of our encoder. We set the parameters of the CSWin transformer as dimension 64, block numbers 1, 2, 21, and mult-head numers 2, 4, and 8, respectively as well as CSWin-T.

The recognition performance is evaluated using CER. In stage 3, we use the full attention, so we set sW, sH only in stage 1 and 2. The input image resolution was set to 48 x 864, and the speed estimation was performed. The computer used for the performance evaluation is as follows.

12th Gen Intel(R) Core(TM) i9-12900F 2.40 GHz

We implemented the neural networks using the onnx_runtime library. Table 1 shows the performance evaluation results, changing (sH, sW).

Table 1. *The performance comparison of encoders, changing (sH, sW).*

(sH, sW)	Inference time (ms)	CER(%)
(1, 1), (2, 2)	58.3	10.88
(1, 4), (2, 4)	58.2	9.78
DeiT-S	59.2	11.39

All encoders are trained using the database described in Section 4.1 and the encoder training method described in Section 4.2, and performance was evaluated. It can be seen from the table that three methods are similar in inference speed but when

(sH, sW) in stages 1 and 2 are set to (1, 4) and (2, 4), the recognition performance is slightly improved than (1, 1), (2, 2) as in CSWin. Therefore, we use the neural network in the case of setting (sH, sW) to (1, 4), (2, 4) respectively as an encoder.

Meanwhile, we compare our method with DeiT-S, the encoder of TrOCR. From the table, we can see that our encoder based on CSWin transformer outperforms the encoder used in TrOCR.

Next, we demonstrate the advantages of the decoder in our method, namely advantages of our language token projector through experiments. To do this, the encoder in the TrOCR method uses the same encoder as our proposed encoder and the decoder is replaced by the GPT language model trained in the way described in Section 3.2, rather than RoBERTa. Beam search is used to generate the final output.

Table 2 shows the performance comparison results of the two methods. It can be seen from the table that the proposed language token projector-based decoder architecture and training method improve the recognition performance over the TrOCR method. On the other hand, Table 2 estimates the average time taken to process a single line image of resolution 48 x 864. In our decoder, only the language token projector is added to the language model, but as the TrOCR method has to compute six cross-attention layers, thus this makes the computation of the decoder slower than our method.

Table 2. *The performance comparision with TrOCR.*

	CER(%)	Inference time(ms)
The proposed encoder + TrOCR	6.8	141
The proposed method	4.7	125

We also see from Tables 1 and 2 that the recognition performance is significantly improved by the language model-based decoder. Our language model is trained on very small text corpus, and therefore, the language modeling ability is not high. We believe that if we train the language model in much larger text corpus, using a larger GPT model, the ability of the language model will improve, and the handwriting recognition rate will be also improved.

5. Conclusions

In this paper, we have proposed Korean HTR using CSWin transformer-based encoder and language token projector+GPT-based decoder. The CSWin transformer is modified according to the characteristics of text recognition for better performence. We also propose a method to train Korean HTR in a simple finetuning method by converting the encoder features directly into language-like token features by introducing a language token projector. Through the experiments, we have shown that the proposed Korean HTR method improves the recognition performance and speed. In the future, we will further improve the performance of Korean HTR by boosting the performance of GPT-based Korean language model.

Abbreviations

GPT	Generative Pre-Training
CSWin	Cross-Shaped Window
TrOCR	Transformer-based Optical Character Recognition
DTrOCR	Decoder only Transformer Optical Character Recognition
CTC	Connectionisity Temporal Classification
CER	Character Error Rate

Funding

The authors declare that no fund and no support were received during the preparation of the research paper.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] Masato Fujitake, DIFFUSIONSTR: DIFFUSION MODEL FOR SCENE TEXT RECOGNITION, arXiv: 2306.16707v1, 2023.

[2] Masato Fujitake, JSTR: Judgment Improves Scene Text Recognition, arXiv: 2404.05967v1, 2024.

[3] Radford, Language Models are Unsupervised Multitask Learners, https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf, 2019.

[4] Minghao Li, TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models, SecarXiv: 2109.10282v5, 2022.

[5] Masato Fujitake, DTrOCR: Decoder-only Transformer for Optical Character Recognition, arXiv: 2308.15996v1, 2023.

[6] Philipp Koch, Gilary Vera Nunez, and Esteban Garces Arias, A tailored Handwritten-Text-Recognition System for Medieval Latin, arXiv: 2308. 09368v1, 2023.

[7] Denis Coquenet, Clément Chatelain, and Thierry Paquet, End-to-end Handwritten Paragraph Text Recognition Using a Vertical Attention Network, arXiv: 2012.03868v2, 2021.

[8] Sennrich, R et al, Neural machine translation of rare words with subword units, arXiv: 1508.07909, 2015.

[9] Hyon-Gwang O, Myong-Chol Kim, RanPil: New Dataset and Benchmark for Offline Handwritten Korean Text Recognition, International Journal on Data Science and Technology, 11, 2, 27-34, 2025.

[10] Liu, Y., RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv: 1907.11692, 2019.

[11] Pengyuan Lyu, Chengquan Zhang, Shanshan Liu, Meina Qiao, Maskocr: text recognition with masked encoder-decoder pre-training, arXiv: 2206.00311, 2022.

[12] Carlos Garrido-Munoz, Antonio Rios-Vila, and Jorge Calvo-Zaragoza, Handwritten Text Recognition: A Survey, arXiv: 2502.08417v1, 2025.

[13] Xiaoyi Dong1, Jianmin Bao, CSWin Transformer: A General Vision Transformer Backbone with Cross-Shaped Windows, arXiv: 2107.00652v3, 2022.