applied sciences

MDPI

Article

A Three-Stage Uyghur Recognition Model Combining the Attention Mechanism and Different Convolutional Recurrent Networks

Wentao Li, Yuduo Zhang *, Yongdong Huang, Yue Shen and Zhe Wang

College of Science, Dalian Minzu University, DaLian 116600, China; 18811715936@163.com (W.L.); huang_yongdong@163.com (Y.H.); 18840065479@163.com (Y.S.); zhewang25@iflytek.com (Z.W.)
* Correspondence: zhangyuduo@dlnu.edu.cn

Abstract: Uyghur text recognition faces several challenges in the field due to the scarcity of publicly available datasets and the intricate nature of the script characterized by strong ligatures and unique attributes. In this study, we propose a unified three-stage model for Uyghur language recognition. The model is developed using a self-constructed Uyghur text dataset, enabling evaluation of previous Uyghur text recognition modules as well as exploration of novel module combinations previously unapplied to Uyghur text recognition, including Convolutional Recurrent Neural Networks (CRNNs), Gated Recurrent Convolutional Neural Networks (GRCNNs), ConvNeXt, and attention mechanisms. Through a comprehensive analysis of the accuracy, time, normalized edit distance, and memory requirements of different module combinations on a consistent training and evaluation dataset, we identify the most suitable text recognition structure for Uyghur text. Subsequently, utilizing the proposed approach, we train the model weights and achieve optimal recognition of Uyghur text using the ConvNeXt+Bidirectional LSTM+attention mechanism structure, achieving a notable accuracy of 90.21%. These findings demonstrate the strong generalization and high precision exhibited by Uyghur text recognition based on the proposed model, thus establishing its potential practical applications in Uyghur text recognition.

Keywords: text recognition; CRNN; GRCNN; ConvNeXt; BiLSTM; attention mechanism; Uyghur

Citation: Li, W.; Zhang, Y.; Huang Y.; Shen Y.; Wang Z. A Three-Stage Uyghur Recognition Model Combining the Attention Mechanism and Different Convolutional Recurrent Networks. *Appl. Sci.* **2023**, *13*, 9539. https://doi.org/10.3390/app13179539

Academic Editor: Christos Bouras

Received: 11 July 2023
Revised: 17 August 2023
Accepted: 21 August 2023
Published: 23 August 2023

1. Introduction

The Uyghur language is difficult to recognize due to the sticky nature of its words, the variability of its characters, and the right-to-left nature of its writing. Specifically, the following four key problems need to be addressed: 1. Uyghur datasets are scarce. Uyghur scripts were recognized late and Uyghur script technology for printed text has only been studied in recent years. There is no standard Uyghur dataset in the world, so we need to build our own Uyghur dataset. 2. Uyghur characters have a multivariate nature, leading to a difficult recognition process. Uyghur characters are variable; the same character in different positions of the word can have different forms and a character showing different forms may be very similar to another character in another position. For example, the character "ﻛ" in the word "تىكىرقاش" will become "ﻜﻰ" before the character "ى" (Uyghur is written from right to left), and the first half of the two characters that are stuck together is quite similar to the character "گ", which may lead to unsatisfactory final results. 3. Uyghur has strong adhesion which makes the recognition process infinitely difficult. At the same time, it also causes all the letters in the same word to basically stick together without gaps, so that some of the Uyghur characters are extremely similar, such as the characters "ۆ" and "ۇ". This makes the recognition of Uyghur characters much more difficult than those of English or other languages. When using "connectionist temporal classification" (CTC) [1] for prediction, the same frame may contain information of two characters that

are stuck together, and the adjacent pages before and after the frame may appear with the same condition, which may contribute to the missed predictions, over-predictions, or wrong predictions in CTC, especially in the prediction of long words, such as the word " قىلغىنىنىڭ " being predicted as " قىلمىشنىڭ ". The combination of the character "ى" and the character "ن", "ى ", is predicted to be the character "ش " in the dense part of the middle of the word. We also found that the overall effect of decoding prediction using CTC was not excellent during experiments. 4. There are a vast number of Uyghur words. There are a lot of common words in Uyghur and Uyghur sentences are even more diverse after combination. Therefore, to accurately identify Uyghur words in a line of text requires training on a large-scale dataset.

To address the above challenges, the research aim of this paper is as follows. First, a comprehensive and usable Uyghur text dataset is produced by taking line-level printed Uyghur text as the research object. After that, we propose a three-stage model structure for Uyghur text recognition for the Uyghur dataset by combining models that have been applied and not used for Uyghur text recognition before for training. For the variability of Uyghur characters, we use more complex and excellent feature extraction networks to extract more detailed features of Uyghur text, such as "Very Deep Convolutional Networks" (VGG) [2], "Convolutional Neural Networks With Gated Recurrent Connections" (GRCNN) [3], "Deep Residual Network" (ResNet) [4] and "ConvNet for the 2020s" (ConvNeXt) [5]. These methods are used to extract visual features of Uyghur images. For the textual characteristics of Uyghur with strong adhesion, we use the attention mechanism (Attn) [6] instead of CTC for prediction. In addition, we also remove the "bi-directional long short-term memory"(BiLSTM) [7] layer to reduce the overall parameters of the network and improve the performance of the network. Eventually, the trade-off was evaluated between the model parameters and the end accuracy to ultimately select the most suitable language model for Uyghur language recognition.

We used a variety of recognition structures for Uyghur recognition, including networks such as "Convolutional Recurrent Neural Network" (CRNN) [8], "Recursive CNN" (RCNN) [9], "Rosetta" [10], etc., which have performed well on public datasets, as well as recognition networks that have not been applied to Uyghur before, such as the combination of a GRCNN and the attention mechanism. We ultimately found that the text recognition model using the attention mechanism (Attn) overall outperforms the text recognition model using CTC decoding, which may be due to the stickiness of the Uyghur language. Lastly, we selected three optimal models for Uyghur text recognition because of their excellent results, which are (RCNN + BiLSTM + Attn), (ResNet + BiLSTM + Attn) and (ConvNeXt + BiLSTM + Attn). In experiments, (RCNN + BiLSTM + Attn), and (ResNet + BiLSTM + Attn) can achieve 86.19% and 89.32%, respectively, while balancing the number of parameters and time to maintain a good network performance. If the best accuracy is desired, the structure of (ConvNeXt + BiLSTM + Attn) can be used, because its recognition accuracy can reach 90.21%.

2. Related Work

As a classical text recognition network, CRNNs were introduced as early as 2015, while new network structures have emerged in recent years. Recognition models based on publicly available text recognition datasets have also made great progress up to now. Currently applied to other public languages, such as English and Chinese characters, many mature text recognition models and algorithms have been created based on these languages due to their huge number of speakers and researchers, which has profound implications for Uyghur recognition. Both Beak et al. [11] and Diaz et al. [12] adopted the idea of phased text recognition using different network structures as replacements to train trade-offs for optimal models. Shi et al. [13] implemented a unified combination of correction network and recognition network training for irregular images. Xie et al. [14] changed the loss function to the ACE loss function instead of CTC, which also obtained good results.

He et al. [15] studied Chinese character recognition with respect to the characteristics of the Chinese language.

In addition, in 2016, Lee et al. [9] fully exploited the use of a recursive convolutional layer (RCL) to improve the feature extraction network, and later Wang et al. [3] improved the recurrent convolutional layer based on this by adding gating units to construct a GRCNN. Nowadays, most of the Recurrent Neural Networks that are part of the recognition network still use the classical BiLSTM network; Rosetta et al. [10] proposed to remove the Recurrent Neural Network part directly to solve this problem and performed the final prediction directly with visual feature sequences, still obtaining terrific results. In the final sequence prediction stage, Shi et al. [16] developed the use of an attention mechanism to replace the use of CTC in 2017 to predict the last value, i.e., the RARE model. In the same year, Cheng et al. [17] introduced the method of text recognition by combining ResNet with an attention mechanism. In 2023, Liu et al. [18] proposed the use of real-time scene text detection with differentiable binarization (DBNet) [19], a combination of a text orientation classifier and the Retinex algorithm for image enhancement, to improve the accuracy of text recognition in complex scene images.

In a similar Uyghur sticky script, ASGHAR et al. [20] used different text recognition structures (CRNN with ResNet) as a feature extraction network versus VGG as a feature extraction network in their study of the Urdu language and confirmed that CRNN text recognition structures using residual links have better recognition accuracy. In 2023, Bhatti et al. [21] used a Convolutional Neural Network for classification and recognition of Urdu language (and its variants) datasets and developed handwritten Urdu language digit and Pakistani currency digit datasets. Faizullah et al. [22] presented a review of Arabic recognition, where they presented the entire process of optical character recognition (OCR) for Arabic, including some of the most advanced techniques, and provided future research directions for Arabic recognition. After this, Najam et al. [23] provided the latest deep learning techniques for OCR technology and correction after OCR for Arabic handwriting, developing a practical space for future trends in Arabic OCR applications.

There have been some excellent deep learning algorithms proposed in the research of Uyghur text recognition. In 2017, Wang [24] of Xi'an University of Electronic Science and Technology in China conducted research on the application of key technologies for printed Uyghur text recognition, obtaining several contiguous segments and single standing characters by the morphological expansion method for line slicing. After this, the contiguous segments were sliced into characters by the vertical projection method. In 2020, Chen [25] of Chengdu University of Technology in China studied and designed Uyghur text detection and recognition based on deep learning, in which a text recognition model combining CRNN with CTC was exploited in the paper, improving the output of CTC for Uyghur characters with right-to-left writing characteristics and designing the character arrangement rules in accordance with the Uyghur writing order. Moreover, the accuracy of Uyghur recognition was 64%. In 2021, Tang [26] from Xinjiang University established the network structure of ResNet+BiLSTM+Attn to recognize a scanned dataset of Uyghur text and achieved better recognition results. In 2021, Xiong [27] from Xinjiang University also researched and applied the method of detecting and recognizing printed Uyghur text and recognized Uyghur text based on the Django system, establishing a network model of GRCNN + BiLSTM + Attn, slightly improving the test accuracy. However, regarding how to carry out standardized Uyghur recognition, including handwriting and print recognition of Uyghur, as well as how to use more and better text recognition models in the recognition of Uyghur text, there is no clear answer. We compared the above methods with ours during the following experiments and showed the superiority of our method.

3. Structure of Recognition

In this paper, we use a three-stage Uyghur recognition structure, as shown in Figure 1, which divides Uyghur into three different successive operation stages: the feature extraction stage (Feat.), the sequence modeling stage (Seq.), and the prediction stage (Pred.). The

model can not only be applied to the current existing methods, but also can improve models by using some possible combinations.

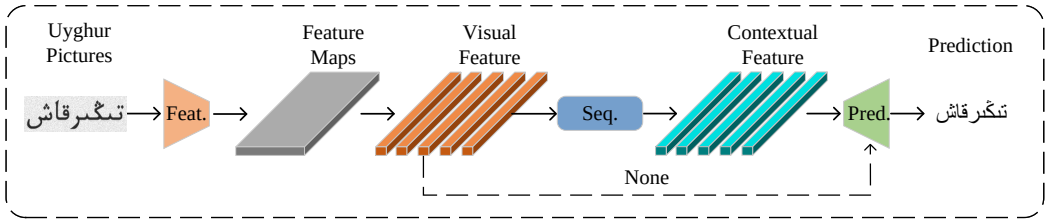

Figure 1. Three-stage Uyghur identification structure.

3.1. Feature Extraction Stage

In this study, we used four different Convolutional Neural Network structures, VGG, RCNN, ResNet, and ConvNeXt, to extract network features. The VGG network is a Convolutional Neural Network consisting of multiple convolutional layers and several fully connected layers. In the classical text recognition algorithm CRNN, the VGG network is the network structure chosen for the CNN phase. Adjustments were made for better suitability to linguistic texts. A rectangular window of 1 × 2 size is used in the third and fourth maxpooling layers to generate feature maps with larger widths instead of the traditional square maxpooling window. As shown in Figure 2, after inputting a normalized 100 × 32 Uyghur image, the VGG network generates a feature map with a smaller width and height and more channels and finally outputs a visual feature map with 512 channels (24 width and 1 height), which is then input into the sequence model for the next stage of training.

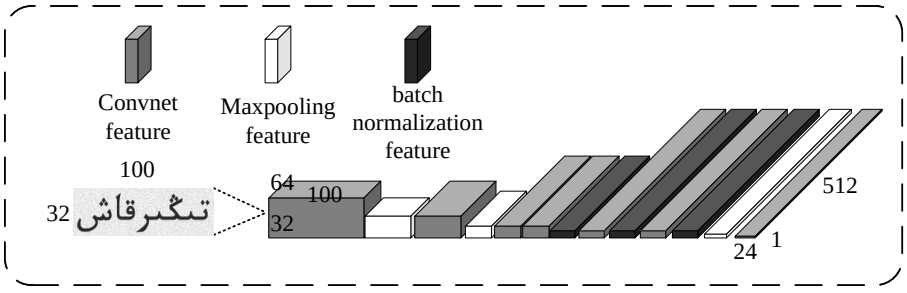

Figure 2. VGG feature extraction network.

The RCNN is an improved feature extraction network using recurrent convolutional layers (RCLs). With the same number of parameters, RCLs increase the depth of the CNN and produce a more compact feature response than the CNN. In order to be more suitable for text recognition and take into account different perceptual fields, Wang et al. [3] further improved the RCNN by adding a gate to the recurrent convolutional layer (RCL), which is a key component of the RCNN, so as to construct a gated recurrent convolutional layer (GRCL) to balance the feedforward information and recurrent information and to finally constitute the GRCNN network. The gating unit is computed as (1):

$$G(t) = \begin{cases} 0 & t = 0 \\ \sigma_{sig}\left(BN\left(w_g^f(u(t))\right) + BN\left(w_g^r(x(t-1))\right)\right) & t > 0 \end{cases} \quad (1)$$

where t represents the time scale and w_g^f and w_g^r denote the feedforward weights and the cyclic weights of the gate for two 1 × 1 kernels, which are convolved with the feedforward input and the cyclic convolution input, respectively. Then, after batch normalization and the sigmoid function, $G(t)$ is generated, which indicates the strength of information inflow in the hidden state. "$x(t)$" is then calculated from (2), where w^f and w^r also use a 1 × 1 convolution kernel to calculate the weights, and finally $x(t)$ is calculated by batch

normalization and the ReLU function. When $G(t)$ is 1, all the information from the previous t-1 steps is retained. When $G(t)$ is 0, the information from the previous hidden state is discarded and only the input $u(t)$ at moment t is considered.

$$x(t) = \begin{cases} \sigma_{\mathrm{ReLu}}(BN(w^f * u(t))) & t = 0 \\ \sigma_{\mathrm{ReLu}}(BN(w^f * u(t)) + BN(BN(w^r * x(t-1)) \odot G(t))) & t > 0 \end{cases} \quad (2)$$

In the overall feature extraction network, the GRCL layer is used to replace the convolutional layers, replacing the second, third, fourth and fifth convolutional layers with three GRCL layers with five iterations, of which the convolutional kernel size is 3 × 3. The final feature extraction network outputs a feature map with a channel number, height and width of 512 × 1 × 26. The overall network feature map is shown in Figure 3.

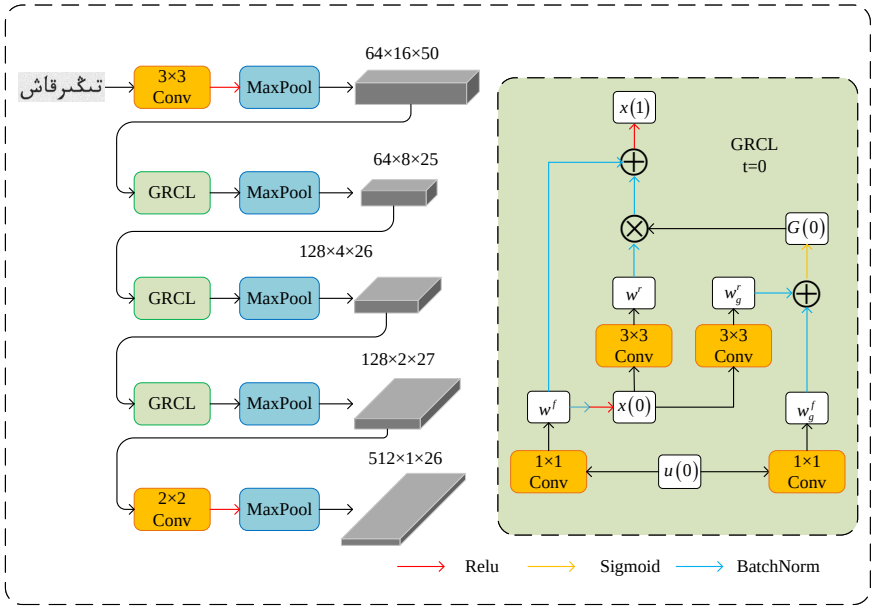

Figure 3. GRCNN feature extraction network.

ResNet, a CNN network with residual connections, moderates two problems of network gradient explosion and recognition accuracy saturation which lead to a rapid decrease in accuracy, learning the difference between the target and input values instead of the complete output so as to make deeper recognition network training possible. We used ResNet to improve the feature extraction network part of our three-stage Uyghur recognition structure and used an attention mechanism in the subsequent decoding prediction stage.

ResNet network blocks were sued to construct ResNet network layers to replace the convolutional layers in the middle of the original VGG network, where a deeper ResNet network block is used, including two convolutional layers, two batch normalization layers and a ReLU activation function. The final output is connected to the input with residual connection.

Finally, we use ConvNeXt network as a feature extraction network. ConvNeXt is a pure Convolutional Neural Network, benchmarked against Swin transformer [28] and optimized on the basis of ResNeXt [29] after a series of experimental comparisons. The optimization strategies are as follows.

Firstly, a macro design is adopted in stage 3 of the convolution operation, a high proportion of block stacking times is used and the overall four-stage stacking times ratio is (3, 3, 9, 3). Stage 1 of the downsampling module uses a 4 × 4 size convolution kernel with a four-step convolution layer. Stages 2–4 of the downsampling layer use a convolutional layer with a 2 × 2 size convolutional kernel with a step of 2. In order to adapt to our long, striped dataset and to fit the subsequent network structure, we adjust the step size of the subsequent three layers here to (2 × 1) and use a padding of (0, 1). Secondly, we draw on

ResNeXt-ify with more aggressive depthwise convolution, i.e., group convolution with the number of groups equal to the number of channels. The initial number of channels is 96, the same as the Swin transformer. Thirdly, an inverted bottleneck was designed by first using deep convolution to extract network features and then using 1 × 1 point-by-point convolution to convert the number of channels from 96 to 384 and then from 384 back to 96. Then, the kernel sizes are increased, following Transformer's [30] example of moving the depthwise conv module up and using a larger convolutional kernel ((7 × 7) instead of the original (3 × 3) convolutional kernel). Lastly, regarding the micro design, a GELU [31] activation function is used instead of ReLU, as well as fewer activation functions and fewer normalization layers. Batch normalization is replaced with layer normalization. S separate downsampling layer is set up. The network module of ResNet is compared with the network module of ConvNext in Figure 4.

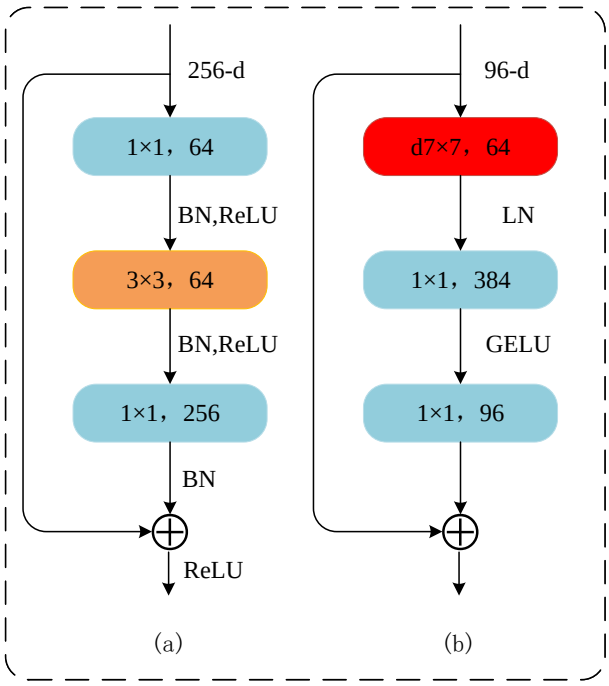

Figure 4. (**a**) ResNet and (**b**) ConvNeXt feature extraction network block.

We finally chose ConvNeXt-Base as our feature extraction network, with the number of channels as C = (128, 256, 512, 1024) and the number of block stacks as B = (3, 3, 27, 3). To fit our recognition network, we removed the final linear layer and changed the final output channel number from 1024 to 512.

3.2. Sequence Modeling Stage

The sequence modeling phase mainly uses the deep bi-directional LSTM network, which currently performs better in text recognition, to capture the contextual information of the sequences. In addition, to reduce the computational complexity and memory consumption, Rosetta removed the BiLSTM layer, and we will compare the impact of the BiLSTM layer in the subsequent experimental section.

As shown in Figure 5, the network is designed as a two-layer, bi-directional LSTM network, and the visual feature map in the feature extraction network stage will be input to the network as X. The hidden state and parameters are added to the computation to obtain the output h. After passing through the deep BiLSTM layer and then a linear layer, the final output is a text feature map with 256 channels, with a width of 24 and a height of 1. After this, it is input to the prediction stage for processing. If the LSTM layer is not used in the sequence modeling stage, the visual feature maps are directly input as text feature maps to the prediction stage for processing.

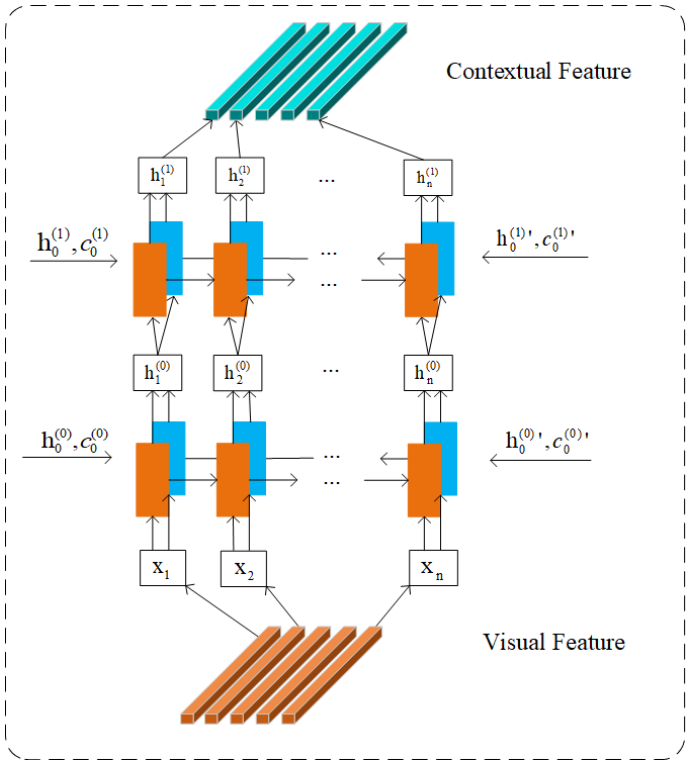

Figure 5. Deep bidirectional LSTM network.

3.3. Model Prediction Stage

The prediction phase of the model uses two methods: CTC and an attention mechanism.

Connectionist temporal classification (CTC) is used to solve the classification problem of temporal class data. It allows the RNN to learn sequence data directly without prior labeling of the mapping relationship between input and output sequences in the training data, which breaks the data dependence constraint of RNN applications in speech recognition, handwriting recognition, and other fields so as to make RNN models achieve better application results in sequence learning tasks. The key approach of CTC is to predict a character in each column ($y_i \in$ y) and modify the complete character sequence into a stream of unfastened characters by removing duplicate characters and spaces, as shown in Figure 6.

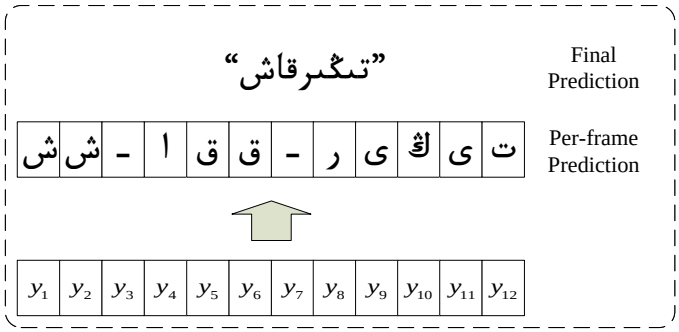

Figure 6. Connectionist temporal classification (CTC).

The loss function of CTC is implemented using forward–backward propagation and is defined as (3):

$$L_{CTC} = -\sum_{I_i, \mathbf{l}_i \in \mathcal{X}} \log p(\mathbf{l}_i \mid \mathbf{y}_i) \tag{3}$$

where $\mathcal{X}$ is the training dataset, I_i is the trained image, $\mathbf{l_i}$ is the real label, and $\mathbf{y_i}$ is the output sequence after the CNN and LSTM layers.

The attention mechanism originates from the study of human vision. In cognitive science, due to bottlenecks in information processing, humans selectively attend to a portion of all information while ignoring the rest of the visible information. The attention mechanism has also been used in recent years for prediction of text recognition, automatically capturing the flow of information in the input sequence to predict the output sequence. As shown in Figure 7, using the attention mechanism in the prediction stage requires transforming the text feature map into a one-hot encoding to generate the input h. The correlation between the query vector q and the input h is determined by a score function, and h is formed by the decoder's LSTM on a time-step-by-time-step basis. The score function we use here is the additive model (4).

$$s(h, q) = v^T \tanh(Ws_t - 1 + Uh_t) \tag{4}$$

where v, W, and U are the trainable parameters, s_t is the decoder LSTM hidden state at time t, and h_t is the sequence input at time t.

After performing softmax calculation to generate the attention distribution vector a and matrix multiplication and a splicing operation on a to generate the text feature sequence, the decoder's LSTM network maps the predicted values of Uyghur text.

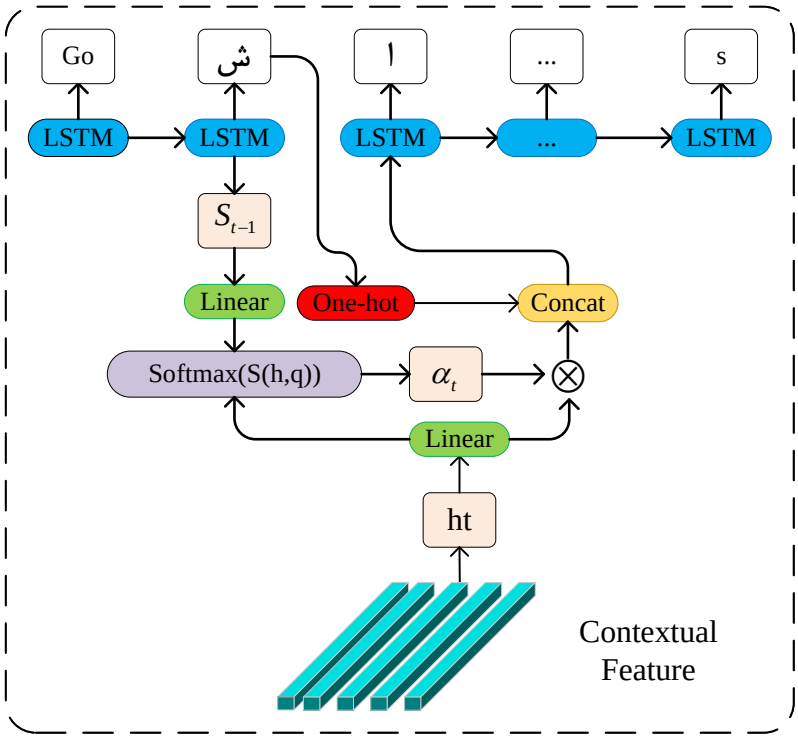

Figure 7. Attention mechanism (Attn).

Thus, the loss function is defined as (5):

$$L_{Att} = -\sum_t \ln P(\hat{y}_t | \mathrm{I}, \theta) \tag{5}$$

where I is the picture of training, $\hat{y}_t$ is the true value of the feature at time moment t, and θ is the vector combining all network parameters.

4. Datasets

4.1. Printed Dataset Generation

For the study of text recognition in English, Chinese, and other public languages, relatively standard and publicly available text recognition corpora and datasets are available worldwide, so the performance of the analysis methods can be compared by using different recognition models on public datasets. However, there is no standard and open text recognition dataset available for the Uyghur language.

In this paper, we collected 16,555 Uyghur words through Uyghur websites such as People's Daily Online Uyghur (http://uyghur.people.com.cn/ accessed on 4 June 2021), China Uyghur Radio Network (http://www.uycnr.com/lwxj/ accessed on 11 June 2021), and Aksu News (http://uy.aksxw.com/ accessed on 19 June 2021); the obtained Uyghur text was used as labels and the data were cleaned, sorted, and divided into words to generate Uyghur images according to Uyghur words. The training set has five w images, including one w original images with a white background, one w image with different degrees of Gaussian blurring, one w image with random waves and one w image with different backgrounds; 5 k images with a Gaussian noise background; and 5 k images with a quasi-crystal background, and each part contains different Uyghur fonts. The validation set and test set were constructed with one-third of the training set. The specific data enhancement method is shown in Figure 8.

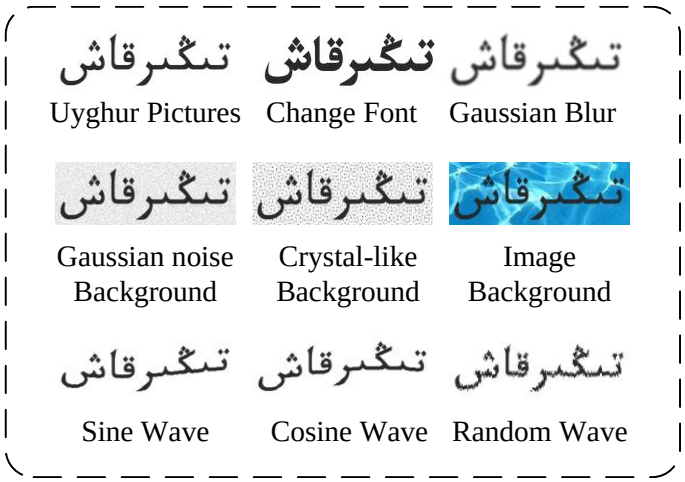

Figure 8. Data enhancement of Uyghur images.

4.2. Handwritten Dataset Generation

We selected a total of 270 words, each with 140 different writing samples, according to the nouns from "5000 Words Commonly Used in Uyghur" [32], written by more than 70 students according to their individual writing styles and different handwriting forms of Uyghur. In this paper, a combination of manual and computer methods was used to pre-process, crop, categorize, and filter the raw data, and finally a standardized, high-quality, and diverse handwritten Uyghur word picture dataset was obtained. The process of constructing the handwritten dataset in this paper is as follows:

1. A 20 × 5 blank grid paper template was designed and printed, where each blank paper can contain 100 written Uyghur words. The grid paper template and the sample Uyghur words were distributed to the students and they were required to write in order according to the sample. The handwritten sample is shown in Figure 9a, due to the condition, some students used the hand-drawn grid paper template to complete the writing as in Figure 9b, some students used the electronic template, and some students used electronic templates and capacitive pens to complete handwriting, as shown in Figure 9c.

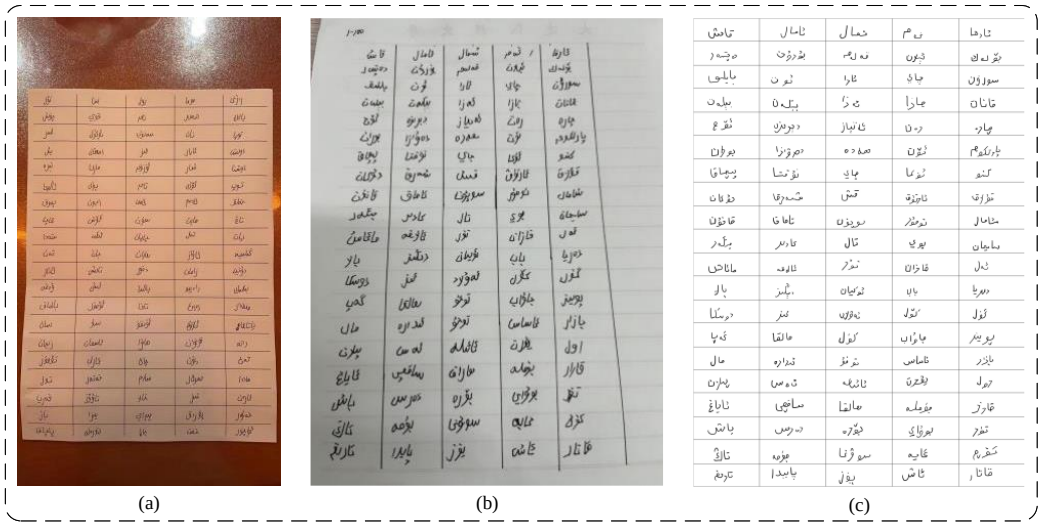

Figure 9. Example of a computer-cut Uyghur words. (**a**) Sample print template data. (**b**) Sample hand-drawn template data. (**c**) Sample electronic template data.

2. The completed paper images were scanned or photographed, and the electronic images were filed in a single folder. One dataset contains three pictures, and the pictures in each folder were named a.jpg, b.jpg, and c.jpg. A total of 140 files were organized and stored in the same directory, and the 140 files were named according to E1–E140.
3. Due to the differences in shooting or scanning angles, the picture was distorted and the horizontal line (vertical line) in the original template was no longer horizontal (vertical). In order to facilitate the computer to crop each Uyghur word more accurately in the later stages, it was necessary to adjust the angle of the picture so that the original horizontal line (vertical line) in the picture is as horizontal (vertical) as possible and to crop the edge part which is not related to the data. The electronic sample did not need to be cropped. The manually adjusted and cropped data sample is shown in Figure 10.
4. Due to errors or omissions in manual writing, there exist data samples written by some students with misordered or missing words. In order to facilitate batch cropping by the computer as well as labeled order classification, the archived data needed to be manually screened and corrected before computer cropping was performed.
5. Regarding computer cropping, morphological operations were used to identify the horizontal and vertical lines in the picture, and they were matched to obtain intersection coordinates. Then, the top and bottom boundaries were divided according to the distance between the coordinates, and each Uyghur word in the picture was cropped from left to right and from top to bottom according to the boundary coordinates. Automatic computer cropping was performed in the order a.jpg, b.jpg, and c.jpg in each folder, traversing E1–E140, with 270 Uyghur word pictures cropped out of each folder. All the data in E1–E140 were computer cropped to obtain a total of 140 folder directories named S1–S140. The images under each file were named in the format x.jpg, with x taking values from 1 to 270 (e.g., if x is 112, it corresponds to the second word from left to right in the third line of the b.jpg image). The corrected cropped image is shown in Figure 11.
6. For computer categorization, the same words corresponding to the same serial numbers in the S1–S140 folder directories were evaluated and categorized. The final result was 270 folders, each file containing 140 samples of the same words but with different handwriting styles. The file names and the names of the pictures in each file were automatically labeled by the computer. A total of 270 folders were named T1–T270, and the pictures in each file were named in the format of M(N).jpg (e.g., 243(7).jpg means the seventh word in file T243). The detailed semantics and correspondence of each word in the language are shown in Appendix A.

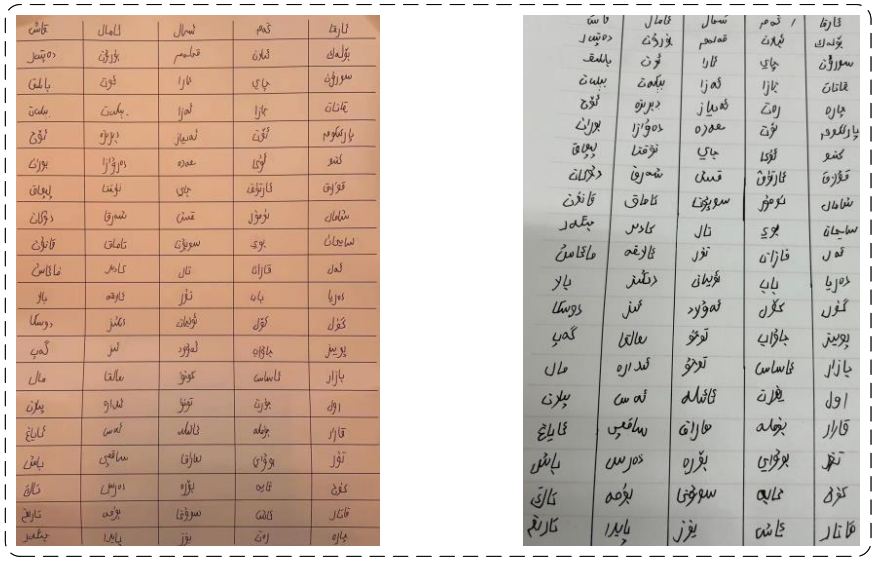

Figure 10. Example of manual adjustment data.

Figure 11. Example of a computer-cut Uyghur words.

7. In order to reduce the influence of the background on the Uyghur words in the images, the blank part of the edge of each Uyghur word was cropped. The categorized and cropped data are shown in Figure 12.

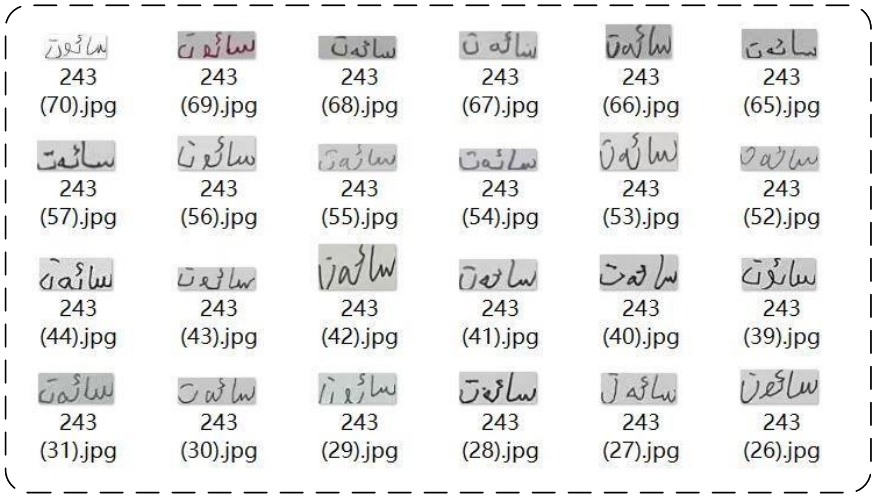

Figure 12. Sample data after trimming the edges.

8. Due to the random nature of the handwritten data, there were some misspelled words, and two Uyghur students were invited to screen out this part of misspelled words and remove them. A total of 20,250 handwritten Uyghur pictures including 250 words were finally used to construct the dataset.

4.3. Handwritten Dataset Pre-Processing

For deep network models, a standardized data input is crucial. Therefore, for the handwritten Uyghur word dataset established in this paper, the following data pre-processing was carried out, where the process of data pre-processing is demonstrated in Figure 13.

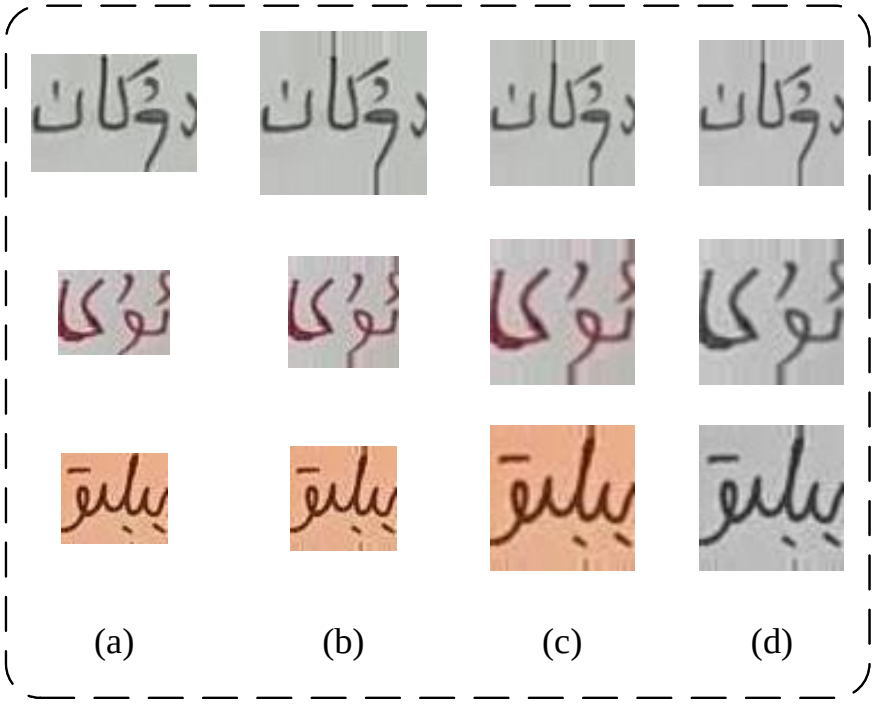

Figure 13. Data pre-processing steps. (**a**) Original Picture. (**b**) Padding. (**c**) Uniform Scale. (**d**) Grayscale.

1. Filling the images as square: Filling the input data as square will ensure the handwritten Uyghur words are trained in a normal glyph shape without distortion due to resizing during network input. Therefore, in this paper, all images were filled as squares according to the background color, and an equal amount of filling was

performed on both sides (length or width) according to the dimension which is smaller. The effect after filling is shown in Figure 13b.

2. Uniform scale: most of the dataset images established in this paper are in the size range of 50×50 to 80×80; in order to ensure the uniformity of all the data and to improve the convergence speed and accuracy of the model, all the image sizes were standardized to 64×64 size, as shown in Figure 13c.
3. Grayscale image: The handwritten Uyghur language data collected in this paper have a mostly gray background and the image background noise is large. The use of binarization will amplify the effect of noise on the data, affecting the model when conducting recognition; thus, in this paper, we used grayscale for normalized processing, as shown in Figure 13d.

4.4. Handwritten Dataset Augmentation

Data augmentation is the process of increasing the amount of data available by making valid data and improving the diversity of data samples. This helps to alleviate the overfitting of the network and improves the generalization ability of the network.

Using basic data enhancement methods can have a great impact on the model. Therefore, in this paper, several methods in the geometric transformation class as well as the color transformation class were selected for data enhancement in the handwritten text recognition task, as shown in Figure 14a, which is the original picture.

1. Stochastic affine transformation: The stochastic affine transformation implements translation, rotation, scaling, and shearing transformations on the image, and is more random and realistic compared to single data enhancement methods. Performing random affine transformation on handwritten Uyghur data can simulate the effect of text deformation that occurs in real writing, as shown in Figure 14b.
2. Gaussian noise: In image processing, Gaussian filters can be used to suppress high-frequency noise features in an image to improve image quality. The background of handwritten Uyghur text images is cluttered with a lot of noise and interference points, and Gaussian noise can mitigate such effects, as shown in Figure 14c.
3. Elastic transformation: Random perturbation of pixel points is realized by applying a small random translation to the position of each pixel point in the image. The distortion of characters in text recognition can simulate the pictures produced in the context of handwritten text due to muscle shaking, lighting changes, etc. In the handwritten digit recognition experiments, the recognition effect of handwritten digits was significantly improved after the original picture was augmented. Therefore, for the recognition of handwritten Uyghur language in this paper, this method was also chosen for data augmentation of the dataset, and the effect is shown in Figure 14d.

In this paper, the pre-processed data were augmented with a mixture of techniques using several methods mentioned above. Finally, the handwritten dataset was merged with the printed dataset for text recognition experiments. To ensure a fast input–output performance and improve the training speed of the network, the Lightning Memory-Mapped Database (LMDB) was used to store the dataset. It is encoded in the 'utf-8' encoding format for storage. When reading the dataset, the images are transformed into a fixed size of width 100 and height 32 and then randomly scrambled into the network for training. When using the CTC loss function, the dictionary is constructed by adding one character [CTCblank], and the final dictionary length is 33. When using the attention mechanism, the dictionary is constructed by adding two characters [GO] and [s], and the final dictionary length is 34.

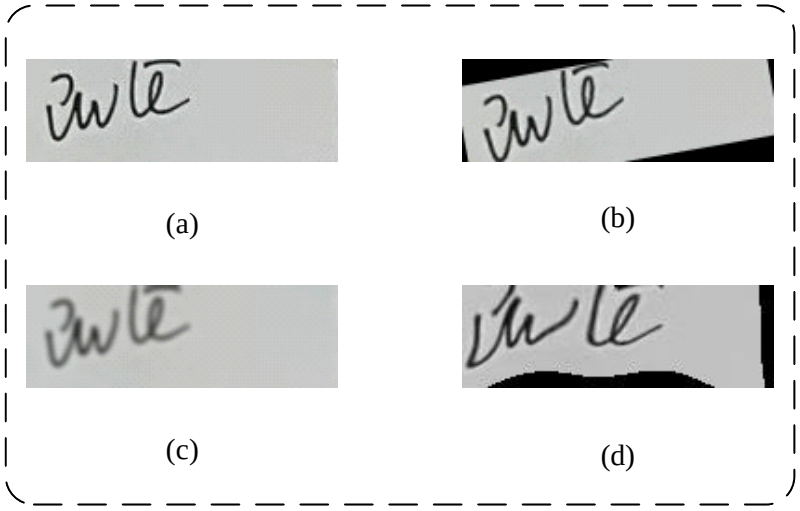

Figure 14. Figure showing the effect of data augmentation via different methods. (**a**) Original picture. (**b**) Stochastic affine transformation. (**c**) Gaussian noise. (**d**) Elastic transformation.

5. Results and Discussion

5.1. Evaluation Standards

We use three metrics, accuracy (ACC), normalized edit distance (Norm_ED), and model testing time (Time), as evaluation criteria for the recognition networks. Accuracy is used to measure the effectiveness of recognition, and is defined in (6).

$$ACC = \frac{N_{correctwords}}{N_{allwords}} \times 100\% \tag{6}$$

where $N_{correctwords}$ represents the total number of words recognized correctly and $N_{allwords}$ represents the total number of words recognized.

The normalized edit distances were summarized and divided by the number of test images. The obtained average edit distance was used as a metric, defined in (7)

$$Norm\_ED = 1 - \frac{1}{N}\sum_{i=1}^{N}\frac{D(s_i, \widehat{s_i})}{\max(s_i, \widehat{s_i})} \tag{7}$$

where D denotes the Levenshtein distance, s_i denotes the predicted text line, $\hat{s_i}$ denotes the corresponding ground truth, and N is the total number of text lines.

The experiments in this paper measure the performance of the network by the recognition accuracy, the model testing time, the normalized edit distances, and the number of parameters of the network.

5.2. Description of Experimental Environment and Parameters

The experiment was based on the PyTorch framework and ran on a NVIDIA GeForce RTX 3070Ti GPU. The training batch was set to 256 and the number of epochs was 10,000. The adaptive learning rate was determined using Adadelta optimizer [33], the initial learning rate lr was 1.0, the decay value β was 0.95, and the parameter was $1e^{-8}$. Gradient cropping was used in the fifth magnitude. Parameter initialization of the model was performed before training.

5.3. Model Interpretability

In this section, we present Saliency Maps [34] to present an interpretation of our model. A Saliency Map is an interpretive tool for visualizing deep learning models. It measures the sensitivity of the model's output to the input by calculating the gradient of the input image. It then maps this information onto the input image to produce an image of the same dimensions as the input image, where the value of each pixel indicates the effect of that

pixel on the model predictions. We analyzed the prominence of the image and how much influence each character has on the model prediction by drawing Saliency Maps.

In Figure 15, we can find that each pixel in the Saliency Map drawn by calculating the degree of influence of the whole word on the prediction indicates the degree of influence on the prediction. In this map, pixels that may be true values and pixels that have a positive effect on the prediction are indicated by brightly colored pixels, especially at the inflection points of the text in the image. The upper and lower points of the characters in the text are highlighted, and on the whole, it can be seen that the pixels that have a positive effect on the prediction of the model make up images that are basically the same as the textual values.

Figure 15. Saliency Map of the whole word for predicting the degree of influence.

As shown in Figure 16, through the Saliency Maps of the influence of each character on the prediction, we found that a single character making a prediction will appear as a brighter pixel at the position where it is supposed to be in the picture, and darker in other places. Since Uyghur is written from right to left, the order of the predicted characters corresponds to the pictures getting brighter from right to left.

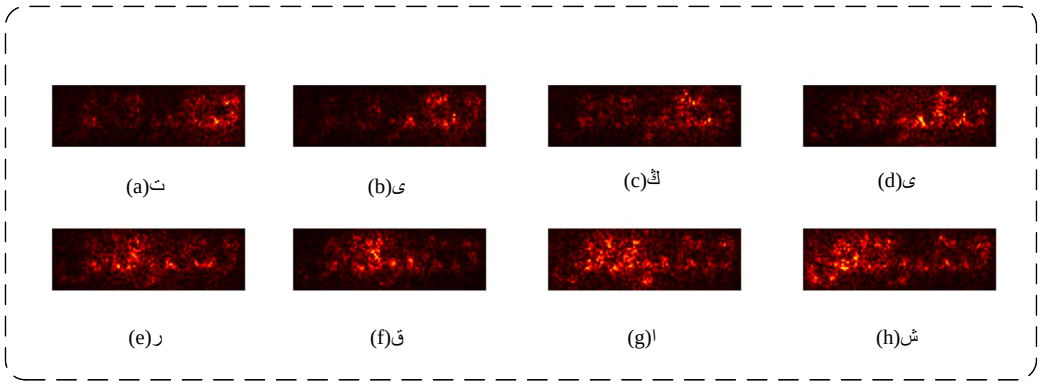

Figure 16. Saliency Maps for each character for predicting the degree of influence.

In order to compare how different data points are correctly predicted from one another, we used LIMEs (Local Interpretable Model-Agnostic Explanations) [35] to effectively explain how the model works by selectively adding noise to certain parts of the image. We have selected LIME visualization images of some of the characters that represent the predicted meaning of the word indecision. This is shown in Figure 17. We find that the model focuses its attention between the character itself and its neighboring characters, which effectively explains that Uyghur is adhesive, and that to correctly recognize a character it is also necessary to focus on its neighboring characters.

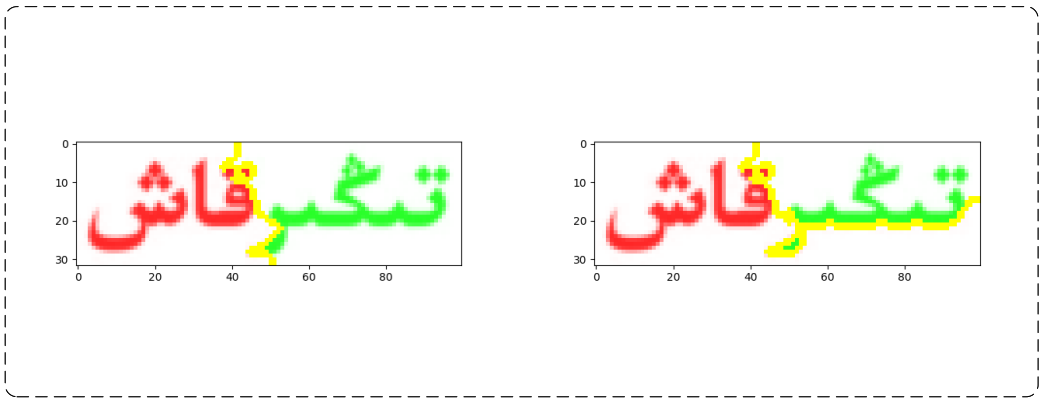

Figure 17. Local Interpretable Model-Agnostic Explanations.

We also used the SHAP (Shapley additive explanation) [36] algorithm, which has performed extremely well in classification models, to conduct an interpretable analysis of the recognition model; not only does the SHAP model help to explain the contribution of certain pixels to the model output, but by visualizing the SHAP values of different images, it is also possible to identify patterns and differences in the model's decision making across different inputs. We analyzed the significance of all the characters of the word "تىگىرقاش" and plotted an SHAP heat map. This is shown in Figure 18. We show the prediction results for the top two predicted values separately, and we find that for each character, the results predicted by the first pair of plots are correct, while the results of the second plot are all wrong, but the characters represented by the second plot are morphologically extremely close to the correct results; thus, we believe that our model has a very reliable ability to recognize the correct character through minute details.

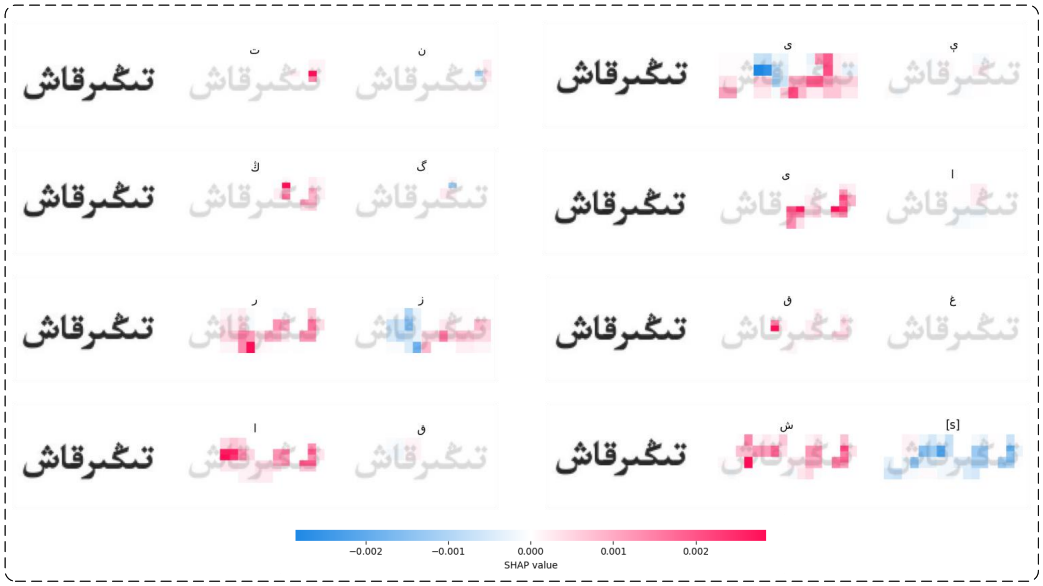

Figure 18. Shapley additive explanation.

Afterwards, we plotted the attention feature maps for each layer of the network, again for the sake of model interpretability. We did this by extracting and plotting all the output feature maps from the convolutional and linear layers of the entire network, and some of the feature maps are shown in Figure 19. We can find that the networks in the bottom layers, such as layers 1, 2, and 3, are more concerned with the basic features of the image such as texture and shape, while the networks in the higher layers, such as layers 12, 14, and 23, are more concerned with the abstract information of the image.

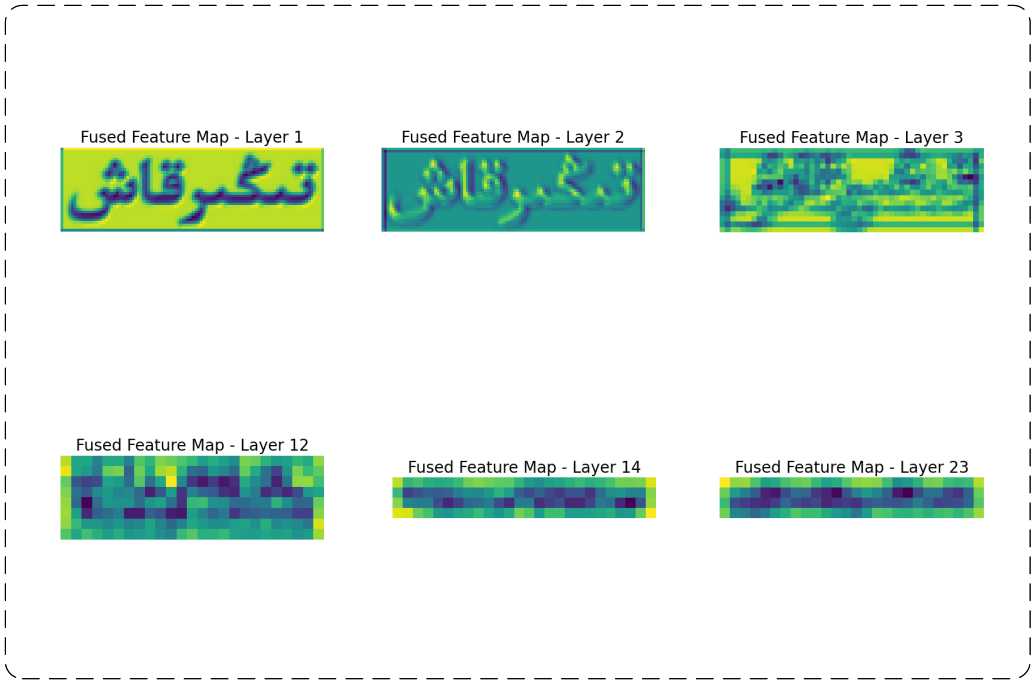

Figure 19. Feature maps for partial networks.

5.4. Classification Experiments

In this paper, a three-stage text recognition structure was used for Uyghur recognition experiments, developing different combinations of recognition models ($4 \times 2 \times 2 = 16$ kinds). The accuracy, the model testing time, the normalized edit distance, and the overall number of parameters were compared under different network structures. The comparison results are shown in Table 1.

Table 1. Recognition accuracy, normalized edit distances, and number of parameters of the network.

#	Feat.	Seq.	Pred.	ACC.%	Time ms/Image	Norm_ED	Params $\times 10^6$
P 1	VGG	None	CTC	29.62	0.996	0.55	5.57
P 2	VGG	None	Attn	80.36	6.977	0.93	6.58
P 3 [1]	VGG	BiLSTM	CTC	69.03	1.993	0.89	8.45
P 4	VGG	BiLSTM	Attn	84.70	7.973	0.95	9.14
P 5	RCNN	None	CTC	28.55	4.983	0.55	1.88
P 6 [2]	RCNN	None	Attn	83.82	10.964	0.94	2.89
P 7 [3]	RCNN	BiLSTM	CTC	69.15	5.980	0.90	4.76
P 8	RCNN	BiLSTM	Attn	86.19	12.956	0.95	5.46
P 9 [4]	ResNet	None	CTC	53.41	2.990	0.78	44.28
P 10	ResNet	None	Attn	85.80	9.968	0.95	45.29
P 11	ResNet	BiLSTM	CTC	78.30	3.986	0.93	47.16
P 12 [5]	ResNet	BiLSTM	Attn	89.32	11.960	0.96	47.86
P 13	ConvNeXt	None	CTC	56.81	5.979	0.83	67.57
P 14	ConvNeXt	None	Attn	86.27	11.960	0.95	68.58
P 15	ConvNeXt	BiLSTM	CTC	79.24	6.976	0.93	70.45
P 16	ConvNeXt	BiLSTM	Attn	90.21	14.951	0.97	71.14

[1] CRNN [2] R2AM [3] GRCNN [4] Rosetta [5] RBA.

According to Table 1, we see the trade-off between accuracy and the number of parameters for all 16 different module combinations. The comparative results are shown more clearly in Figure 20. In Figure 20, we show the results of current text recognition models applied to Uyghur text recognition. Among them, R2AM performs better, with accuracy rates of 83.82%; on the other hand, Rosetta uses ResNet for feature extraction but removes the BiLSTM layer, which leads it to perform poorly in the recognition network using CTC for prediction, with only 53.41%. In addition, the networks using VGG or RCNN for feature extraction, also with the BiLSTM layer removed in the sequence modeling phase, perform the worst in the network using CTC for prediction, with a recognition accuracy of only 29.62% or 28.55%, respectively. Therefore, we believe that in Uyghur recognition, if CTC is used for prediction, it is better to use the BiLSTM layer for sequence modeling. If the visual features output from the feature extraction layer are only directly decoded using CTC, the results will be unsatisfactory, which may be due to the fact that Uyghur is sticky, i.e., the preceding and following characters in a word are stuck together, thus requiring more textual contextual information.

Compared to previous prominent approaches in Uyghur text recognition, which were primarily applied to Uyghur printed or scanned texts using models like CRNN, GRCNN, and RBA, we have achieved substantial progress in the realm of Uyghur script recognition. Employing the same three models on a dataset comprising a mixture of printed and handwritten Uyghur texts, we achieved accuracy rates of 69.03%, 86.19%, and 89.32%, respectively. However, our most advanced model, ConvNeXt+BiLSTM+Attn, demonstrated a remarkable accuracy rate of 90.21%. This achievement underscores the superiority of our approach. This accomplishment not only serves as a definitive benchmark but also underscores the distinctive value and effectiveness of our method in handling Uyghur text recognition.

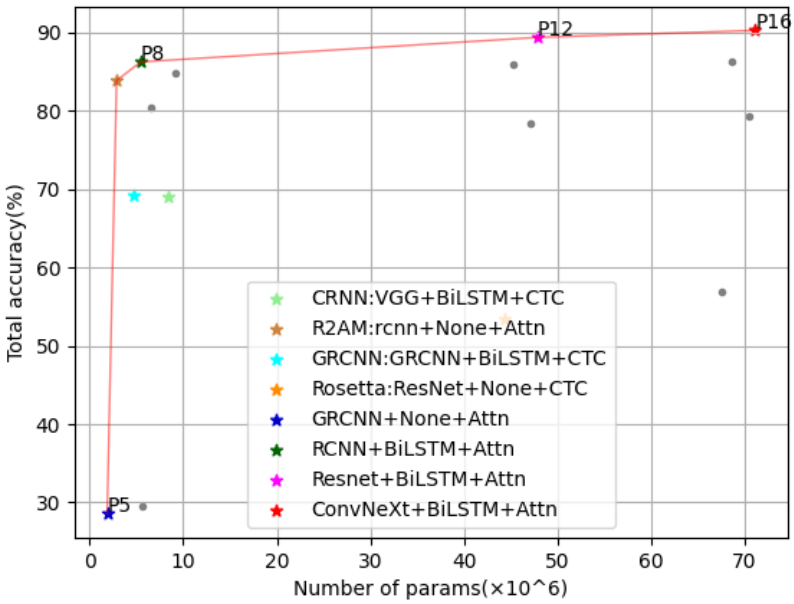

Figure 20. Accuracy—parametric chart.

In the trade-off between accuracy and time, we find that the overall trend in test time is similar to that of accuracy and the number of parameters, with the difference in the test time using ResNet as a feature extraction network is better than that of the CRNN network, even though the number of parameters in Resnet is higher. The best points overall are P4 (VGG + BiLSTM + Attn) and P12 (ResNet + BiLSTM + Attn), with accuracies of 84.70% and 89.32%, respectively, achieving a relatively high accuracy in a shorter test time. The optimal point of accuracy is P16 (ConvNeXt + BiLSTM + Attn) with an accuracy of 90.21%, as shown in Figure 21.

Corresponding to CTC is the use of the attention mechanism for prediction; as shown in Figure 22, we can clearly observe regardless of the feature extraction network and regardless of whether BiLSTM is used for sequence model modeling, excellent recognition results can be achieved, all reaching more than 80%. In particular, the three best performing points are P8, P12 and P16. The P8 network structure can achieve a better accuracy with a lower number of parameters using RCNN and BiLSTM with Attn. The second best is the P12 network structure, combining ResNet and BiLSTM with Attn, which has an increased number of network parameters and a slightly improved recognition accuracy compared to the P12 network. Finally, if the P16 network structure combining ConvNeXt and BiLSTM with Attn is used without taking into account the number of training parameters of the network, the optimal accuracy of 90.21% can be achieved. Thus, why does the use of the attention mechanism improve the accuracy of Uyghur recognition so much? After our analysis, it was determined that the memory of the network structure combining BiLSTM with CTC is not high and can only alleviate part of the long- and short-term memory problems, while the use of the attention mechanism enabled focusing on the global situation and can better deal with the semantic information of the text. It has an important role in the recognition process of Uyghur because Uyghur words are generally long and stick together, which requires a high level of training of semantic information. In addition, during training, we found that the network structure using the attention mechanism tends to converge much faster than the network structure using CTC prediction, so we believe that the use of the attention mechanism is more effective in Uyghur recognition.

In terms of model testing time, the inference time of models using the attention mechanism is generally longer than that of models using CTC. However, this is accompanied by an increase in accuracy, as shown in Figure 23.

The effect of the network feature extraction layer on the overall recognition network is shown in Figure 24. In terms of parameters, the RCNN has the least number of parameters, followed by VGG, then ResNet and finally ConvNeXt. Overall, the accuracy of recognition tends to increase as the parameters of the network increase, especially in the model structure using CTC. In the case of the model structure using the attention mechanism, the trend is slower because the recognition accuracy has reached a high level; however, in general, the recognition accuracy increases with the complexity of the feature extraction network.

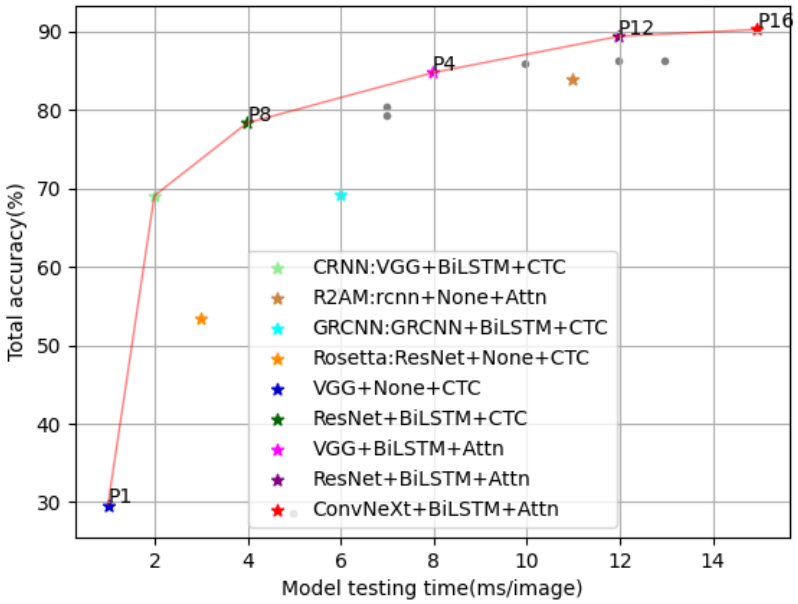

Figure 21. Accuracy—time chart.

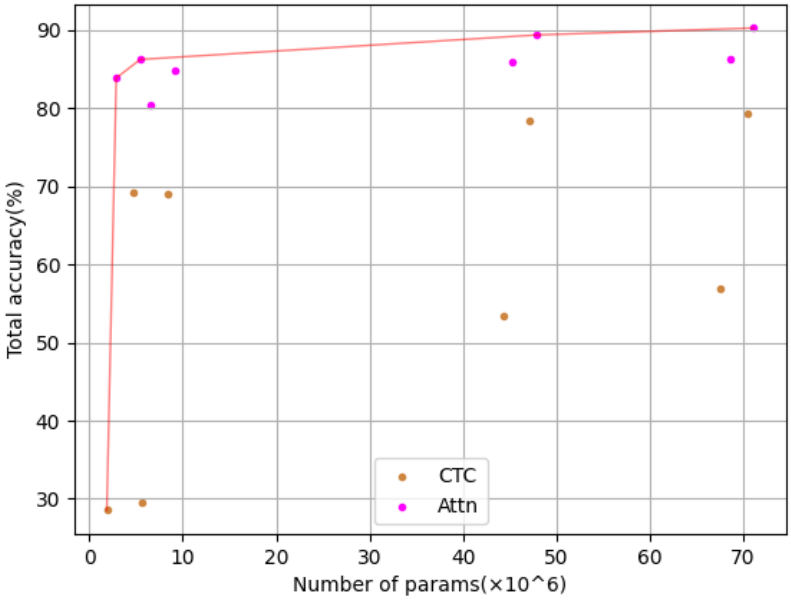

Figure 22. CTC versus Attn. (Number of parameters.)

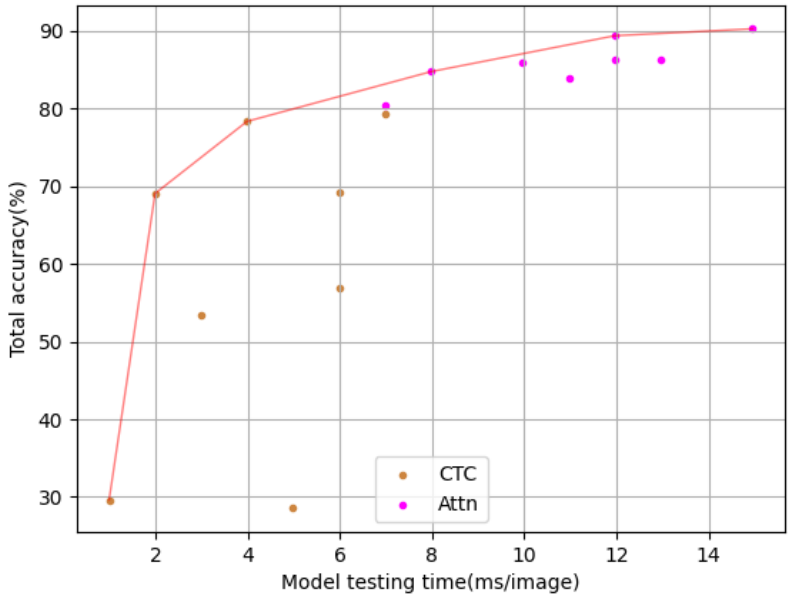

Figure 23. CTC versus Attn. (Model testing time.)

In the comparison of model testing times using different feature extraction networks, the VGG network performs better at low model testing times. As the model test time increases, ResNet exhibits a better accuracy. Ultimately, the highest accuracy was achieved using ConvNext's network structure, but at the same time, the model testing time was also the highest, as shown in Figure 25.

Finally, we performed comparisons of the network structure in terms of accuracy, the model testing time and the number of parameters. The accuracy was obtained by averaging the recognition results for the combination of models using this module. This is shown in Table 2.

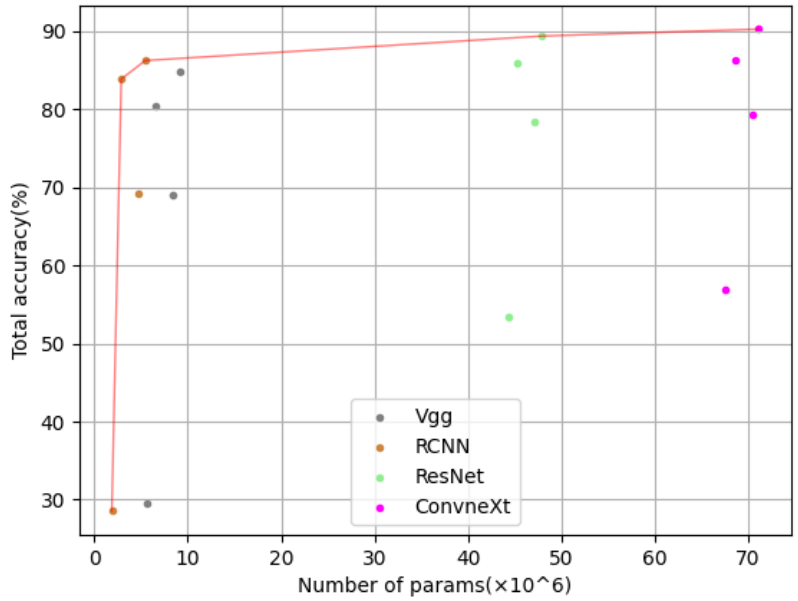

Figure 24. Comparison of feature extraction networks. (Number of parameters.)

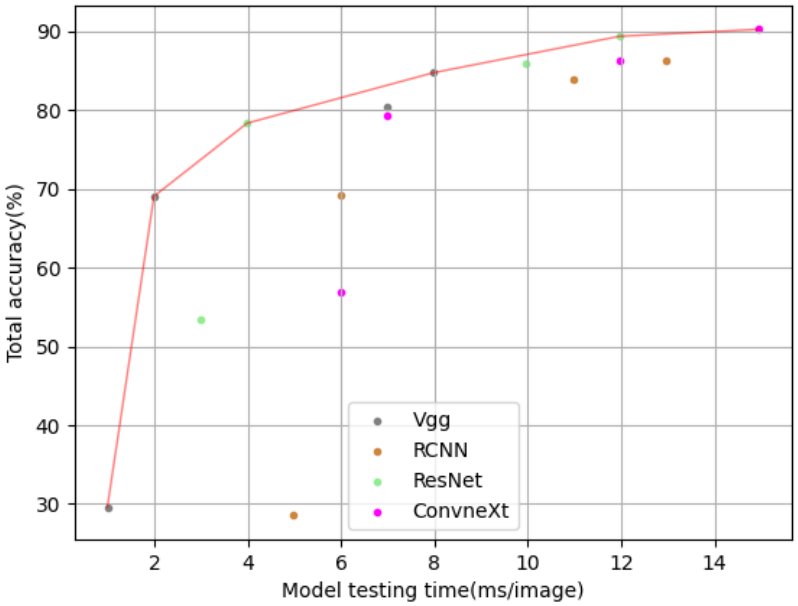

Figure 25. Comparison of feature extraction networks. (Model testing time.)

Table 2. The comparison of recognition accuracy, normalized edit distances, and number of parameters of the network.

Stage	Model	ACC.%	Time ms/Image	Norm_ED	Params × 10^6
Feat.	VGG	65.93	4.48	0.830	5.55
	RCNN	66.93(+1.00)	8.72	0.835(+0.005)	1.86
	ResNet	76.71(+10.78)	7.23	0.905(+0.075)	44.26
	ConvNeXt	78.13(+12.20)	9.97	0.920(+0.090)	44.26
Seq.	None	63.08	6.85	0.810	N/A
	BiLSTM	80.77(+17.69)	7.47	0.932(+0.122)	2.89
Pred.	CTC	58.01	4.24	0.795	0.01
	Attn	85.83(+27.82)	10.96	0.950(+0.155)	1.03

We find that in the feature extraction phase, as the number of parameters and the testing time of the feature extraction network increase, the network becomes increasingly complex and the accuracy of recognition increases. On the other hand, although the recognition results using ResNet and ConvNeXt as the feature extraction network are better in terms of the accuracy of recognition, they are accompanied by sudden increases in the number of parameters and time. In the sequence modeling stage, using the BiLSTM network to process text features is much better than removing BiLSTM directly. Although Borisyuk et al. [10] suggested that removing BiLSTM would improve the network performance, in the case of Uyghur recognition, the processing of text features is quite important, and more attention needs to be paid to the long- and short-term memory. In the final prediction stage, using the attention mechanism can significantly increase the recognition accuracy with a minimal increase in the number of model parameters, and the convergence speed of prediction using the attention mechanism is much faster than that using CTC. This also illustrated in the fact that the Uyghur language requires more attention to the semantic information of the text due to its long words and sticky nature with similar characters.

6. Conclusions

We have established a standardized, high-quality, and diverse Uyghur word image dataset and employed a unified three-stage recognition network structure for Uyghur text recognition. By replacing different stages of the network structure, we have identified the most suitable model for Uyghur text recognition. In terms of the trade-off between parameter size, time, and accuracy, the recognition model incorporating the attention mechanism demonstrated a favorable performance.

With a relatively small parameter size and short time, the combinations of VGG, BiLSTM, and the attention mechanism or RCNN, BiLSTM, and the attention mechanism achieved accuracies of 84.70% and 86.19%, respectively. For a higher accuracy, combinations of ResNet, BiLSTM, and the attention mechanism or ConvNeXt, BiLSTM, and the attention mechanism can achieve accuracies of 89.32% and 90.21%, respectively. Due to the specific characteristics of Uyghur script, which involves longer words and ligatures between characters, the attention mechanism is required to focus on global information, capture contextual details, and extract key information.

However, it should be noted that as the accuracy increases, the complexity of the network also grows. In the future, we aim to research and develop models that achieve an exceptionally high recognition accuracy while maintaining a low parameter size.

Author Contributions: Conceptualization, W.L. and Y.Z.; methodology, W.L. and Y.Z.; investigation, Y.S.; resources, W.L. and Z.W.; writing—original draft preparation, W.L.; writing—review and editing; Y.S., Y.Z. and Y.H.; visualization, Y.H.; supervision, W.L.; project administration, Y.Z.; funding acquisition, Y.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Liaoning Ning Department of Science and Technology, Natural Science Foundation of Liaoning Province under Grant 2020MZLH06.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The used data are available on request.

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

Appendix A

Figure A1. *Cont.*

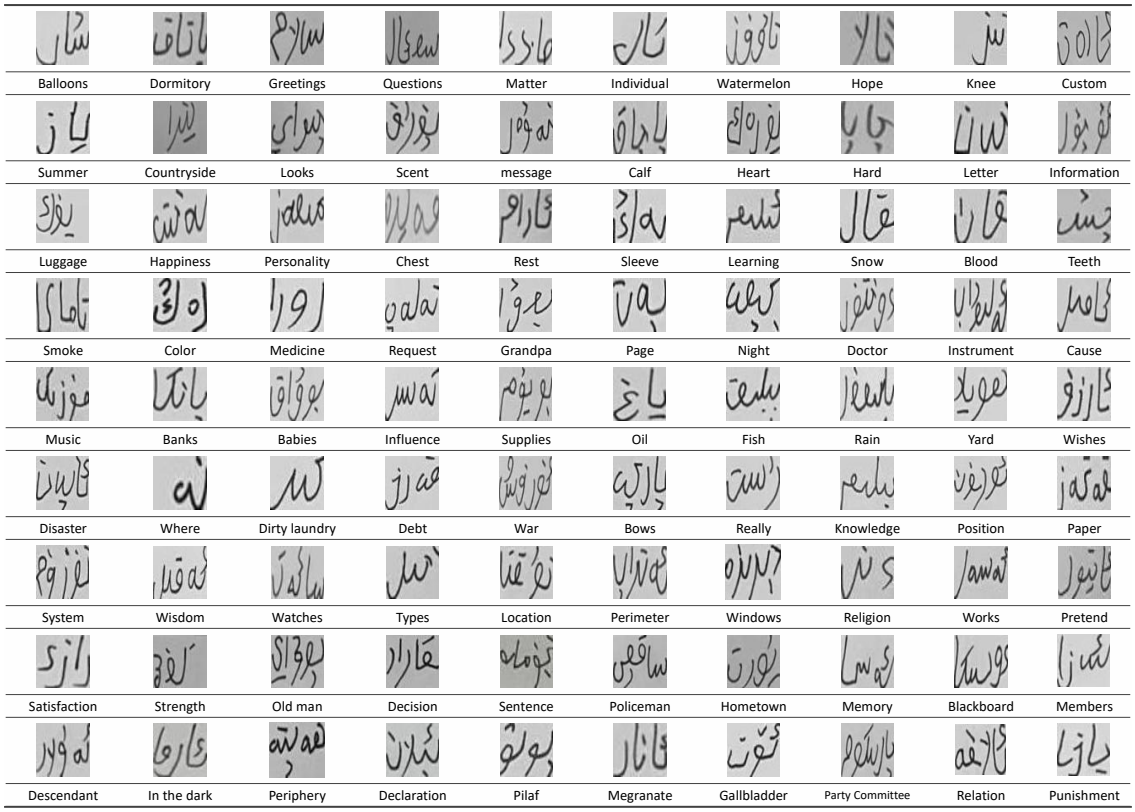

Figure A1. Different writing styles of handwritten Uyghur words.

References

1. Graves, A.; Fernández, S.; Gomez, F. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, USA, 25–29 June 2006; pp. 369–376.
2. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
3. Wang, J.; Hu, X. Convolutional Neural Networks With Gated Recurrent Connections. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 3421–3435. [CrossRef] [PubMed]
4. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
5. Liu, Z.; Mao, H.; Wu, C.Y.; Feichtenhofer, C.; Darrell, T.; Xie, S. A convnet for the 2020s. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 11976–11986.
6. Mnih, V.; Heess, N.; Graves, A. Recurrent models of visual attention. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2204–2212.
7. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
8. Shi, B.; Bai, X.; Yao, C. An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2298–2304. [CrossRef] [PubMed]
9. Lee, C.Y.; Osindero, S. Recursive Recurrent Nets with Attention Modeling for OCR in the Wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 21–23 June 2016; pp. 2231–2239.
10. Borisyuk, F.; Gordo, A.; Sivakumar, V. Rosetta: Large scale system for text detection and recognition in images. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 71–79.
11. Baek, J.; Kim, G.; Lee, J.; Park, S.; Han, D.; Yun, S. What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27–28 October 2019; pp. 4714–4722.
12. Diaz, D.H.; Qin, S.; Ingle, R.; Fujii, Y. Rethinking text line recognition models. *arXiv* **2021**, arXiv:2104.07787.
13. Shi, B.; Yang, M.; Wang, X.; Lyu, P.; Yao, C.; Bai, X. ASTER: An Attentional Scene Text Recognizer with Flexible Rectification. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 2035–2048. [CrossRef]
14. Xie, Z.; Huang, Y.; Zhu, Y.; Jin, L.; Liu, Y.; Xie, L. Aggregation Cross-Entropy for Sequence Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 6531–6540.

15. He, S.; Hu, X. Chinese Character Recognition in Natural Scenes. In Proceedings of the 2016 9th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 10–11 December 2016; pp. 124–127.
16. Shi, B.; Wang, X.; Lyu, P.; Yao, C.; Bai, X. Robust Scene Text Recognition with Automatic Rectification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4168–4176.
17. Cheng, Z.; Bai, F.; Xu, Y.; Zheng, G.; Pu, S.; Zhou, S. Focusing Attention: Towards Accurate Text Recognition in Natural Images. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5086–5094.
18. Liu, Y.; Wang, Y.; Shi, H. A Convolutional Recurrent Neural-Network-Based Machine Learning for Scene Text Recognition Application. *Symmetry* **2023**, *15*, 849. [CrossRef]
19. Liao, M.; Wan, Z.; Yao, C.; Chen, K.; Bai, X. Real-time scene text detection with differentiable binarization. *AAAI* **2020**, *37*, 11474–11481. [CrossRef]
20. Chandio, A.A.; Asikuzzaman, M.D.; Pickering, M.R.; Leghari, M. Cursive Text Recognition in Natural Scene Images Using Deep Convolutional Recurrent Neural Network. *IEEE Access* **2022**, *10*, 10062–10078. [CrossRef]
21. Bhatti, A.; Arif, A.; Khalid, W.; Khan, B.; Ali, A.; Khalid, S.; Rehman, A.U. Recognition and classification of handwritten urdu numerals using deep learning techniques. *Appl. Sci.* **2023**, *13*, 1624. [CrossRef]
22. Faizullah, S.; Ayub, M.S.; Hussain, S.; Khan, M.A. A Survey of OCR in Arabic Language: Applications, Techniques, and Challenges. *Appl. Sci.* **2023**, *13*, 4584. [CrossRef]
23. Najam, R.; Faizullah, S. Analysis of Recent Deep Learning Techniques for Arabic Handwritten-Text OCR and Post-OCR Correction. *Appl. Sci.* **2023**, *13*, 7568. [CrossRef]
24. Wang, X. Research and Application of Key Technologies for Printed Uyghur Recognition. China XiDian University. 2017. Available online: https://kns.cnki.net/reader/review?invoice=E0BHzLmOAztuvDM6NECx5tY0qrvYJ9uyW%2FGjN%2FX9KGiWam%2BHGEAtL4BGdLgp21SL2FuGRlzFO8%2BRuX%2B3im7Sj7Ad769FhI5qWhENCPYhGtbttupPl%2FFVdCu1X7YFNTW5i53ieUC1p7ovIpDUkG3aPwpZYnOxVvdPDaU0trGTgL0%3D&platform=NZKPT&product=CMFD&filename=1017301920.nh&tablename=cmfd201801&type=DISSERTATION&scope=trial&cflag=overlay&dflag=&pages=&language=chs&trial=&nonce=327839BEC1664DD69EEF336A5EE6E039 (accessed on 3 July 2020).
25. Chen, Y. Research and Design of Uyghur Language Detection and Recognition Based on Deep Learning. Master's Thesis, China Chengdu University of Technology, Chengdu, China, 2020. Available online: https://kns.cnki.net/kcms2/article/abstract?v=3uoqIhG8C475KOm_zrgu4lQARvep2SAkyRJRH-nhEQBuKg4okgcHYvv4vXrBT6PYbsMn7WEdE2OP-_8B7-YusUQvfmf8uVLO&uniplatform=NZKPT (accessed on 9 August 2020).
26. Tang, J. Uyghur Scanned Body Recognition Based on Deep Learning. *China J. Northeast. Norm. Univ. (Natural Sci. Ed.)* **2021**, *13* , 71–76. Available online: https://kns.cnki.net/kcms2/article/abstract?v=3uoqIhG8C44YLTlOAiTRKibYlV5Vjs7iy_Rpms2pqwbFRRUtoUImHboqwGMQdpFVOy_Z6EXzOfvlndeg_RIeccuGRM3ph9Vp&uniplatform=NZKPT (accessed on 21 November 2021).
27. Xiong, L. Design and Implementation of Django-based Printed Uyghur Recognition System. *China J. Zhengzhou Univ. (Nat. Sci. Ed.)* **2021**, *53* , 9–14. Available online: http://www.xml-data.org/ZZDXXBLXB/html/3987c5aa-7f51-4e6c-8c33-9a1e21f7fe93.htm (accessed on 6 June 2021).
28. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.;Wei, Y.; Zhang, Z.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021 ; pp. 10012–10022.
29. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1492–1500.
30. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing System, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
31. Hendrycks, D.; Gimpel, K. Gaussian error linear units (gelus). *arXiv* **2016**, arXiv:1606.08415.
32. Saimati, B.; Gomel, A. *5000 Words Commonly Used in Uyghur*; People's Publishing House: Xinjiang, China, 2012. Available online: https://book.douban.com/subject/26690805/ (accessed on 3 July 2020).
33. Zeiler, M.D. Adadelta: An adaptive learning rate method. *arXiv* **2012**, arXiv:1212.5701.
34. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv* **2013**, arXiv:1312.6034.
35. Ribeiro, M.T.; Singh, S.; Guestrin, C. "Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1135–1144.
36. Lundberg, S.M.; Lee, S.I. A unified approach to interpreting model predictions. In Proceedings of the Advances in Neural Information Processing System, Long Beach, CA, USA, 4–9 December 2017.