scientific data

Check for updates

OPEN

DATA DESCRIPTOR

# Ancient Yi Script Handwriting Sample Repository

Xiaojuan Liu[1,4], Xu Han[2,4], Shanxiong Chen[3] ✉, Weijia Dai[1] & Qiuyue Ruan[1]

**The ancient Yi script has been used for over 8000 years, which can be ranked with Oracle, Sumerian, Egyptian, Mayan and Harappan, and is one of the six ancient scripts in the world. In this article, we collected 2922 handwritten single word samples of commonly used ancient Yi characters. Each character was written by 310 people respectively, with a total of 427,939 valid characters. We completed continuous handwritten text sampling, written by 250 people, with 5 texts per person, covering topics such as Yi astronomy, geography, rituals, and agriculture. In the process of data collection, we proposed an automatic sampling method for ancient Yi script, and completed the automatic cutting and labeling of handwritten samples. Furthermore, we tested the recognition performance of the sorted data set under different deep learning network models. The results show that ancient Yi script has diverse shape structures and rich writing styles, which can be used as a benchmark data set in related fields such as handwritten text recognition and handwritten text generation.**

## Background & Summary

The Yi are one of the many distinct ethnic minorities in China. They have a rich history dating back thousands of years, and over time, they have developed their own distinct culture. Ancient Yi is about 8,000 years old and is the script of the Yi people. Together with the five scripts of Oracle Bone, Sumerian, Egyptian, Maya, and Halabian, it is one of the six ancient scripts in existence. It has been in use for a long time and has left behind some priceless classics that have significant historical and societal value.

Major libraries, research centers, and translation agencies in China have amassed a sizable collection of ancient Yi works, as do certain organizations in the UK, Japan, France, and Switzerland. Religion, history, philosophy, literature, linguistics, medicine, astronomy, geography, and agricultural technology are all included in these Yi writings. There are currently around 100,000 volumes of old Yi books dispersed throughout the population. The majority of the Yi ancient texts have long since suffered significant harm as a result of a poor protection idea, thus it is imperative to implement digital protection and use.

At present, the digitization of ancient books primarily involves scanning, classifying, and storage after scanning, which can effectively converting paper documents into digital images. But it is impossible to retrieve, analyze and extract text contents from documents, which greatly limits the dissemination of ancient books. Therefore, the digitization of Yi ancient books needs to identify the scanned ancient books and convert them into computer text documents to enable information processing and analysis in Yi. As the carriers of ancient Yi books, stone engravings, cliff paintings, wooden slips and paper books are often blurred or incomplete due to their age, which brings great challenges to the recognition of ancient Yi books (as shown in Fig. 1). Therefore, the establishment of a standardized ancient Yi handwritten character library is the basic work for recognizing ancient Yi[1,2].

In collaboration with specialists from Guizhou Institute of Engineering and Application Technology's Yi Studies Institute, we finished creating 9,845 historical Yi characters and created an input technique utilizing pinyin and strokes[3]. After spending two years sampling multiple times in Bijie, Guizhou's minority communities, the team completed 2,922 handwritten monosyllabic samples of common old Yi characters.The sample of continuous written texts was completed, and 250 people wrote 5 texts each covering astronomy, geography, sacrifice, farming, and other aspects of Yi nationality. A total of 427,939 valid characters were written, with 310 persons writing each character.

[1]School of Artificial Intelligence, Chongqing University of Technology, Chongqing, 400054, China. [2]School of Electrical and Information Engineering, Tianjin University, Tianjin, 300072, China. [3]College of Computer & Information Science, College of Software, Southwest University, Chongqing, 400715, China. [4]These authors contributed equally: Xiaojuan Liu, Xu Han. ✉e-mail: csxpml@163.com

**Fig. 1** From left to right, ancient Yi written in stone, wooden calf and sheepskin respectively.


**Fig. 2** From left to right: Concise Yi-Han Dictionary, Yunnan-Chuanqian-gui Yi character set, general Yi dictionary.

In order to ensure that the dataset covers a variety of handwriting styles, the 310 participants include middle school students, college students who are learning Yi language, and teachers engaged in Yi education. Specifically, the study collected samples of 150 middle school students, 100 college students, and 60 teachers, ranging in age from 13 to 55 years old, with a gender ratio of 160 males and 150 females. In order to fully understand the factors that affect handwriting style, the study recorded the participants' educational background, occupational background, language usage frequency, writing tools, writing environment, and ethnic background in detail. In terms of educational background, there are 120 junior high school students, 80 high school students, and 110 college students; in terms of occupational background, there are Chinese teachers, mathematics teachers, and other subject teachers. In terms of language usage frequency, 180 participants use Yi writing every day, and 130 use it several times a week. Regarding writing tools, 80% of the participants use fountain pens, 15% use ballpoint pens, and 5% use pencils. Regarding the writing environment, 60% of the samples were collected in classroom environments, 30% in home environments, and 10% in other environments. In addition, among the 310 participants, 270 were from the Yi ethnic group and 40 were from other ethnic minorities.

The development of the ancient Yi character set is currently falling behind, with the current Yi character set mostly targeted at the standard Yi. The digitization of the ancient Yi script has faced significant obstacles in regions where it is still in use. As a result, human transcription, which is tedious and time-consuming, is the primary method utilized for the archiving of many historic books. Building the character set for the ancient Yi is therefore crucial. Figure 2 illustrates the accomplishments of several Yi scholars who have sorted through the old Yi characters.

A Concise Yi-Han Dictionary (Guizhou Edition) with over 6,000 historical Yi characters was released in 1991. It has been a standard reference work for studying ancient Yi ever since it was published. The Yunnan-ChuanQian-Gui Yi Characters Collection, which was released in 2004, gathered around 87,000 historical Yi characters from Guizhou, Sichuan, Guangxi, and other areas. These characters essentially comprised the majority of the characters found in historical Yi literature. In order to compile and publish the General Yi Dictionary, the Yi Studies Institute of Guizhou Institute of Engineering and Application Technology created an input method software based on Yi character components in 2016. While the contents of the ancient Yi books are handwritten, the collation of the ancient Yi handwriting is still behind schedule. The aforementioned studies are focused on gathering and compiling the ancient Yi standard printing fonts.

To complete the collecting and classification of ancient Yi handwriting, this work presents an automatic sample approach for ancient Yi handwriting based on the ancient Yi fonts provided by Yi Studies Institute of Guizhou Institute of Engineering and Applied Technology. Utilizing the Unicode encoding interval 0xAC01-0xD456, the font includes 2,922 frequently used ancient Yi characters, as illustrated in Fig. 3. This covers the characters found in the majority of ancient Yi texts[4].

On this basis, the research team aims to establish a public platform for intelligent recognition and document transcription and extraction of Yi language, similar to intelligent applications of Chinese, to provide convenient

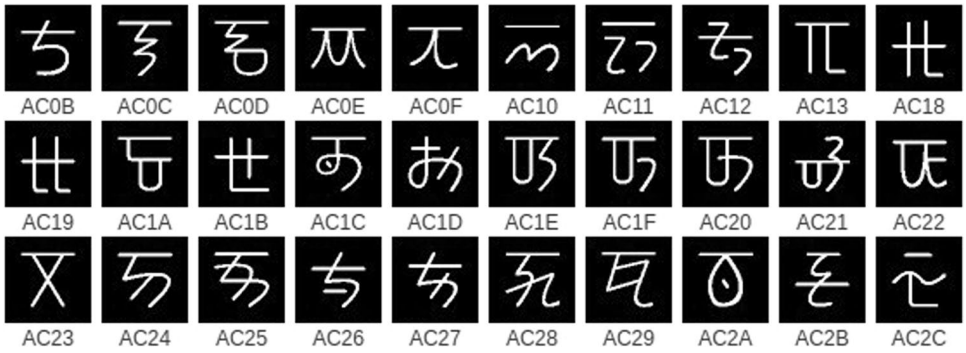


**Fig. 3** Traditional Yi fonts.


**Fig. 4** Sampling table Scanning samples.

digital tools for areas where Yi language is used. The research results will not only be promoted to Yi-populated areas in southwest China, such as schools and educational institutions. Still, they will also be extended to other areas where Yi language is used, including Vietnam, Myanmar and other places, as well as some overseas Yi immigrant groups, to support the learning and use of Yi language and help protect and inherit Yi culture. Based on China's Yi literature and materials, the research team will cooperate with Yi research institutions in other countries worldwide to promote international exchanges and cooperation in Yi language by sharing literature resources, ensuring that the research results truly benefit the Yi community and contribute to cultural heritage.

## Methods

**Sample Base Construction.** Two procedures are involved in creating the handwritten sample base: creating the sample table and automatically cutting and labeling the handwritten sample. First, as seen in Fig. 4, the ancient Yi handwriting sample table is automatically constructed with the use of the ancient Yi font. The table head and collection area make up the two primary sections of the sampling table.The table header comprises the QR code area, the sample table title, the writer's details, and the version and page number in the lower right corner, arranged from left to right. The version number and page number are encoded in utf-8 format and stored in the QR code region. For instance, the current version number is shown by vmfu-ref, while the current page number is indicated by p0. Starting from the first row, the collection area measures 18 rows by 14 columns. The handwriting of the even behavior writer is in the standard font found in the singular behavior library. A total of 126 characters can be collected per page, with the dotted line enclosing each character's collection area[5].

The jinja2 template engine is used to automatically fill the template according to the selected character set to generate the final tex file, which is then compiled to obtain the collection table. As illustrated in Fig. 5, tikz drawing library is used to draw and generate Tex template, and pyzbar library is used to package two-dimensional code[6].

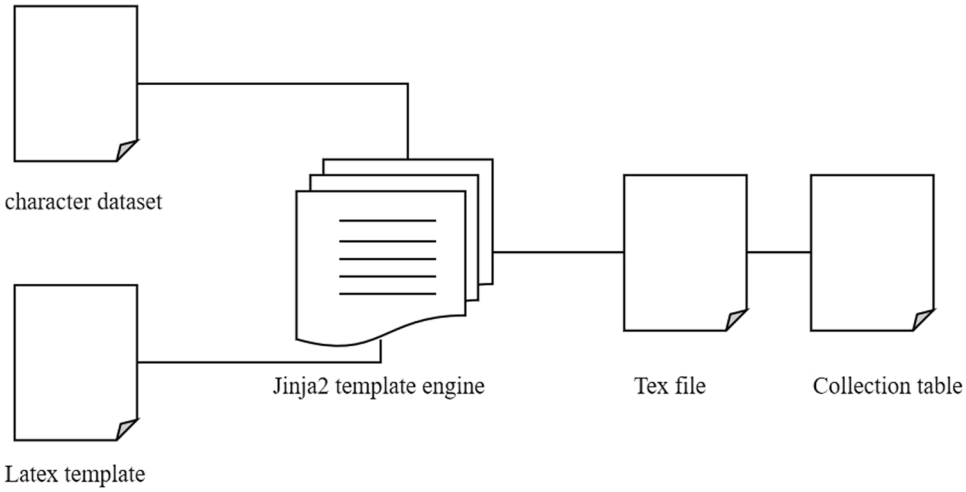


**Fig. 5** Sampling table generation process.

To achieve automatic annotation of handwritten samples, storing the character set in JSON format is also required. The JSON file, as seen in Fig. 6, stores the last four digits of the Unicode code of each character on each page as the value field and uses the page number of the sampling table as the key field. Simultaneously, these data are consistently kept beneath the data field, enabling the automatic annotation process to finish the handwritten sample of the page by simply reading the page number from the JSON file[7,8].

It is required to automatically cut and mark the handwritten samples in the adopted table after the sampling table is prepared and manually written. Figure 7 depicts the entire sample extraction procedure.

It is necessary to identify the exterior frame line once the background has been fixed. As seen in Fig. 9, the external border is found by fitting a quadrangle using OpenCV's polygon fitting algorithm[9]. The following are the steps in the polygon fitting algorithm:

**Algorithm 1** Polygon Fitting Algorithm.

**Require:** Curve $AB$
**Ensure:** Polygon after fitting
1: Initialization: threshold $0T$
2: Calculate the distance between all points on the curve $A$ and $B$ and line $AB$, and find the point with the largest distance from line $AB$, as shown in Figure 9
3: Compare the size of the maximum distance $H$ with the threshold value $0T$. If $H$ is less than $0T$, the point cannot be used as the end point of the fitted polygon, otherwise go to the next step
4: Point $C$ divides the curve into $AC$ and $CB$ curves, and repeat steps 2-3 for these two curves
5: If the number of edges fitted is greater than 4, the threshold is expanded to $0T$, and vice versa. Until the final number of fitted edges is equal to 4
6: **Finish**

First, the scanned picture of the sample table is preprocessed, and noise is removed by filtering with a $5 \times 5$ Gaussian check image. After that, the image's background has to be fixed. The backdrop correction is necessary since the scanning procedure may result in uneven illumination, which will also create a white border around the sampling table. The border's stark color contrast with the sample table's background has a significant impact on the subsequent outer frame line identification. The gray histogram must be used to examine the gray interval of the full image in order to solve this problem. The gray value range of 0 to 255 is divided into 8 gray intervals. The pixel interval with the densest distribution of pixel values is then identified and utilized as the background color interval. Subsequently, every pixel value in the image is contrasted with the lowest value inside the interval, and any pixels exceeding that value are assigned the pixel value. The background is depicted in Fig. 8 both before and after restoration.

The border line must be corrected once it has been identified by quadrilateral fitting[10]. This is because the quadrilateral that results from fitting is irregular, and since a rectangle is the final area we wish to cut from the sample, the irregular quadrilateral must also be transformed into a rectangle using perspective transformation.The target rectangle following the quadrilateral's perspective translation can be found using opencv's minAreaRect() function. This is the minimum rotating outer rectangle of the quadrilateral. Rigid body transformation, affine transformation, and translation transformation make up perspective transformation. The following are the precise steps in the algorithm:

Firstly, translation transformation is carried out on the image. The mathematical model of translation transformation is as follows:

$$\begin{cases} x_p = x + x_1 \\ y_p = y + y_1 \end{cases} \quad (1)$$

```
{
    "title": "贵州彝文常用字符",
    "author": "[redacted]",
    "address": "[redacted]",
    "from": "简明彝汉字典（贵州本）—贵州民族出版社",
    "size": 2162,
    "size_per_page": 126,
    "data": {
        "P00": ["AE52","AC41","ADC6","AF0C","AC3C","B0FE","AE36","ADA9","BA6A","ACDB","AC01","AF44","AE06","B082","B436",
        "P01": ["AC7E","B412","AF20","AF6D","CF5A","B433","AF10","AF66","AD52","B290","CADE","B14F","AE2F","BDEB","B076",
        "P02": ["B369","AF46","B15C","C70F","B2D0","AC67","B208","C1D8","C0C8","AC29","B327","AF4C","B378","AE91","AE57",
        "P03": ["B3C0","BC26","AE1F","B3A1","CCE0","D174","AD6E","B18D","AD68","B384","AF62","C037","C038","B09B","ADE1",
        "P04": ["ACE0","AE6E","B324","B3FC","B261","AFC7","B338","ACFB","B257","B221","B104","AFC5","CBAE","B340","CA14",
        "P05": ["AF6F","B3E0","AE14","BC96","AC69","AD50","ACF9","B1FE","B343","ACF7","B116","ACF3","C5AD","B154","B153",
        "P06": ["ADCA","AD99","B26B","B2C9","B12D","B239","AF63","AF52","AD26","AD48","AD4B","B229","AE3F","AE96","B430",
        "P07": ["AFB0","B27A","AFA7","D0F3","D0F5","BFC7","B2BF","CDFD","B197","B1F8","B2E6","AE58","B0C5","AD57","B41F",
        "P08": ["B396","AD45","B718","AE49","CB46","B2EB","B2EA","B42A","B3E9","AE3C","AC59","ACD5","B124","AC6E","C96A",
        "P09": ["ADDE","B0A4","B0E7","B422","AFDE","B3FF","ADF8","AD2A","AD5A","AD5C","B0E8","AF7F","ACE4","AF83","AE7B",
        "P10": ["B400","AD5F","AE09","ACBA","B0D6","AEB9","ADDD","AE80","B240","ADBE","AEDC","B22A","AD24","B1C3","AD92",
        "P11": ["B403","B29E","AF2D","ACC8","AEF5","B17A","CE22","AFED","B0E0","AD66","B2DD","B1C9","B2CC","AD1D","B2CA",
        "P12": ["B06B","AF4D","ADB0","B3BB","AF45","B3C2","ADC2","AD01","AD2B","B3ED","BA55","AC63","AEB1","B11C","AC81",
        "P13": ["AECF","B189","AE31","B0B2","B1D9","B146","B1B9","AE26","AC73","ADD5","AF67","AE9F","BA1A","ACFD","AC0F",
        "P14": ["B4A4","B6C9","AC27","AE8C","B322","AC7A","AF9A","AF9C","AC7C","B38B","C9A1","AC8C","AF60","AC9B","B317",
        "P15": ["B092","B905","AF3D","AFC3","AC8B","B3BD","AC87","B1CB","AE56","B0D3","B397","AECA","CDCC","AEBB","B350",
        "P16": ["B08B","AD09","AEDE","B00F","AD59","B018","AE11","AFD3","ADC1","B2A3","B0BA","B41C","AE8D","B41B","B423",
        "P17": ["AEC3","B754","B375","B09E","AE7C","AECB","AEC9","AEC0","B0F9","B218","AE93","B1B6","B282","B2E1","AEFF",
    }
}
```


**Fig. 6** JSON format of the ancient Yi character set.

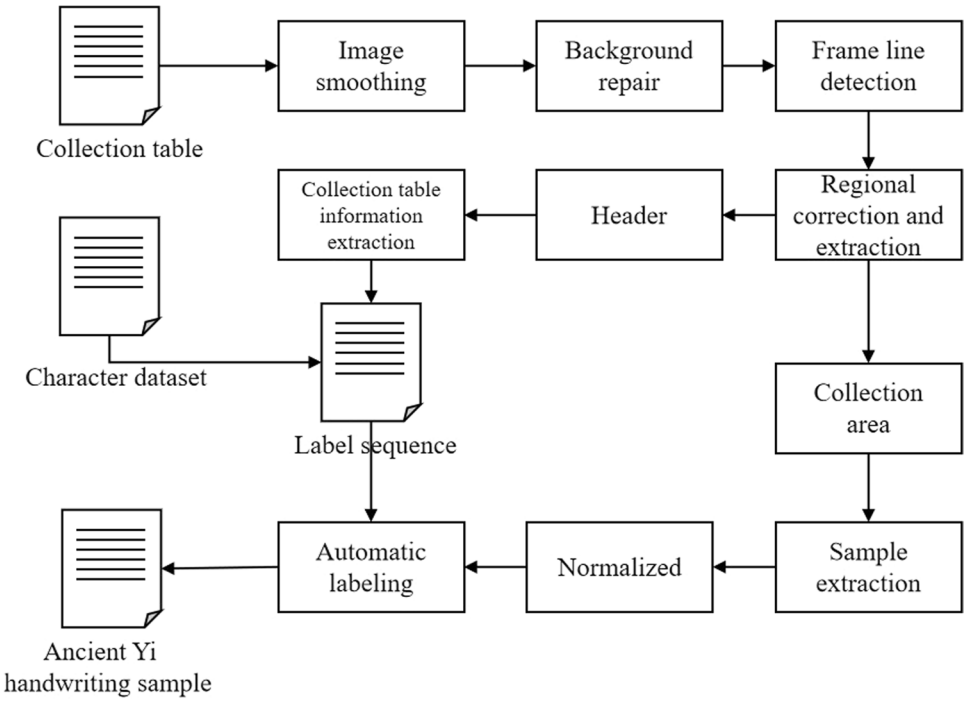


**Fig. 7** Flowchart of automatic sample extraction and labeling.

$(x_p, y_p)$ represents the new coordinate after translation, $x_1, y_1$ represents translation parameter, translation transformation can be expressed by the following matrix transformation form, where $T_p$ is translation transformation matrix:

$$T_p = \begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2}$$


**Fig. 8** (left) before background repair (right) after background repair.

After translation transformation, affine transformation is carried out, and the mathematical model of affine transformation is as follows:

$$\begin{cases} x_f = a_1 x + a_2 y + x_1 \\ y_f = b_1 x + b_1 y + y_1 \end{cases} \tag{3}$$

$(x_f, y_f)$ is the new coordinate after the affine transformation, $a_1$, $a_2$, $b_1$, $b_2$ represents the affine parameter, and the matrix of the affine transformation is expressed as follows, where $T_f$ is the affine transformation matrix:

$$T_f = \begin{bmatrix} a_1 & a_2 & x_1 \\ b_1 & b_2 & y_1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4}$$

Finally, the rigid body transformation is carried out, and the mathematical model of the rigid body transformation is as follows:

$$\begin{cases} x_g = \cos\theta x - \sin\theta y + x_1 \\ y_g = \sin\theta x + \cos\theta y + y_1 \end{cases} \tag{5}$$

**Algorithm 2** Perspective Transformation Algorithm.

**Require:** Coordinates of a point in the image $(x, y)$.
**Ensure:** Corresponding coordinates after perspective transformation $(x_t, y_t)$
1: Initializing translation parameters $(x_1, y_1)$, affine parameters $a_1, a_2, b_1, b_2$, rigid body parameters $\theta$.
2: Calculate the coordinate values after translation $x_p = x + x_1, y_p = y + y_1$.
3: After calculating affine transformation of coordinates $x_f = a_1 x_p + a_2 y_p + x_1, y_f = b_1 x_p + b_1 y_b + y_1$.
4: Calculate the coordinate values after the rigid body transformation $a_1, a_2, b_1, b_2$.
5: $x_g = \cos\theta x_f - \sin\theta y_f + x_1, y_g = \sin\theta x_f + \cos\theta y_f + y_1$.
6: **Finish**

$(x_g, y_g)$ is the new coordinate after the rigid body transformation, $\theta$ represents the rigid body parameter, and the matrix of the rigid body transformation is expressed as follows, where $T_g$ is the rigid body transformation matrix:

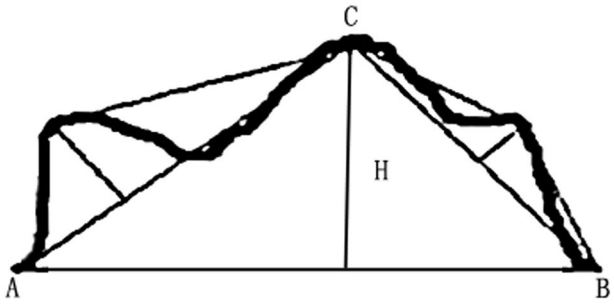


**Fig. 9** Schematic diagram of polygon fitting algorithm.

$$T_g = \begin{bmatrix} \cos\theta & -\sin\theta & x_1 \\ \sin\theta & \cos\theta & y_1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{6}$$

Then, the perspective transformation model is as follows, where $(x_t, y_t)$ is the new coordinate after the perspective transformation:

$$(x_t, y_t) = T_p T_f T_g (x, y) \tag{7}$$

According to the above analysis, the algorithm of perspective transformation is described as follows:

The outside frame line is transformed into a rectangle with a specific rotation angle after perspective transformation. The openCV minAreaRect() function can be used to determine the rotation angle. Lastly, by rotating the rectangle in accordance with the Angle, the corrected detection area may be achieved. The comparison before and after regional correction is displayed in Fig. 10.

The handwritten example can be automatically trimmed after picture correction. Initially, the pyzbar module may read the image's two-dimensional coding information to determine the sample table's page number. Next, the 14 × 18 aspect ratio is used to divide the collection area into equal sections. It is not required to gather the handwritten portions of even lines for the standard typefaces of singular lines; instead, the handwritten portions of even lines are chopped. Finally, using the previously obtained page numbers, each handwritten sample is matched to a field in the JSON file one at a time. The label of each sample is then appended to the file name of the cut picture file.In other words, the samples have finished their automatic tagging. Below is the pseudo-code for the example portion that is automatically chopped and labeled[11,12]. The sample collected following the automated cutting is depicted in Fig. 11.

To ensure the integrity of the dataset and facilitate its use by other researchers, this article presents the complete metadata of the dataset in the form of a table, as shown in the Table 1:

**Sample Increment and Sample Library Storage.** After the samples from the sampling tables were extracted, a total of 2418 sampling tables and 297414 samples were obtained. Following the samples' manual verification, 3164 unqualified samples, a 1.06% failure rate, and a high of 17 unqualified samples in a single character group were found. Smearing and writing errors during the writing process are the primary cause of nonconformity. A few nonconformity samples that were eliminated are depicted in Fig. 12.

Background removal of all samples is done once unqualified samples are eliminated[13,14]. As can be observed, samples obtained via the aforementioned procedures are nevertheless tainted with background noise, primarily in the form of dotted lines. By establishing a few heuristic guidelines using connected region analysis, we may eliminate these noises. The example image is first transformed into a gray-level map, which is then binarized. Following this, the connected region that satisfies the following criteria is eliminated: (1). Points and lines are defined as connected areas with a surface area of fewer than 8 pixels. (2) A dashed line is defined as a region that is less than one-fifth of the image's length, parallel to the edge, and within 15 pixels of the edge.Remove the extra background edges surrounding the boundaries and leave only the character sections after eliminating the dots and dashed lines. Ultimately, every sample is standardized to 64 × 64 pixels.A portion of the sample following background removal and normalization is seen in Fig. 13.

This research uses classical morphological transformation to expand the data of the gathered samples and diversify the samples. Corrosion, expansion, affine transformation, rotation, translation, and the addition of random Gaussian noise are the principal morphological modifications used. The incremental samples are also adjusted to 64 × 64 pixels after edges are removed. Part of the sample following increment is displayed in Fig. 14.

All samples must be saved when the sample increment is finished. Based on the MNIST data collection, this work establishes a new data set storage format.The whole file is made up of a header and a data stream, as Table 2 illustrates.The sample total, check code, and MagicNumber are all contained in the 40-byte file header. Because the Unicode encoding of the first character in this character set is 0xAC01, the default MagicNumber of this data set is 0xAC, which serves to both uniquely identify the data set file and prevent software error authentication. The hash value that is obtained after the SHA-256 algorithm processes the data flow to guarantee its integrity is known as the parity code. 32 bytes, or 256 bits, make up the value.The data stream comprises a single, continuous

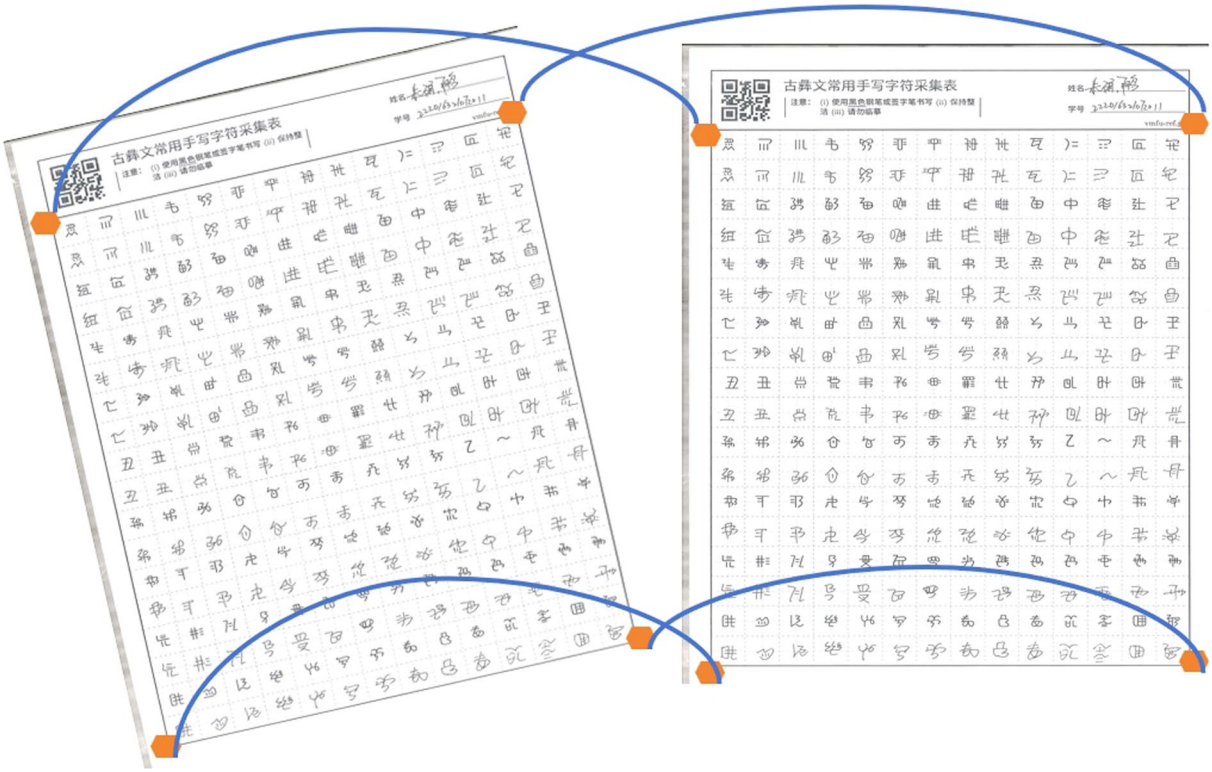


**Fig. 10** (left) before tilt correction (right) after tilt correction.

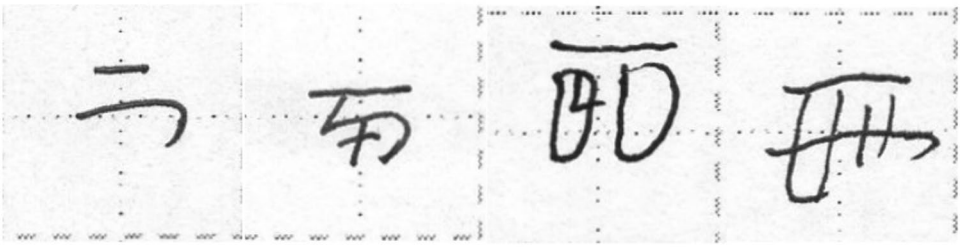

**Fig. 11** Partial handwriting samples taken from the sampling table.

data item that is made up of an image label, an image width, an image height, and an image bitmap. The label is two bytes in size and is formatted in an unsigned numeric format that corresponds to the Unicode encoding.

**Limitations of Methods and Datasets.** Some methods and dataset limitations in this study need to be further discussed and addressed. These improvements are necessary to enhance the comprehensiveness of the dataset and the reliability of the handwriting recognition system.

One of the main problems is that there are a large number of variant characters in Yi script, that is, different writing forms or variations of the same glyph. Because the diversity and frequency of variant characters were not fully considered when collecting samples, some variant characters were not fully collected. This limits the comprehensiveness and representativeness of the data set when dealing with variant characters, which in turn affects the performance of the recognition system in practical applications.

In addition, the participants came from different levels of the Yi people, including middle school students, college students, and teachers engaged in Yi education, which resulted in significant differences in writing quality. Although the research team tried their best to eliminate clerical errors during the data screening process, there were still a small number of omissions. These errors may lead to a small number of inaccurate samples in the data set, thus affecting the training effect and final performance of the model.

Another issue to note is the variability in handwriting styles. Participants include experts and scholars with more standardized handwriting, as well as students with more irregular handwriting. This difference leads to inconsistencies in handwriting styles in the dataset. Although this diversity can enhance the generalization ability of the model, it also increases the difficulty of processing highly variable handwriting, which may cause the model to be unable to accurately recognize certain styles of handwriting, thus affecting the reliability and validity of the dataset.

| Metadata Item | Description |
|---|---|
| Dataset Name | Ancient Yi Script Handwritten Character Dataset |
| Dataset Description | The dataset contains character samples for handwritten Yi character recognition, a total of 2922 commonly used Yi glyphs, each character is written by a different participant. |
| Collection Time | January 2021 to December 2021 |
| Collection Location | Sichuan Province and Guizhou Province, China |
| Participant Information | 9845 Yi writers of different ages, genders, and occupational backgrounds |
| Data Format | The data is stored in image format with a resolution of 300 DPI. Each character sample is stored as a separate image file, with filenames containing character codes and participant IDs. |
| Metadata Recording | Each image file is accompanied by a JSON metadata file containing character codes, participant IDs, collection time, collection location, writing tools, and image processing steps. |
| Image Preprocessing | All images undergo noise filtering, binarization, and normalization to improve accuracy and consistency of character recognition. |
| Cultural Sensitivity | The dataset includes significant cultural and linguistic values. When using the data, respect for the uniqueness of Yi culture and language is required. Avoid misunderstanding and misrepresentation of Yi culture and language. |
| Data Sharing and Collaboration | Researchers are encouraged to collaborate with the Yi community and data providers to ensure the accuracy and social impact of research results. |

**Table 1.** DatBenchmark results using different models on the dataset.

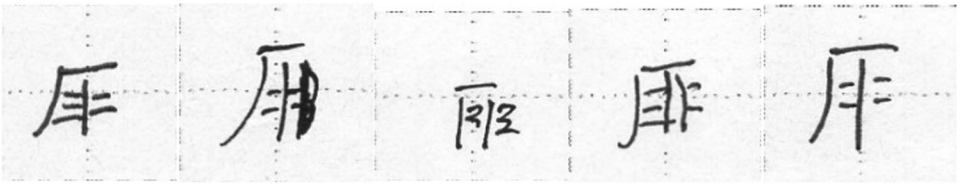

**Fig. 12** Part of the unqualified handwritten sample (the qualified sample is on the far left).

**Algorithm 3** Sample Automatic Cutting and Labeling Algorithm.

**Require:** Corrected sample table image
**Ensure:** Handwritten sample image list *hand_chars*, label list *labels*

The number of rows in the initial sampling table (header + collection area) is $unit\_h = 20$, the number of rows in the collection area is $unit\_body\_h = 18$, and the number of columns in the collection area is $unit\_w = 14$
Calculate the height of the entire image
Calculate the starting point of the vertical direction of the image in the acquisition area $start = (height/unit\_h) * 2$
Obtain the image of the collection area $body = image[start :]$, the width of the collection area $body\_width$, and the height of the collection area $body\_height$
**for** *hid* in $range(unit\_body\_h)$ **do**
  $h1 = hid * (body\_height/unit\_body\_h)$
  $h2 = (hid + 1) * (body\_height/unit\_body\_h)$
  **for** *wid* in $range(unit\_w)$ **do**
    $w1 = wid * (body\_width/unit\_w)$
    $w2 = (wid + 1) * (body\_width/unit\_w)$
    Single character sample $chars = body[h1 : h2, w1 : w2]$
    **if** $hid\%2 \neq 0$ **then**
      Get the current page number $p$ according to the QR code information
      Find the corresponding label through $p$ in the JSON file and store it in *labels*
    **end if**
    Store the word sample *chars* into *hand_chars*
  **end for**
**end for**
**return** *labels*, *hand_chars*
**Finish**

## Data Records

The Yi dataset[15] contains 427,941 samples of 2922 classes, where each class represents a unique Yi glyph character. The dataset is divided into training sets and test sets, and we make sure they are disjoint. The sample number of verification set accounts for 20% of the total sample number. The training set and the validation set are divided according to the proportion of the number of samples in the dataset and belong to 2922 categories. Due to the frequency of appearances in True sourcebooks, it is class unbalanced.

The Yi dataset[15] can be accessed at Science Data Bank10 (https://doi.org/10.57760/sciencedb.18011). We grant free access to the dataset, without the need for user registration. All images are grouped by folder, where the label AC0C-D654 represents the category label. The same folder belongs to the same category. Each image

Fig. 13 Normalizes and removes background samples.

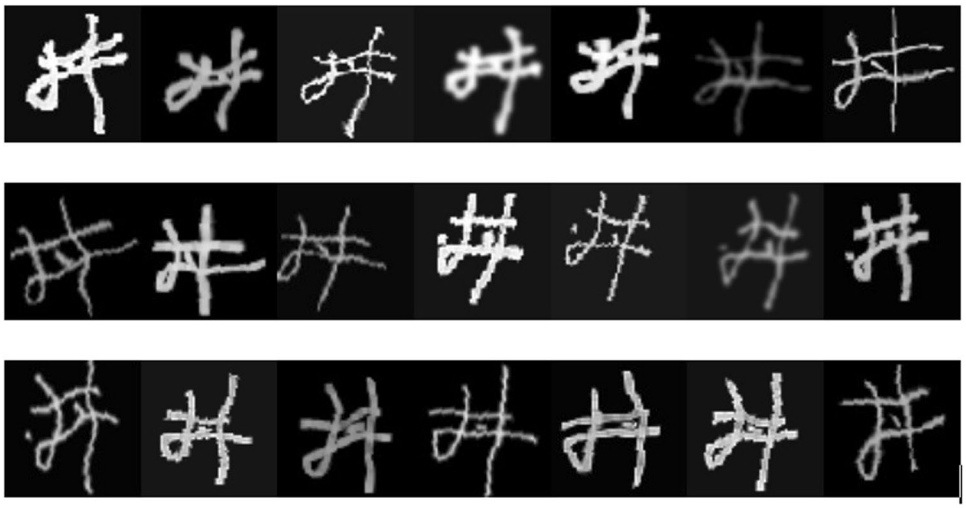

Fig. 14 Sample after increment.

| Data item | Type | Size(bytes) | Description |
|---|---|---|---|
| Magic Number | Uint32 | 4B | Unique identification of the data set. The default is 0xAC |
| Sample count | Uint32 | 4B | |
| Check code | Uint256 | 32B | SHA-256 check code |
| Tag | Uint16 | 2B | Unicode Numerical form |
| Image width | Uint16 | 2B | |
| Image height | Uint16 | 2B | |
| Bitmap | Uint8 array | width*height | A sequence of pixels arranged in rows |

Table 2. Data set storage format.

naming format is similar to "*********.*******-ACOC", the "*" represents different 0-9 similar Numbers of other different image tags, "ACOC" category name on behalf of the folder.

## Technical Validation

We immediately compare the compiled data set using several network models. Below, we briefly outline the key models used in the evaluation, and their relevance to the handwriting recognition task:

1. LeNet[16]: As an early convolutional neural network, LeNet contains two convolutional layers and three fully connected layers, extracting image features through alternating convolutional layers and pooling layers. Its structure is simple and suitable for handling basic image classification tasks, such as handwritten digit recognition. LeNet has been widely used in handwritten character recognition tasks, especially its performance on the MNIST dataset, establishing it as a classic.
2. AlexNet[17]: AlexNet contains five convolutional layers and three fully connected layers, using ReLU activation function and dropout technology, specifically used to extract local features of the image, and reduce the size of the feature map through the pooling layer. It performs well in processing handwritten text data

| Verification model | Data set (%) |
|---|---|
| LeNet | 87.7 |
| AlexNet | 87.4 |
| Inception | 96.0 |
| VGG16 | 94.9 |
| ResNet50 | 97.6 |
| RegNet | 97.3 |

**Table 3.** DatBenchmark results using different models on the dataset.

| Parameters | Data set (%) |
|---|---|
| LR=0.001, Dropout(True) | 78.1 |
| LR=0.0005, Dropout(True) | 80.6 |
| LR=0.001, Dropout(False) | 89.9 |
| LR=0.0005, Dropout(False) | 94.8 |

**Table 4.** Benchmark results on AlexNet.

| Parameters | Data set (%) |
|---|---|
| LR=0.001, Dropout(True) | 96.3 |
| LR=0.0005, Dropout(True) | 97.6 |
| LR=0.001, Dropout(False) | 97.1 |
| LR=0.0005, Dropout(False) | 97.5 |

**Table 5.** Benchmark results on Inception.

| Parameters | Data set (%) |
|---|---|
| LR=0.0001, batchsize=64 | 94.8 |
| LR=0.0005, batchsize=64 | 94.2 |
| LR=0.0001, batchsize=50 | 95.7 |
| LR=0.0005, batchsize=50 | 93.6 |

**Table 6.** Benchmark results onVGG16.

| Parameters | Data set (%) |
|---|---|
| LR=0.001, Relu=ReLU(inplace=True) | 98.2 |
| LR=0.001, Relu=LeakyReLU | 98.0 |
| LR=0.001, Relu=ELU() | 97.2 |
| LR=0.001, Relu=Swish() | 97.8 |

**Table 7.** Benchmark results on ResNet.

with rich details and style variations, achieved significant success in the ImageNet image classification competition, and was subsequently applied to various handwritten text recognition tasks.

3. Inception[18]: The Inception network uses convolution kernels of different sizes and pooling operations in parallel through the Inception module to capture multi-scale features, which enhances the network's representation capability and computational efficiency. This ability to capture multi-scale features makes it perform well when processing images with complex local structures. The Inception architecture has also been used for a variety of text recognition tasks, especially when processing complex fonts and highly diverse handwritten texts.
4. VGG16[19]: VGG16 gradually deepens the network to 16 or 19 layers by using a series of $3 \times 3$ small convolution kernels and max pooling layers. Its deep structure improves the ability to extract image features, making it suitable for various image classification tasks, especially in application scenarios that require deep feature extraction. The architecture of VGG16 has been verified in multiple handwritten character recognition and document analysis tasks, and is especially suitable for occasions where fine feature extraction is required.
5. ResNet50[20]: ResNet50 introduces a residual learning framework, allowing the network to learn the difference between the input and its residual mapping, solving the vanishing gradient problem in deep network training. The residual block skips certain layers through identity shortcut connections, allowing ResNet50 to handle ancient Yi script datasets with complex levels and high variation. The ResNet architecture has

| Parameters | Data set (%) |
|---|---|
| LR=0.001, depth=200 | 98.0 |
| LR=0.0005, depth=200 | 97.7 |
| LR=0.001, depth=400 | 98.1 |
| LR=0.0005, depth=400 | 97.5 |

**Table 8.** Benchmark results on RegNet.

been successfully applied to many complex image recognition tasks, including handwritten text recognition, and performs particularly well when processing character data with high variability.

6. RegNet[21]: RegNet optimizes performance and computational efficiency by controlling activation size and network width. Its adaptive adjustment of network width and depth enables it to perform well on large-scale and highly diverse datasets. RegNet has been used in a variety of image classification and text recognition tasks, especially in situations where the model complexity needs to be flexibly adjusted to meet the needs of different datasets, showing strong adaptability and efficiency.

To determine the overall classification accuracy, each sample in the test set is examined, and the model's predictions are compared to the actual label. Once the test set's classification accuracy has been determined, the number of samples that were properly classified by the total number of samples on the test set can be divided to determine the test set's average classification accuracy. The end result of assessing the dataset benchmark is this average classification accuracy, which is a measurement of classification accuracy over the whole test set.as displayed in Table 3.

Following data augmentation procedures like rotation and salt-and-pepper noise reduction, we then employ a number of different algorithms to assess the data set. Twenty percent of the data set samples are utilized as the verification set and eighty percent of the data set samples are used as the training set in each model training procedure. Tables 4 through 8 display the average classification accuracy of the same algorithm under various parameter modifications that are acquired via adjusting specific parameters of various algorithm models[22–24].

We can make the following observations based on the results. First, as the algorithm's time is provided from far to near, the average performance on the data set gets better and better. Following data improvement, there is a direct and significant interference with the LeNet network's ability to learn features, making it impossible to get the data set's benchmark test results. Nonetheless, the AlexNet algorithm's average performance fluctuates only by 85%, and the value fluctuates significantly depending on the settings. On the data set, the ResNet50 and RegNet algorithms can still achieve 97%.

This illustrates how complicated our dataset is, and for simpler datasets, LeNet and AlexNet's performance might be more comparable to ResNet's. For instance, the LeNet model has significant interference since it is mostly employed for tasks related to handwritten digit recognition, which differs greatly from the classification tasks in our dataset. AlexNet, on the other hand, is a member of the early convolutional neural network model. Its inability to capture abstract and complicated features in large-scale data sets may be due to its design, network depth, parameter number, and feature combination capabilities. Models with deeper complexity, more parameters, and more potent feature extraction and expression capabilities-like Inception, VGG16, ResNet50, and RegNet-better adapt to complicated data sets and produce superior benchmarking results.

With 2922 Yi words for a total of over 400,000 graphs in our dataset, many data improvement procedures are run on the dataset to suggest increasingly difficult classification tasks. Furthermore, the intricate characteristics of the Yi characters themselves, along with the potential for significant image degradation from noise, occlusion, and blurring, will make the classification assignment extremely challenging.

Second, on data sets, ResNet and RegNet perform noticeably better than other algorithms. The two network models may adapt more effectively to data sets with complex features and have more potent feature extraction and expression capabilities because of the deeper network structure. It can suit a wider range of data sets and extract more intricate features from the data with more parameters. ResNet helps to tackle the issue of deep network degradation and enhances the network's performance on the dataset since it introduces a residual learning mechanism and has a deeper network structure. RegNet, on the other hand, applies models at various scales in accordance with the principles of network architecture, which offers superior scalability and adaptability and can more adaptably handle the peculiarities of various data sets. As a result, the dataset's performance can be enhanced by partially addressing real-world issues in Yi, such as various writing styles, noise, and occlusion.

However, the dataset's performance is not yet saturated. With an error rate of 7.4% on the dataset, the VGG16 utilized in this research can yet be improved. Even with VGG16's strong representational powers, these Yi recognition issues are still unresolved.

**Note**: Existing studies have used the expanded and category-added Yi dataset. For example, Chen Shanxiong *et al.*[25] proposed a dual-discriminator generative adversarial network (GAN) method in their 2022 article to restore ancient Yi characters, and used the expanded dataset in their study. In addition, Chen Shanxiong *et al*[26]. proposed a method for character detection of ancient Yi texts based on MSER and CNN in their 2020 study, also using the category-added dataset.

## Code availability

The ancient Yi script Handwriting samples are freely available online at GitHub. Tutorials for loading the dataset and code for training and testing Yi character recognition models are also publicly available without restriction. GitHub code link ((https://github.com/JueGeYuJun/Dataset-benchmark-code/tree/master).

Received: 22 April 2024; Accepted: 23 September 2024;
Published online: 30 October 2024

## References

1. Su, X., Gao, G., Wei, H. & Bao, F. A knowledge-based recognition system for historical mongolian documents. *International Journal on Document Analysis and Recognition (IJDAR)* **19**, 221–235 (2016).
2. Wu, Y. & Kit, C. Hong kong corpus of chinese sentence and passage reading. *Scientific data* **10**, 899 (2023).
3. Xie Wu, M. W., Yuping Lu. On the construction of guizhou ancient yi coded character set. *Chinese Information Journal* 153–158 (2014).
4. Yang, H. *et al*. Dense and tight detection of chinese characters in historical documents: Datasets and a recognition guided detector. *IEEE Access* **6**, 30174–30183 (2018).
5. Gao, F., Wang, Y., Yang, Z., Ma, Y. & Zhang, Q. Single image super-resolution based on multi-scale dense attention network. *Soft Computing* **27**, 2981–2992 (2023).
6. Ntirogiannis, K., Gatos, B. & Pratikakis, I. A combined approach for the binarization of handwritten document images. *Pattern recognition letters* **35**, 3–15 (2014).
7. Aizezi, Y., Jiamali, A., Abdurixiti, R. & Ubul, K. Research on the methods for extracting the sensitive uyghur text-images for digital forensics. In *Biometric Recognition: 13th Chinese Conference, CCBR 2018, Urumqi, China, August 11-12, 2018, Proceedings 13*, 709–718 (Springer, 2018).
8. Zhu, Y., Yao, C. & Bai, X. Scene text detection and recognition: Recent advances and future trends. *Frontiers of Computer Science* **10**, 19–36 (2016).
9. Zhang, C., Wang, W., Liu, H., Zhang, G. & Lin, Q. Character detection and segmentation of historical uchen tibetan documents in complex situations. *IEEE Access* **10**, 25376–25391 (2022).
10. Jinliang, Y., Lubin, W. & Xiaohua, W. A text region localization method based on connected component. *pattern recognition and artificial intelligence* (2012).
11. Xiaodong, J., Wendong, G. & Jie, Y. Handwritten yi character recognition with density-based clustering algorithm and convolutional neural network. In *2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, vol. 1, 337–341 (IEEE, 2017).
12. Aili, Y., Wang, Y., Liu, P., Abudiriyimu, A. & Ubul, K. Construction of uyghur scene text image database. In *Proceedings of the 2021 10th International Conference on Computing and Pattern Recognition*, 231–236 (2021).
13. Arica, N. & Yarman-Vural, F. T. An overview of character recognition focused on off-line handwriting. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* **31**, 216–233 (2001).
14. Rahman, A. F. R. & Fairhurst, M. C. Multiple classifier decision combination strategies for character recognition: A review. *Document Analysis and Recognition* **5**, 166–194 (2003).
15. Chen, S., Liu, X., Han, X., Dai, W. & Ruan, Q. Ancient Yi Script Handwriting Sample Repository. https://doi.org/10.57760/sciencedb.18011 (2024).
16. Prashanth, D. S., Mehta, R. V. K., Ramana, K. & Bhaskar, V. Handwritten devanagari character recognition using modified lenet and alexnet convolution neural networks. *Wireless Personal Communications* **122**, 349–378 (2022).
17. James, A., Manjusha, J. & Saravanan, C. Malayalam handwritten character recognition using alexnet based architecture. *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)* **6**, 393–400 (2018).
18. Huang, S., Zhong, Z., Jin, L., Zhang, S. & Wang, H. Dropregion training of inception font network for high-performance chinese font recognition. *Pattern Recognition* **77**, 395–411 (2018).
19. Cheng, S., Shang, G. & Zhang, L. Handwritten digit recognition based on improved vgg16 network. In *Tenth International Conference on Graphics and Image Processing (ICGIP 2018)*, vol. 11069, 954–962 (SPIE, 2019).
20. Cheekati, B. M. & Rajeti, R. S. Telugu handwritten character recognition using deep residual learning. In *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, 788–796 (IEEE, 2020).
21. Mhapsekar, M., Mhapsekar, P., Mhatre, A. & Sawant, V. Implementation of residual network (resnet) for devanagari handwritten character recognition. In *Advanced Computing Technologies and Applications: Proceedings of 2nd International Conference on Advanced Computing Technologies and Applications-ICACTA 2020*, 137–148 (Springer, 2020).
22. Tang, Y. Y., Cheriet, M., Liu, J., Said, J. & Suen, C. Y. Document analysis and recognition by computers. In *Handbook of Pattern Recognition and Computer Vision*, 579–612 (World Scientific, 1999).
23. Wang, M. & Deng, W. A dataset of oracle characters for benchmarking machine learning algorithms. *Scientific Data* **11**, 87 (2024).
24. Pan, Y., Fan, D., Wu, H. & Teng, D. A new dataset for mongolian online handwritten recognition. *Scientific Reports* **13**, 26 (2023).
25. Chen Shanxiong, Z. S. X. L. y. Dual discriminator gan: Restoring ancient yi characters. In *ACM Transactions on Asian and Low-Resource Language Information Processing*, 1–23 (2022).
26. Chen Shanxiong, L. X. L. Y. W. M., Han Xu. Character detection method for yi ancient books based on mser and cnn. In *Journal of South China University of Technology*, 123–133 (2020).

## Author contributions

Xiaojuan Liu and Xu Han co-designed the study, developed the experimental plan and selected the test models, and led the data collection. Shanxiong Chen was responsible for data processing and analysis, Weijia Dai assisted in experimental design and data validation, and Qiuyue Ruan participated in data collation and paper writing. All authors read and approved the final manuscript.

## Competing interests

The authors declare no competing interests.

## Additional information

**Correspondence** and requests for materials should be addressed to S.C.

**Reprints and permissions information** is available at www.nature.com/reprints.

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.