




# Handwritten Kazakh and Russian (HKR) database for text recognition

Daniyar Nurseitov<sup>1,3</sup> · Kairat Bostanbekov<sup>1,3</sup> · Daniyar Kurmankhojayev<sup>2</sup> · Anel Alimova<sup>1,3</sup> · Abdelrahman Abdallah<sup>1,3</sup>  · Rassul Tolegenov<sup>3</sup>

Received: 25 November 2020 / Revised: 29 July 2021 / Accepted: 2 August 2021 /  
Published online: 13 August 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

## Abstract

In this paper, we introduce a large scale dataset, called HKR, to address challenging detection and recognition problems of handwritten Russian and Kazakh text in the scanned documents. We present a new Russian and Kazakh database (with about 95% of Russian and 5% of Kazakh words/sentences respectively) for offline handwriting recognition. A few pre-processing and segmentation procedures have been developed together with the database. The database is written in Cyrillic and shares the same 33 characters. Besides these characters, the Kazakh alphabet also contains 9 additional specific characters. This dataset is a collection of forms. The sources of all the forms in the datasets were generated by LaTeX which subsequently was filled out by persons with their handwriting. The database consists of more than 1500 filled forms. There are approximately 63000 sentences, more than 715699 symbols produced by approximately 200 different writers. It can serve researchers in the field of handwriting recognition tasks by using deep and machine learning. For experiments, we used several popular text recognition methods for word and line recognition like CTC-based and attention-based methods. The results indicate the diversity of HKR. The dataset is available at [https://github.com/abdoelsayed2016/HKR\\_Dataset](https://github.com/abdoelsayed2016/HKR_Dataset).

**Keywords** Document analysis and recognition · Handwritten Russian and Kazakh text recognition · Benchmark dataset

## 1 Introduction

Today, handwriting recognition is a very urgent task. A solution to this problem would automate the business processes of many companies. One of the clear examples is a postal company, where the task of sorting a large volume of letters and parcels is an acute issue.

---

✉ Abdelrahman Abdallah  
abdoelsayed2016@gmail.com

<sup>1</sup> KazMunayGas Engineering LLP, Nur-Sultan, Kazakhstan

<sup>2</sup> Hong Kong Polytechnic University, Hung Hom, Hong Kong

<sup>3</sup> Satbayev University Almaty, Almaty, Kazakhstan

Many researchers have made different types of handwritten text recognition systems for different languages such as English [19, 35, 59], Chinese [54, 60], Arabic [43], Japanese [14], Bangla [8], Malyalam [31], etc. Having said that, the recognition problems of these scripts cannot be considered to be entirely solved.

Any language contains a large number of words. For example, dictionaries of the Russian and Kazakh languages on average register more than 100,000 words, and the Oxford English dictionary more than 300,000 words. In this regard, collecting an exhaustive database of handwritten words, which include all words with a large variation in handwriting, seems almost impossible. In other words, there is always a word that the system cannot recognize. To the best of our knowledge, the analogs of handwritten text database for Russian and Kazakh languages do not exist. To create such a database, we decided to adopt the general principles of data collection and storage described in the IAM Database [39]. In the context of handwritten address recognition, it is necessary to identify the many keywords that can occur in the address.

In this paper, we describe the first version of a database that contains Russian words and also present a new database for offline handwriting recognition. The collection of this database combines the following steps. As an initial step, we collected the first data set with our own hands, since it is almost impossible to find such a set publicly available. This dataset was obtained by using forms, which consisted of machine-typed texts, and empty lines next to those texts. These empty lines were subsequently filled out by persons with their handwriting. It can serve as a basis for a variety of handwriting recognition tasks. Next, the same way we collected handwritten Kazakh and Russian alphabet in Cyrillic. The last set of data came from handwritten samples of poems also filled by our own hands in Russian language. Overall, the databases were produced by approximately 200 different writers, each having 5 to 10 forms (made up of poem and keyword texts) to fill.

For these purposes, we determined the minimum set of words, which includes all the names of cities, towns, villages, districts, and streets in Kazakhstan, and created layouts for filling out forms. Forms were created in such a way as to simplify the process of “cutting” words from the form as much as possible (Fig. 1). Extensive experiments related to the pre-processing of forms were also carried out in order to automatically identify forms, determine the contours of forms, compensate rotations, and also remove edge artifacts at the boundaries of segmented words.


To solve the problem of recognition and processing of natural languages (natural language processing), which consists of optical recognition of characters of the manuscript texts in Russian and Kazakh languages, innovative software is being developed using state-of-the-art neural network-based machine learning methods.

The following section defines the related work on Handwriting Databases. Section 3 presents the Data collection and storage phases as one of the most time consuming and costly stages. Section 4 provides Automated Labeling and Words Segmentation. Section 5 provides further characteristics of the Database. Section 6 provides Experiment Result on the HKR dataset and conclusion and future work are given in Sect. 7.

## 2 Related work

The IAM Handwriting Database [39, 40] comprises handwritten samples in English that can be used to evaluate systems like text segmentation, handwriting recognition, writer identification and writer verification. The database is developed on the Lancaster-Oslo/

**Fig. 1** One of the poem form in the dataset. The database consists of more than 1500 filled forms

Олжас	<i>Олжас</i>	Судейменов	<i>Судейменов</i>	
Валчата	<i>Валчата</i>	1961	<i>1961</i>	

Шёл человек.	<i>Шёл человек.</i>	Шёл степью,	<i>Шёл степью,</i>
долго, долго.	<i>долго, рано</i>	Куда? Зачем?	<i>Куда? Зачем?</i>
Нам это	<i>Нам что</i>	не узнать.	<i>не узнать</i>
В густой ли- стве.	<i>В густой листве</i>	он увидел волка.	<i>он увидел волка,</i>
Первая, вос- ходи.	<i>Первой, востану</i>	А, точно,	<i>А, точнее, мать...</i>
Она лежала в	<i>Она лежала</i>	зрелых полей,	<i>зрелых полей</i>
Откинул лав- ку.	<i>Откинул лавку</i>	и осканил часть.	<i>и осканил часть</i>
Из горла	<i>Из горла</i>	перекочевавшего плеска	<i>перекочевавшего плеска</i>
Толчками крови,	<i>Толчками крови,</i>	густая, словно грязь.	<i>густая, словно грязь</i>
Кем? Кем? Возмож?	<i>Кем? Кем? Возмож?</i>	Одотичивши плем?	<i>Одотичивши плем?</i>
Слезами во- лчатым	<i>Слезами волчатым</i>	это не узнать.	<i>это не узнать.</i>
Они, томлясь	<i>Они, томлясь</i>	и форма, соса- ли	<i>и форма, сосали</i>
Большую	<i>Большую</i>	неподвижную мать.	<i>неподвижную мать.</i>
Полднем вол- чать	<i>Полднем волчать</i>	повыли,	<i>повыли,</i>
Как известно падают	<i>Как известно падают</i>	и заросли узелю.	<i>и заросли узелю.</i>
Они, прожа- вшись	<i>Они, прожавшись</i>	к маме, ждали плви	<i>к маме, ждали плви</i>
Густую напо- деющую	<i>Густую наподеющую</i>	кровь.	<i>кровь.</i>
С глазами в них	<i>С глазами в них</i>	вышла жа- да мести.	<i>вышла жагда мести.</i>
Кому? Любо- му?	<i>Кому? Любому?</i>	Личь бы не простить.	<i>Личь бы не простить.</i>
И будет мстить	<i>И будет мстить</i>	В отдален- сти.	<i>В отдаленности,</i>
Не вместе.	<i>Не вместе.</i>	А встретится—	<i>А встретится.</i>
Друг другу	<i>Друг другу</i>	будут мстить.	<i>будут мстить.</i>
И человек по- шёл	<i>И человек пошёл</i>	своей дорогой.	<i>своей дорогой.</i>
Куда? За- чем? За- чем?	<i>Куда? Зачем? Зачем?</i>	Нам это не узнать.	<i>Нам это не узнать</i>
Он был вос- матчик.	<i>Он был восматчик.</i>	Но волчат не трошу.	<i>Но волчат не трошу.</i>
Робот уже не	<i>Робот уже не</i>	защитила мать.	<i>защитила мать.</i>

Bergen Corpus and comprises forms where the contributors copied a given text in their natural unconstrained handwriting. Each form was subsequently scanned at 300 dpi and saved as a gray level (8-bit) PNG image. The IAM Handwriting Database 3.0 includes contributions from 657 writers, making a total of 1539 handwritten pages comprising 5685 sentences, 13,353 text lines, and 115,320 words. The database is labeled at the sentence, line, and word levels. It has been widely used in word spotting [18, 21, 57, 58], writer identification [7, 11, 13, 30, 51], handwriting gender prediction [37, 38], handwritten text segmentation [46, 47, 61] and offline handwriting recognition [12, 16, 22, 26].

RIMES [24] is a representative database of an industrial application. The main idea of developing this database was to collect handwritten samples similar to those that are sent to different companies by postal mail and fax by individuals. Each contributor was assigned a fictitious identity and a maximum of up to five different scenarios from a set of nine themes. These themes included real-world scenarios like damage declaration or modification of contract. The subjects were required to compose a letter for a given scenario using their own words and layout on white paper using black ink. A total of 1300 volunteers contributed to data collection, providing 12,723 pages corresponding to 5605 mails. Each mail contains two to three pages, including the letter written by the contributor, a form with information about the letter, and an optional fax sheet. The pages were scanned, and the complete database was annotated to support evaluation of tasks like document layout

analysis [41], mail classification [32], handwriting recognition [25] and writer recognition [51].

The National Institute of Standards and Technology, NIST, developed a series of databases [23] of handwritten characters and digits supporting tasks like isolation of fields, detection and removal of boxes in forms, character segmentation, and recognition. The form comprises boxes containing writer information, 28 boxes for numbers and 2 for alphabets, and 1 box for a paragraph of text. The NIST Special Database 1 comprised samples contributed by 2100 writers. The latest version of the database, the Special Database 19, comprises handwritten forms of 3600 writers with 810,000 isolated character images along with ground truth information. This database has been widely employed in a variety of handwritten digits [27] and character recognition systems [52].

CVL [33] is a database of handwritten samples supporting handwriting recognition, word spotting, and writer recognition. The database consists of seven different handwritten texts, one in German and six in English. A total of 310 volunteers contributed to data collection, with 27 authors producing 7 and 283 writers providing 5 pages each. The ground truth data is available in XML format, which includes a transcription of the text, the bounding box of each word, and the identity of the writer. The database has been used for writer recognition and retrieval [17] and can also be employed for other recognition tasks. In addition to regular text, a database of handwritten digit strings written by 303 students has also been compiled [15]. Each writer provided 26 different digit strings of different lengths, making a total of 7800 samples. Isolated digits were extracted from the database to form a separate dataset — the CVL Single Digit Dataset. The Single Digit Dataset comprises 3578 samples for each of the digit class (0-9). A subset of this database has also been used in the ICDAR 2013 digit recognition competition [15].

The AHDB [4] is an offline database of Arabic handwriting together with several pre-processing procedures. It contains Arabic handwritten paragraphs and words. Words used to represent numbers on checks produced by 100 different writers. The database was mainly intended to support automatic processing of bank checks, but it also contains pages of unconstrained texts allowing evaluation of generic Arabic handwriting recognition systems as well. The database was employed in handwriting recognition [5] and writer identification tasks [3].

IFN/ENIT [44] is an database of handwritten Arabic town/village names is presented in this paper. 411 writers filled out forms with approximately 26400 names totaling over 210000 characters. It's made for training and evaluating handwritten Arabic word recognition systems. There are 26459 handwritten Tunisian town/village names in the IFN/ENIT database.

CASIA [34] This dataset includes collections of isolated characters and handwritten texts from online and offline Chinese handwriting databases. A total of 1,020 authors contributed to the samples. Isolated character datasets, whether online or offline, contain approximately 3.9 million samples of 7,356 groups (7,185 Chinese characters and 171 symbols), while handwritten text datasets contain approximately 5,090 pages and 1.35 million character samples. Each dataset is divided into standard training and test subsets and segmented and annotated at the character level. Various handwritten text review activities can be researched using the online and offline databases. For Chinese handwriting, Shusen Zhou proposed [60] First, utilizing digital ink techniques, the client end samples and redisplay handwritten text, segments handwritten characters, change them, and stores original handwritten information into a self-defined document. Second, using the suggested Gabor feature extraction and affinity propagation clustering (GFAP) approach, the server recognizes handwritten documents and delivers the recognition results to the client.

### 3 Data collection and storage

#### 3.1 Data collection

A data collection phase is one of the most time consuming and costly stages. Our main task is to simplify and automate as much as possible. The sources of all the forms in the datasets were generated by LaTeX, then converted to PDF and printed to be filled by writers. So, it was an easy task to generate the correct labels for the printed text on the forms. Each writer filled approximately between 5-10 forms from keyword and poem forms, so each form in the dataset is written by approximately 50-100 writers. Each form has a unique id at the name of the form. The word or letter is placed in the rectangle. The filled forms and letters were scanned with a Canon MF4400 Series UFR II scanner at a resolution of 300 dpi and a color depth of 24 bits.

We collected three different Datasets described as the following:

- Handwritten samples (Forms) of keywords in Kazakh and Russian (Areas, Cities, Village, etc.) are shown in Fig. 2.
- Handwritten Kazakh and Russian alphabet in Cyrillic are shown in Fig. 2.
- Handwritten samples (Forms) of poems in Russian are shown in Fig. 1.

##### 3.1.1 Keyword database

To begin with, we consider correspondence addresses relevant for the Republic of Kazakhstan, as the list of keywords containing the following names:

- Areas
- Cities
- Village
- Settlements
- Streets
- Poems
- Russian Letter

Additional information, such as:

- Indices
- Phones
- Surnames
- Company Names

were not included in the database.

##### 3.1.2 Handwritten alphabet and forms

There are two fundamental approaches to text recognition: character recognition (Optical Character Recognition, OCR) and word recognition (Optical Word Recognition, OWR). With OCR, a model dataset required to train the model should contain handwritten samples





of all the characters in a language alphabet. It is important for each language to compose separate forms since the set of letters of different alphabets can vary greatly. On the other hand, with OWR, a model dataset required to train the model should contain handwritten samples of all the Words for the language. Further, for subsequent training and testing of the model, handwritten samples of target words are needed. An example of one of the forms for collecting word samples and letters (Fig. 2).

### 3.1.3 Data collection methods

A person who has agreed to provide a sample of his handwriting will fill the forms and give the form to us and we scan and save it in our database.

## 4 Automated labeling and words segmentation

### 4.1 Automated labeling

Labeled data are data that have been marked with labels identifying certain features, characteristics, or a kind of object. The labeling of data is a prerequisite for recognition experiments. Labeling data is expensive, time consuming, and error prone. Like in “IAM Dataset” [39], we decided to do as much automation as possible automatically. The sources of all the forms printed (and subsequently filled by writers) were saved on a text file with a unique id for the form and the cell number in the form. So it was an easy task to generate the correct labels for the printed text on the forms. In this regard, we have developed a recommendation system that allows us to simplify the process of labeling data in forms.

### 4.2 Segmentation

The form is designed so that it is possible to easily identify and segment by cells. To identify the form, there is a marker in the upper-right corner of each form. To simplify the process of segmentation, the entire form is divided by horizontal and vertical lines, which makes it quite easy to restore the structure of the document, and accordingly, the spatial position of the word. Words are indexed (annotated) according to their position in the table. In order to cut out cells from the form, the following actions (pre-processing) are performed:

- filtering forms to enhance table boundaries
- defining of the contours of the table
- Determination and compensation of the angle of rotation
- exclusion of lines

**Fig. 3** Example of a region cut out of a form with a word. A pronounced cell line and a piece of letter from a neighboring area are visible along the edges

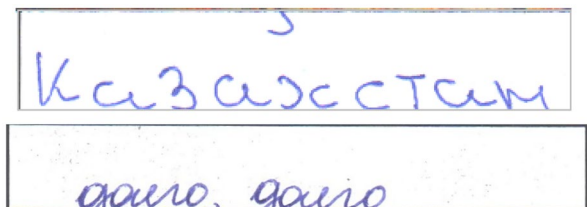
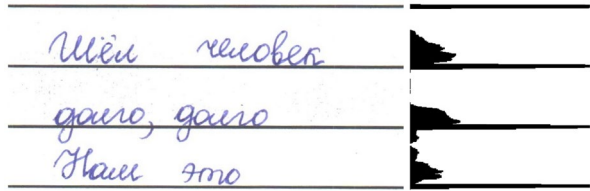


Fig. 4 Horizontal histogram



- Sorting forms by id (marker)
- Streaming the division of forms into words
- name and storage of words

After the image areas corresponding to the word cells are segmented, segmented image areas corresponding to the word cells may contain some edge artifacts. For example, line artifacts cut out with a cell or parts of a word from a neighboring cell can be attributed to artifacts (Fig. 3). We eliminate these artifacts by constructing vertical and horizontal histograms (Figs. 4 and 5) also by cutting off parts that are separately localized closer to the edges of the cell.

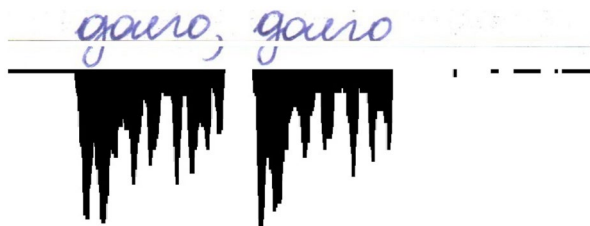
However, it is not always possible to eliminate all artifacts. The following are some aspects that make further processing of a segmented word difficult:

- Letters may not be interconnected.
- Letters can be perfected with artifacts.
- The position of the letters and their size vary significantly from word to word.
- Letters can be written in different colors (blue, black, red).

In this regard, we have developed a recommendation system that allows us to simplify the process of selecting areas with words from the form.

- We suggest filling out the form using the blue pen. This will allow the system to distinguish the word from the table borders at the color level. For example, by converting an image from RGB to HSV, we get a color representation of objects that is invariant with respect to lighting. In this color space, blue remains blue, regardless of the brightness and intensity of the image.
- sometimes eliminating parts of words from neighboring cells is impossible without distorting the target content of a given cell; therefore, when filling out the form, it is desirable that the subject does not go beyond the boundaries of the cell.
- find the region of interest (ROI) in forms, the ROI in our forms is two columns that are filled by writers.
- We segmented the cells depending on the horizontal white space between the cell by using the histogram (Fig. 4).

Fig. 5 Vertical histogram







Training (70%), Validation (15%), and Testing (15%). The test dataset was equally split into two sub-datasets (7.5% each): the first dataset was named TEST1 and consisted of words that did not exist in Training and Validation datasets; the second was named TEST2 and was made up of words that exist in Training dataset but with totally different handwriting styles. The primary purpose of splitting the Test dataset into TEST1 and TEST2 was to check the differences between accuracies of recognizing totally unseen words and the other words, which were seen in the training phase but with unseen handwriting styles. After training, validation, and testing datasets were prepared, the models were trained, and a series of comparative evaluation experiments were conducted. As experiment results proved, the Attention-Gated-CNN-BGRU model demonstrated the best performance with 8.34% character error rate (CER) Levenshtein, 38.54% word error rate (WER) and 12.12% CER our algorithm for the first test dataset and 8.36% CER Levenshtein, 56.36% WER and 16.5% CER our algorithm for the second test dataset.

## 6.1 Evaluation methods

In this article, we evaluated models using two methods: in the first method the standard performance measures are used for all results presented: the character error rate (CER) and word error rate (WER) [20]. The CER is determined as the Levenshtein distance, which is the sum of the character substitution ( $S$ ), insertion ( $I$ ), and deletions ( $D$ ) required to turn one string into another, divided by the total number of characters in the ground truth word ( $N$ ).

$$CER = \frac{S+I+D}{N} \quad (1)$$

Similarly, the WER is calculated as the sum of the number of the term substitutions ( $S_w$ ), insertion ( $I_w$ ), and deletions ( $D_w$ ), which is necessary for the transformation of one string into another, and divided by the total number of ground-truth terms ( $N_w$ ).

$$WER = \frac{S_w+I_w+D_w}{N_w} \quad (2)$$

The second method is character error rate (CER) [10] which was developed by the authors to evaluate our results. CER by our algorithm for counting each character's errors goes in a loop through all the results, then counts the frequency of characters and the number of correctly recognized characters. The error for each character is calculated by

$$CER_c = (1 - \frac{pred_c}{freq_c}) * 100 \quad (3)$$

where  $c$  is a character,  $pred_c$  is the number of correct predictions of  $c$  and  $freq_c$  is the number of character  $c$ .

Then we calculate the average of CER using the following formula, where errors for each letter are multiplied by the fraction of a character in the whole test dataset and summed.

$$CER_{avg} = \sum CER_c * \frac{freq_c}{total} \quad (4)$$

where  $c$  is a character,  $CER_c$  character error rate of  $c$ ,  $freq_c$  is the number of character  $c$ ,  $total$  is the total number of all characters. In the rest of the paper, we will mention  $CER^*$  for our algorithm and  $CER$  for Levenshtein.

## 6.2 Training

All models have been trained using Tensorflow [1] deep learning library in Python. Tensorflow allows for transparent use of highly optimized mathematical operations on GPUs through Python. A computational graph is defined in the Python script to define all operations that are necessary for the specific computations.

The experiments were run on a machine with 2x “Intel(R) Xeon(R) E-5-2680” CPUs, 4x “NVIDIA Tesla k20x” and 100 GB RAM. The use of a GPU reduced the training time of the models by approximately a factor of 3, however, this speed-up was not closely monitored throughout the project, hence it could have varied.

The plots for the report were generated using the matplotlib library for Python, and the illustrations have been created using Inkscape, which is a vector graphics software similar to Adobe Photoshop.

All models are trained to minimize validation loss value. The optimization with stochastic gradient descent is performed, using the RMSProp method [29] with a base learning rate of 0.001 and mini-batches of 32. Also, early stopping with patience 20 is applied, we wanted to monitor the validation loss at each epoch, and when the validation loss does not improve after 20 epochs, training is interrupted.

## 6.3 SimpleHTR model

Originally inspired by artificial neural network architectures by [50] and [56], Harald Scheidl proposed a new approach to handwritten recognition task [48] in 2018. The model’s architecture consists of 5 convolutional neural network (CNN) layers, 2 long short term memory (LSTM) layers, connectionist temporal classification (CTC) loss and decoder layers shown in Fig. 8.

Below is a SimpleHTR algorithm pipeline in short [49]:

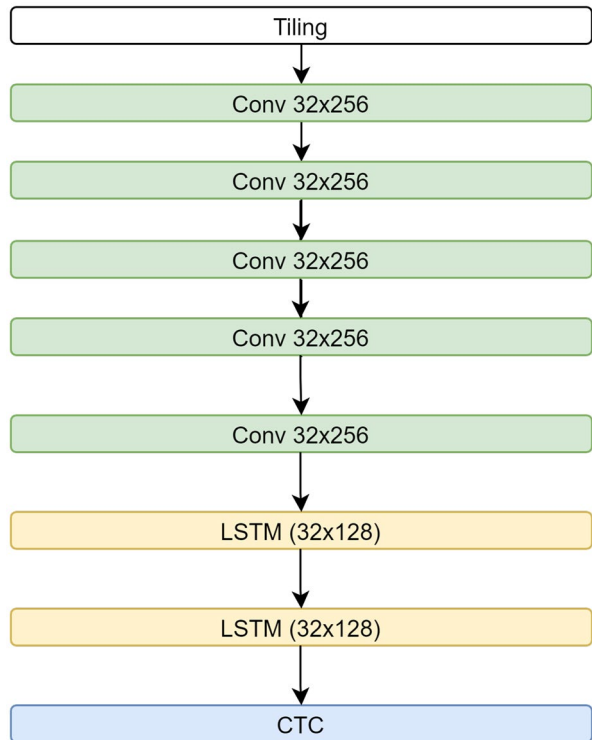
- Input is a gray-scale image of fixed size 128 x 32 (W x H)
- CNN layers map this gray-scale image to a feature sequence of size 32 x 256
- LSTM layers with 256 units map this feature sequence to a matrix of size 32 x 80: here 32 represents the number of time-steps (horizontal positions) in an image with a word; 80 represents probabilities of different characters at a certain time-step in that image)
- CTC layer may work in 2 modes: loss mode - to learn to predict the right character at a time-step when training; decoder mode - to get the recognized word when testing
- batch size is equal to 50

SimpleHTR model ideally requires handwritten texts to be split into words; otherwise, recognition of a full text line would result in low accuracy since 32 time-steps are insufficient to handle a large number of characters in that text line.

## 6.4 LineHTR model

LineHTR model [36] is just an extension of the previous SimpleHTR model, which was developed to enable the model to process images with a full text line (not a single word only), thus, to increase the model’s accuracy further. Architecture of LineHTR model is quite similar to that of SimpleHTR model, with some differences in the

**Fig. 8** SimpleHTR architecture contain 5 CNN layers as an encoder with 2 LSTM layers to decode the feature and pass it to the CTC loss function



number of CNN and LSTM layers and size of those layers' input: it has 7 CNN and 2 Bidirectional LSTM (BLSTM) layers. Below is a LineHTR algorithm pipeline in short [36]:

- Input is a gray-scale image of fixed size 800 x 64 (W x H)
- CNN layers map this gray-scale image to a feature sequence of size 100 x 512
- BLSTM layers with 512 units map this feature sequence to a matrix of size 100 x 205: here 100 represents the number of time-steps (horizontal positions) in an image with a text line; 205 represents probabilities of different characters at a certain time-step in that image)
- CTC layer may work in 2 modes: loss mode - to learn to predict the right character at a time-step when training; decode mode - to get the final recognized text line when testing

## 6.5 Nomeroff net OCR model

According to the authors of Nomeroff Net automatic number-plate recognition system [42], the OCR architecture solution is shown in Fig. 9.

As can be seen from the Fig. 9, the Nomeroff Net OCR algorithm pipeline is as follows:

- Input is a gray-scale image of fixed size 64 x 128 (W x H)
- This gray-scale image is then introduced into 2 subsequent CNN layers, which in turn output feature maps of (16 x 32) x 16 size
- Then these feature maps are reshaped into a single map of 256 x 32 size
- then this map is introduced into a fully connected (FC) layer
- the output from FC layer is directed to 2 parallel recurrent neural network (RNN) layers of gated recurrent unit (GRU)
- outputs from 2 GRU layers are combined into one by Element-wise addition operation to form a map of 512 x 32 size
- then this map is directed to 2 parallel GRU RNN layers again
- outputs from the last 2 GRU layers are concatenated to form a map of 1024 x 32 size
- then this map is introduced into subsequent FC and Softmax layers, before being directed to CTC decoder to obtain the final recognized text

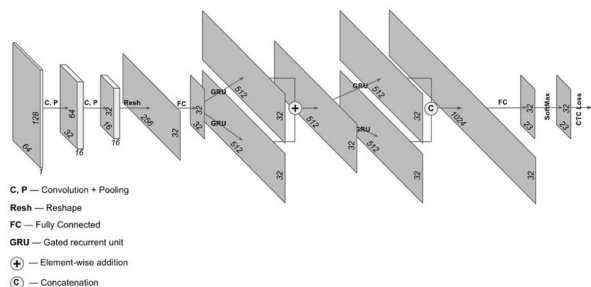
Although Nomeroff Net OCR architecture was designed to recognize “machine typed” car numbers, it is also worth to check the model’s performance on handwritten text recognition tasks. That is why this model is also included in the list of RNN architectures evaluated in this research work.

### 6.6 Bluche model

Bluche model [9] proposes a new neural network structure for modern handwriting text recognition (HTR) as an opportunity for RNNs in multidimensional LSTM. The model is totally based on a deep convolutional input image encoder and a bi-directional LSTM decoder predicting sequences of characters. Its goal is to generate standard, multi-lingual and reusable tasks in this paradigm using the convolutional encoder to leverage more records for transfer learning.

The encoder in the Bluche model contains 3x3 convolutional layer with 8 features, 2x4 convolutional layer with 16 features, a 3x3 gated convolutional layer, 3x3 convolutional layer with 32 features, 3x3 gated convolutional layer, 2x4 convolutional layer with 64 features and 3x3 convolutional layer with 128 features. The decoder contains 2 bidirectional LSTM layers of 128 units and 128 dense layers between the LSTM layers. Figure 10 shows the Bluche architecture.

**Fig. 9** Nomeroff’s number plate recognition model architecture



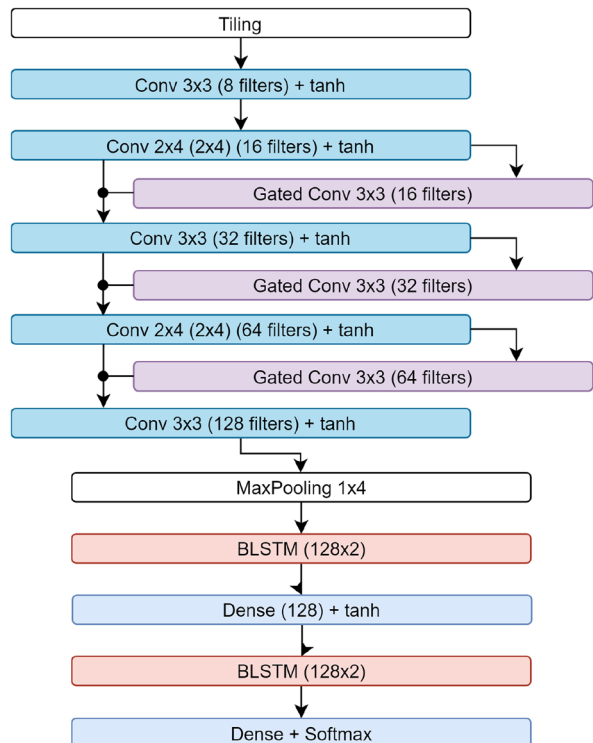
## 6.7 Puigserver model

Modern approaches of Puigserver model [45] to offline HTR dramatically depend on 1-LSTM and 2-LSTM networks. Puigserver model has a high level of recognition rate and a large number of parameters (around 9.6 million). This implies that LSTM dependencies, theoretically modeled by recurrent layers, might not be sufficient, at least in the lower layers of the system, to achieve high recognition accuracy. Figure 11 shows the Puigserver architecture.

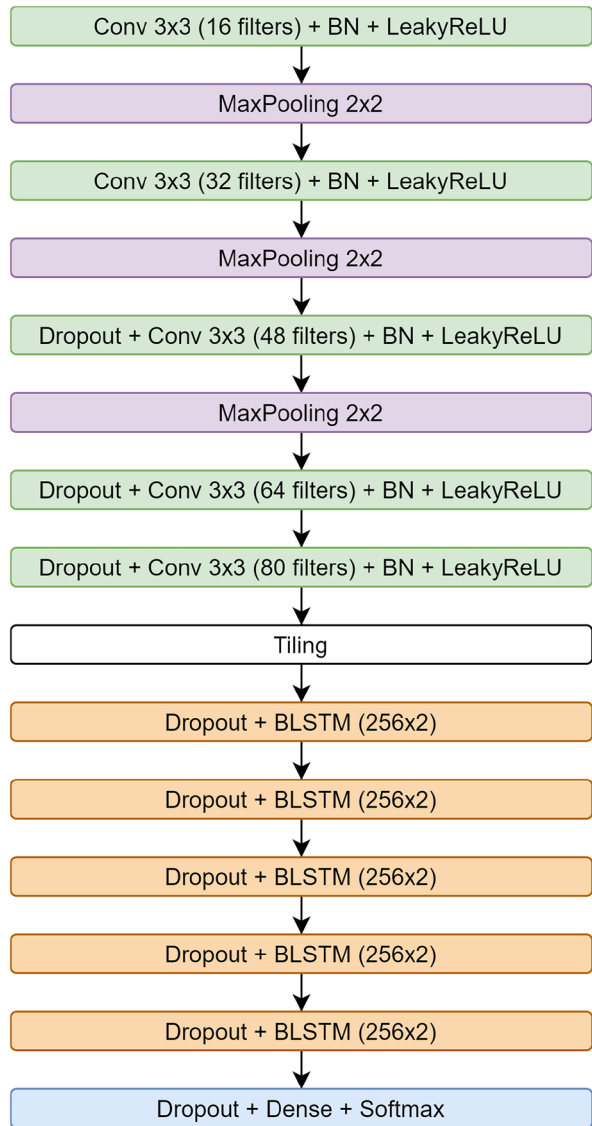
The Puigserver model has three important parts :

- Convolutional blocks: they include 2-D convolutional layer with 3x3 kernel size and 1 horizontal and vertical stride. number of filters is equal to  $16n$  at the  $n$ -th layer of Conv.
- Recurrent blocks: Bidirectional 1D-LSTM layers form recurrent blocks, that transfer the input image column-wise from left to right and from right to left. The output of the two directions is concatenated depth-wise.
- Linear layer: the output of recurrent 1D-LSTM blocks are fed to linear layer to predict the output label. Dropout is implemented before the Linear layer to prevent overfitting (also with probability 0.5).

**Fig. 10** Bluche HTR model contain 8 CNN and 2 BLSTM layers



**Fig. 11** Puigcerver HTR model contain 5 CNN and 5 BLSTM layers



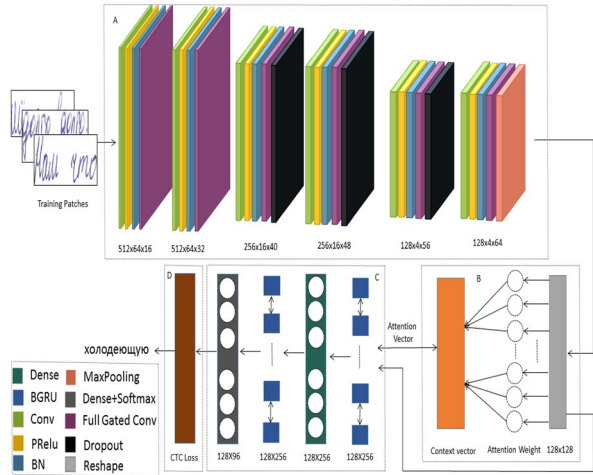
### 6.8 Attention-Gated-CNN-BGRU model

Attention-based Fully Gated CNN-BGRU [2], aiming at improving HTR model accuracy in handwritten Cyrillic text recognition task. The architecture is shown in Fig. 12.

This model’s architecture consists of 4 main parts: encoder, attention, decoder, and CTC. An encoder part consists of 5 convolutional blocks, each of which is made up of a convolutional layer, Parametric Rectified Linear Unit (PReLU) activator [28] with Batch Normalization, and gated convolutional layer [9]. The Dropout technique is also applied at the input of some convolutional layers (with a dropout probability of 0.5) to reduce the



**Fig. 12** Attention-Gated-CNN-BGRU architecture for handwriting recognition. The system contains four main parts: (A) encoder, (B) attention block, (C) decoder, (D) CTC

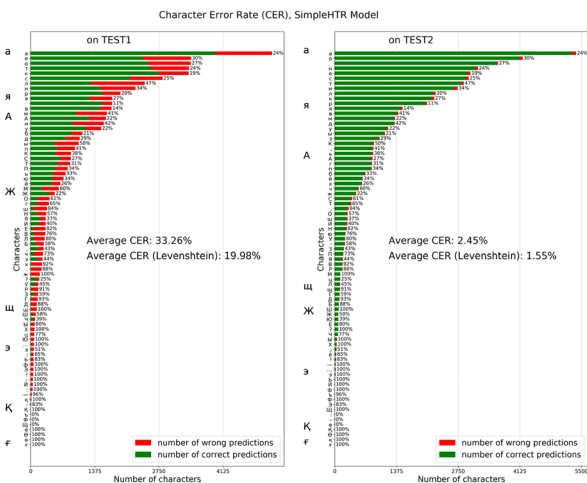


overfitting issue [53]. As an attention part of this model’s architecture, Bahdanau attention mechanism is used [6]. Generally, attention mechanisms encode an input sentence by segmenting it into a fixed number of parts so they can be processed later by a decoder. Bahdanau attention mechanism enabled attention mechanism to focus on relevant parts of an input sentence, rather than hard segmenting it. The key role of Bahdanau attention mechanism applied between an encoder and decoder is to provide a richer encoding of the input sequence.

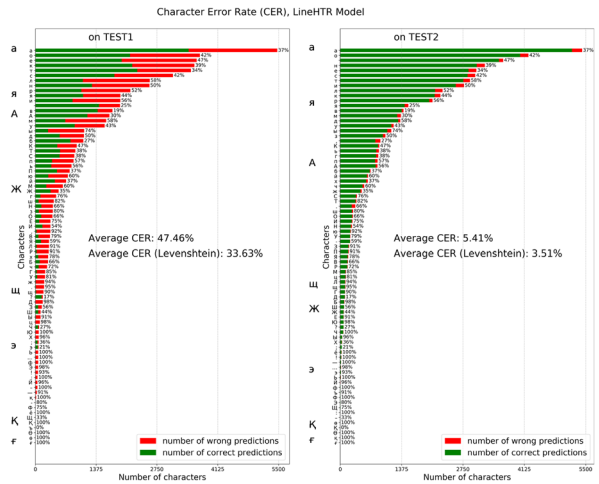
### 6.9 Results

All the models were trained on the HKR dataset. We evaluated these models by the standard performance measures used for all results presented: CER, CER\*, and WER. For all models, the minibatch of 32 size and Early Stopping after 20 epochs without improvement in validation loss value and lr=0.001 were set. For the best use of each model, within the

**Fig. 13** SimpleHTR model performance on TEST1 and TEST2 datasets



**Fig. 14** LineHTR model performance on TEST1 and TEST2 datasets

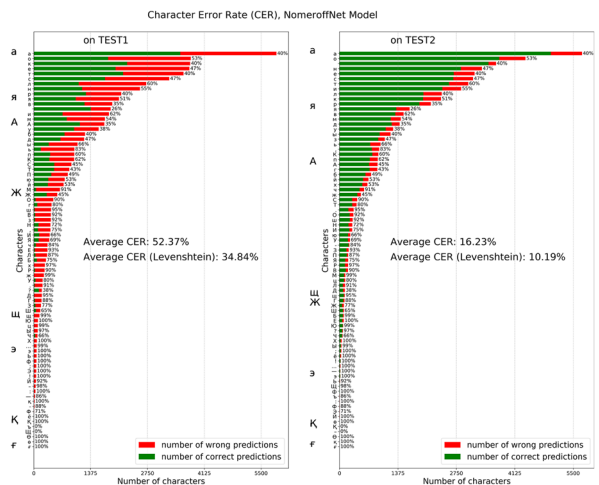


20 tolerance epochs, ReduceLRonPlateau schedule [55] with a decay factor of 0.2 after 10 epochs without improvement in validation loss value was also used. All of the following figures present the character error rate, which shows how the model detects each character.

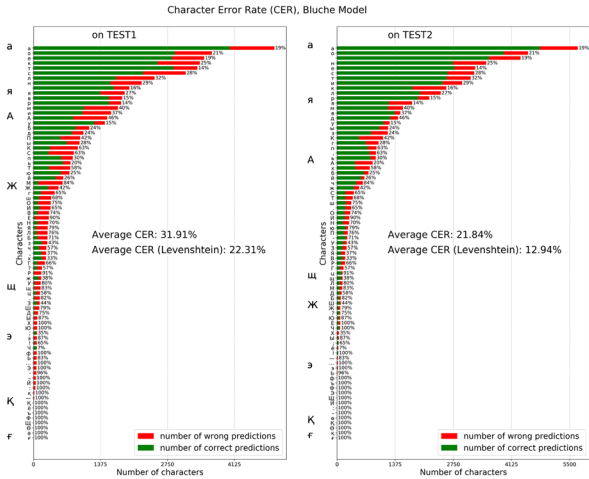
The first experiment was conducted with SimpleHTR model which showed a performance with average of 58.97% WER, 33.26 CER\*, and 19.98% CER on TEST1 and 11.09% WER, 2.45 CER\*, and 1.55% CER on TEST2 datasets (Fig. 13). This big difference in CER rates shows that SimpleHTR model overfitted to words seen in the training stage and demonstrated a lower level of generalization.

The next experiment was carried out with LineHTR model which was trained on data for 100 epochs. This model demonstrated a performance with average 85.66% WER, 47.46 CER\*, and 33.63% CER on TEST1 and 21.99% WER, 5.41 CER\*, and 3.51% CER on TEST2 datasets (Fig. 14). A similar tendency of overfitting to training data can be observed here as well.

**Fig. 15** NomeroffNet HTR model performance on TEST1 and TEST2 datasets



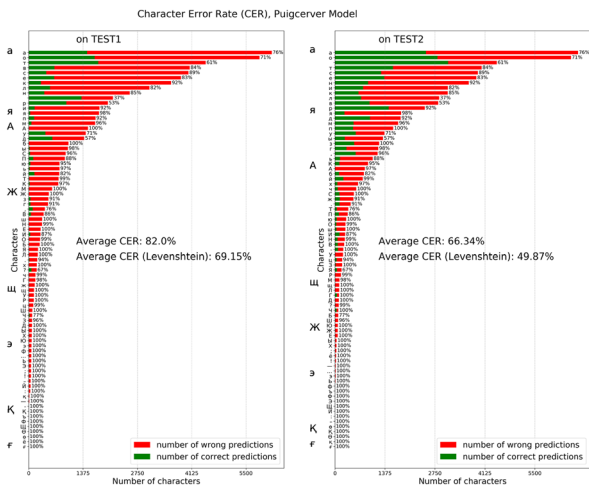
**Fig. 16** Bluche HTR model performance on TEST1 and TEST2 dataset



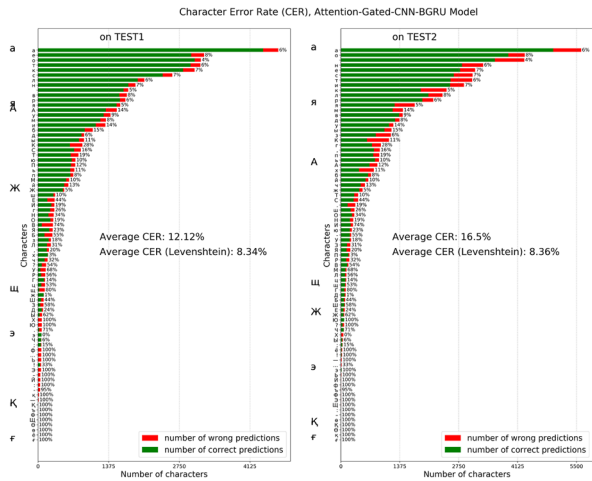
The same experiments were conducted with Nomeroff Net HTR model. Unlike previous models examined, this model showed a lower average of 80.28% WER, 52.37 CER\*, and 34.84% CER on TEST1 and 50.19% WER, 16.23 CER\*, and 10.19% CER on TEST2 datasets (Fig. 15). As can be observed from the figure, NomeroffNet model also suffers from overfitting.

The experiments on HTR were also conducted with Puigcerver and Bluche models and the following results for recognition errors were obtained for the test datasets: 1) The Bluche model achieved 76.43% WER, 31.91 CER\*, and 22.31% CER in the TEST1 dataset and of 69.13% WER, 21.84 CER\*, and 12.94% CER in the TEST2 dataset (Fig. 16); 2) The Puigcerver HTR model showed 100.00% WER, 82.00 CER\*, and 69.15% CER in the TEST1 dataset and of 98.95% WER, 66.34 CER\*, and 49.87% CER in the TEST2 dataset (Fig. 17). We can observe that the Puigcerver has a higher error rate compared with the other models because the Puigcerver model has many parameters ( 9.6M) and underfitting on the dataset.

**Fig. 17** Puigcerver HTR model performance on TEST1 and TEST2 dataset



**Fig. 18** Attention-based Fully Gated CNN-BGRU model performance on TEST1 and TEST2 datasets



Training of Attention-Gated-CNN-BGRU model took 240 epochs. The CER rates on TEST1 and TEST2 datasets were reported as 8.34% and 8.36% respectively (Fig. 18). As can be seen from the figure, Attention-based Fully Gated CNN-BGRU model resulted in lower CER and generalization rates overall.

Table 1 shows the result of comparison between all models.

### 7 Conclusion and future work

In this research work, firstly, we have built the handwritten Kazakh, Russian database. The database can serve as a basis for research in handwriting recognition. This contains Russian Words (Areas, Cities, Village, Settlements, Areas, Streets) by a hundred different writers. It also incorporates the most popular words in the Republic of Kazakhstan. A few pre-processing and segmentation procedures have been developed together with the database. Finally, it contains free handwriting forms in any area of the writer’s interest. This database is meant to provide a training and testing set for Kazakh, Russian Words recognition

**Table 1** CER, CER\*, and WER for Bluche, Puigcerver, NomeroffNet, LineHTR, SimpleHTR, and Attention-Gated-CNN-BGRU

Model	TEST1			TEST2		
	WER	CER*	CER	WER	CER*	CER
Puigcerver	100.00%	82.00%	69.15%	98.95%	66.34%	49.87%
Bluche	76.43%	31.91%	22.31%	69.13%	21.84%	12.94%
NomeroffNet	80.28%	52.37%	34.84%	50.19%	16.23%	10.19%
LineHTR,	85.66%	47.46%	33.63%	21.99%	5.41%	3.51%
SimpleHTR	58.97%	33.26%	19.98%	11.09%	2.45%	1.55%
Attention-Gated-CNN-BGRU	38.54%	12.12%	8.34%	56.36%	16.5%	8.36%

research. In the future, further work on gathering Handwriting samples of keywords and envelope shots will continue. At the same time, envelopes are annotated and various metrics are checked to evaluate the recognition error. In order for the artifacts to not interfere, we need to collect as much tagged data as possible.

Secondly, this research work tried to solve a handwritten Cyrillic postal-address interpretation task using well-known RNN models, such as SimpleHTR, LineHTR, NomeroffNet, Bluche, Puigcerver, and Attention-Gated-CNN-BGRU HTR models. These RNN models were first quantitatively evaluated against each other to select the best performing one. According to experiments, the Attention-Gated-CNN-BGRU HTR model demonstrated the highest recognition rate overall.

One of this research work's goals was to investigate and quantitatively compare the state-of-the-art RNN models to choose the best performing one in a handwritten Cyrillic postal-address recognition task. This goal also incorporates all efforts put into improving the best performing RNN model. According to experiment results, the Attention-Gated-CNN-BGRU HTR model demonstrated comparatively better results in terms of generalization and overall accuracy (see Table 1). This model was then extended to the modified version, called Attention-based Fully Gated CNN-BGRU model.

As figures 13-18 show, generally average CERs of all models tend to be high. It seems the reason for that is large differences between frequencies of Cyrillic characters. In other words, since the dataset includes a small number of Kazakh language handwritings, the language characters have lower frequencies (distribution in the dataset) compared to other Cyrillic letters. Consequently, above mentioned models struggle to recognize these characters resulting in very low recognition rates. Hence, this affects the overall average CER. The dataset also includes non-alphabetic characters (such as “.,!” and so on) with small distributions. SimpleHTR, LineHTR, and NomeroffNet models seemed to overfitting while being trained in Cyrillic handwritings. It seems that enriching the dataset with a variety of Kazakh and Russian words, and making it balanced will solve this issue.

Generally, all models examined in this research proved that there is need for more data, especially containing Kazakh language words. As future work, a Telegram bot was created to collect a new dataset with predominantly Kazakh language words. Currently, the handwriting recognition model developed by this research is not ready to use at a production level, like in a postal company. The development of a web application that enables easy-to-use interface for users is still being developed.

**Acknowledgements** We would like to thank the following people for helping with this research project: Maksat Kanatov and Kuanysh Slyamkhan. This work was funded by the Ministry of Education and Science of the Republic of Kazakhstan (Grant No AP05135175)

## References

1. Abadi M, Agarwal A, Barham P, Brevdo E et al (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. CoRR. <http://arxiv.org/abs/1603.04467>
2. Abdallah A, Hamada M, Nurseitov D (2020) Attention-based fully gated CNN-BGRU for Russian handwritten text. *J Imaging* 6(12), 141. <http://dx.doi.org/10.3390/jimaging6120141>
3. Al-ma'adeed S (2012) Text-dependent writer identification for Arabic handwriting. *J Electr Comput Eng*. <https://doi.org/10.1155/2012/794106>
4. Al-ma'adeed S, Elliman D, Higgins C (2002) A data base for Arabic handwritten text recognition research. In: *Int Arab J Info Technol* vol. 1, pp. 485–489. IEEE. <https://doi.org/10.1109/IWFHR.2002.1030957>

5. Al-ma'adeed S, Higgins C, Elliman D (2004) Off-line recognition of handwritten Arabic words using multiple hidden Markov models. In: The Twenty-third SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence vol. 17, pp. 75–79. <https://doi.org/10.1016/j.knosys.2004.03.002>
6. Bahdanau D, Cho K, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on Learning Representations, San Diego, CA, USA, May 7–9, 2015. Conference Track Proceedings. <http://arxiv.org/abs/1409.0473>
7. Bensefia A, Paquet T, Heutte L (2005) A writer identification and verification system. *Pattern Recogn Lett* 26(13):2080–2092. <https://doi.org/10.1016/j.patrec.2005.03.024>
8. Bhattacharya U, Shridhar M, Parui S, Sen P, Chaudhuri B (2012) Offline recognition of handwritten Bangla characters: An efficient two-stage approach. *Pattern Anal Applic* 15(4):445–458. <https://doi.org/10.1007/s10044-012-0278-6>
9. Bluche T, Messina R (2017) Gated convolutional recurrent neural networks for multilingual handwriting recognition. In: 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 646–651. IEEE. 10.1109/ICDAR.2017.111
10. Bostanbekov K, Tolegenov R (2020) Character error rate (cer) method. <https://github.com/abdoelsayed2016/CAR>
11. Bulacu M, Schomaker L (2007) Text-independent writer identification and verification using textual and allographic features. *IEEE Trans Pattern Anal Mach Intell* 29(4):701–717. <https://doi.org/10.1109/TPAMI.2007.1009>
12. Bunke H, Bengio S, Vinciarelli A (2004) Offline recognition of unconstrained handwritten texts using HMMs and statistical language models. *IEEE Trans Pattern Anal Mach Intell* 26(6):709–720. <https://doi.org/10.1109/TPAMI.2004.14>
13. Daniels Z, Baird H (2013) Discriminating features for writer identification. In: 12th International Conference on Document Analysis and Recognition, pp. 1385–1389. IEEE. <https://doi.org/10.1109/ICDAR.2013.280>
14. Das S, Banerjee S (2014) An algorithm for japanese character recognition. *Int J Image Graph Signal Process* 7(1):9–15. <https://doi.org/10.5815/ijgisp.2015.01.02>
15. Diem M, Fiel S, Garz A, Keglevic M, Kleber F, Sablatnig R (2013) ICDAR 2013 competition on handwritten digit recognition (HDRC 2013). In: 12th International Conference on Document Analysis and Recognition pp. 1422–1427. IEEE. <https://doi.org/10.1109/ICDAR.2013.287>
16. Dreuw P, Doetsch P, Plahl C, Ney H (2011) Hierarchical hybrid MLP/HMM or rather MLP features for a discriminatively trained Gaussian HMM: A comparison for offline handwriting recognition. In: 18th IEEE Int Conf Image Process pp. 3541–3544. IEEE. <https://doi.org/10.1109/ICIP.2011.6116480>
17. Fiel S, Sablatnig R (2013) Writer identification and writer retrieval using the fisher vector on visual vocabularies. In: 12th International Conference on Document Analysis and Recognition pp. 545–549. IEEE. <https://doi.org/10.1109/ICDAR.2013.114>
18. Fischer A, Keller A, Frinken V, Bunke H (2012) Lexicon-free handwritten word spotting using character HMMs. *Pattern Recognit Lett* 33(7):934–942. <https://doi.org/10.1016/j.patrec.2011.09.009>
19. Fischer A, Suen C, Frinken V, Riesen K, Bunke H (2013) A fast matching algorithm for graph-based handwriting recognition. In: International Workshop on Graph-Based Representations in Pattern Recognition pp. 194–203. Springer. [https://doi.org/10.1007/978-3-642-38221-5\\_21](https://doi.org/10.1007/978-3-642-38221-5_21)
20. Frinken V, Bunke H (2014) Continuous Handwritten Script Recognition, pp. 391–425. Springer London, London. [https://doi.org/10.1007/978-0-85729-859-1\\_12](https://doi.org/10.1007/978-0-85729-859-1_12)
21. Frinken V, Fischer A, Manmatha R, Bunke H (2011) A novel word spotting method based on recurrent neural networks. *IEEE Trans Pattern Anal Mach Intell* 34(2):211–224. <https://doi.org/10.1109/TPAMI.2011.113>
22. Gatos B, Pratikakis I, Perantonis S (2006) Hybrid off-line cursive handwriting word recognition. In: 18th Int Conf Pattern Recognit vol. 2, pp. 998–1002. IEEE. <https://doi.org/10.1109/ICPR.2006.644>
23. Geist JC, Wilkinson R, Janet S, Grother PJ, Hammond B, Larsen NW, Klear R, Matsko MJ, Burges CJ, Creecy R et al (1994) The second census optical character recognition systems conference. Tech. rep. National Institute of Standards and Technology
24. Grosicki E, Carr M, Geoffrois E, Prteux F (2006) RIMES evaluation campaign for handwritten mail processing. In: Proceedings of the Tenth International Workshop on Frontiers in Handwriting Recognition, pp. 231–235
25. Guichard L, Toselli AH, Coüasnon B (2010) Handwritten word verification by SVM-based hypotheses re-scoring and multiple thresholds rejection. In: 12th International Conference on Frontiers in Handwriting Recognition pp. 57–62. IEEE. <https://doi.org/10.1109/ICFHR.2010.15>
26. Gnter S, Bunke H (2003) Ensembles of classifiers for handwritten word recognition. *Int J Doc Anal Recognit* 5(4):224–232. <https://doi.org/10.1007/s10032-002-0088-2>

27. Ha TM, Bunke H (1997) Off-line, handwritten numeral recognition by perturbation method. *IEEE Trans Pattern Anal Mach Intell* 19(5):535–539. <https://doi.org/10.1109/34.589216>
28. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: *Proc IEEE Int Conf Comput Vis* pp. 1026–1034
29. Hinton G, Srivastava N, Swersky K (2012) Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. Cited on 14(8)
30. Jain R, Doermann D (2011) Offline writer identification using K-adjacent segments. In: *International Conference on Document Analysis and Recognition*, pp. 769–773. IEEE. <https://doi.org/10.1109/ICDAR.2011.159>
31. John J, Balakrishnan K, Pramod V (2013) A system for offline recognition of handwritten characters in malayalam script. *Int J Image Graph Signal Process* 5:53–59. <https://doi.org/10.5815/ijgisp.2013.04.07>
32. Kermorvant C, Louradour J (2010) Handwritten mail classification experiments with the RIMES database. In: *12th International Conference on Frontiers in Handwriting Recognition*, pp. 241–246. IEEE. <https://doi.org/10.1109/ICFHR.2010.45>
33. Kleber F, Fiel S, Diem M, Sablatnig R (2013) CVL-database: An off-line database for writer retrieval, writer identification and word spotting. In: *12th International Conference on Document Analysis and Recognition*, pp. 560–564. IEEE. <https://doi.org/10.1109/ICDAR.2013.117>
34. Liu CL, Yin F, Wang DH, Wang QF (2011) Casia online and offline chinese handwriting databases. In: *2011 International Conference on Document Analysis and Recognition* pp. 37–41. IEEE
35. Liu H, Ding X (2005) Handwritten character recognition using gradient feature and quadratic classifier with multiple discrimination schemes. In: *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pp. 19–23. <https://doi.org/10.1109/ICDAR.2005.123>
36. Lcm HT (2018) Line-level handwritten text recognition with tensorflow. <https://github.com/lamhoangtung/LineHTR>. Last accessed 11 May 2020
37. Maken P, Gupta A (2021) A method for automatic classification of gender based on text-independent handwriting. *Multimed Tools Appl* pp. 1–30
38. Maken P, Gupta A, Gupta MK (2019) A study on various techniques involved in gender prediction system: a comprehensive review. *Cybern Inf Technol* 19(2):51–73
39. Marti UV, Bunke H (1999) A full English sentence database for off-line handwriting recognition. In: *Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR'99 (Cat. No. PR00318)*, pp. 705–708. IEEE. <https://doi.org/10.1109/ICDAR.1999.791885>
40. Marti UV, Bunke H (2002) The IAM-database: An English sentence database for offline handwriting recognition. *Int J Doc Anal Recognit* 5(1):39–46. <https://doi.org/10.1007/s100320200071>
41. Montreuil F, Grosicki E, Heutte L, Nicolas S (2009) Unconstrained handwritten document layout extraction using 2D conditional random fields. In: *10th International Conference on Document Analysis and Recognition*, pp. 853–857. IEEE. <https://doi.org/10.1109/ICDAR.2009.132>
42. Net N (2020) Nomeroff net. automatic numberplate recognition system. version 0.3.1. <https://nomeroff.net.ua/>. Last accessed 11 May 2020
43. Parvez M, Mahmoud S (2013) Arabic handwriting recognition using structural and syntactic pattern attributes. *Pattern Recognit* 46(1):141–154. <https://doi.org/10.1016/j.patcog.2012.07.012>
44. Pechwitz M, Maddouri SS, Märgner V, Ellouze N, Amiri H et al (2002) Ifn/enit-database of handwritten arabic words. In: *Proc. of CIFED vol. 2*, pp. 127–136. Citeseer
45. Puigcerver J (2017) Are multidimensional recurrent layers really necessary for handwritten text recognition? In: *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 67–72. IEEE
46. Salvi D, Zhou J, Waggoner J, Wang S (2013) Handwritten text segmentation using average longest path algorithm. In: *Proceedings of IEEE Workshop on Applications of Computer Vision*, pp. 505–512. IEEE. <https://doi.org/10.1109/WACV.2013.6475061>
47. Santos R, Clemente G, Ing Ren T, Cavalcanti G (2009) Text line segmentation based on morphology and histogram projection. In: *10th International Conference on Document Analysis and Recognition*, pp. 651–655. IEEE. <https://doi.org/10.1109/ICDAR.2009.183>
48. Scheidl H (2018) Handwritten text recognition in historical documents. Technische Universität Wien
49. Scheidl H (2018) Handwritten text recognition with tensorflow. <https://github.com/githubharald/SimpleHTR>. Last accessed 11 May 2020
50. Shi B, Bai X, Yao C (2016) 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* 39(11):2298–2304
51. Siddiqi I, Vincent N (2010) Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features. *Pattern Recognit* 43(11):3853–3865. <https://doi.org/10.1016/j.patcog.2010.05.019>



52. Smith SJ, Bourgoin MO, Sims K, Voorhees HL (1994) Handwritten character classification using nearest neighbor in large databases. *IEEE Trans Pattern Anal Mach Intell* 16(9):915–919. <https://doi.org/10.1109/34.310689>
53. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res* 15(1):1929–1958
54. Tao D, Liang L, Jin L, Gao Y (2014) Similar handwritten chinese character recognition by kernel discriminative locality alignment. *Pattern Recognit Lett* 35, 186–194. <https://doi.org/10.1016/j.patrec.2012.06.014>. *Frontiers in Handwriting Processing*
55. Vinciarelli A, Luettin J (2001) A new normalization technique for cursive handwritten words. *Pattern Recognit Lett* 22(9):1043–1050
56. Voigtlaender P, Doetsch P, Ney H (2016) Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In: *15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 228–233. IEEE
57. Wshah S, Kumar G, Govindaraju V (2012) Multilingual word spotting in offline handwritten documents. In: *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pp. 310–313. IEEE
58. Wshah S, Kumar G, Govindaraju V (2012) Script independent word spotting in offline handwritten documents based on hidden Markov models. In: *International Conference on Frontiers in Handwriting Recognition*, pp. 14–19. IEEE <https://doi.org/10.1109/ICFHR.2012.264>
59. Zamora-Martinez F, Frinken V, Espana-Boquera S, Castro-Bleda MJ, Fischer A, Bunke H (2014) Neural network language models for off-line handwriting recognition. *Pattern Recognit* 47(4):1642–1652. <https://doi.org/10.1016/j.patcog.2013.10.020>
60. Zhou S, Chen Q, Wang X (2014) Handwritten chinese text editing and recognition system. *Multimed Tools Appl* 71(3):1363–1380
61. Zimmermann M, Bunke H (2002) Automatic segmentation of the IAM off-line database for handwritten English text. In: *Proc Int Conf Pattern Recognit vol. 4*, pp. 35–39. IEEE. <https://doi.org/10.1109/ICPR.2002.1047394>

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