Handwritten text generation and strikethrough characters augmentation

A.V. Shonenkov [1], D.K. Karachev [2], M.Y. Novopoltsev [1], M.S. Potanin [1,3], D.V. Dimitrov [1,4], A.V. Chertok [1,5]
[1] SBER AI, 117312, Moscow, Russia, ul. Vavilova, 19;
[2] OCRV, 107078, Moscow, Russia, Kalanchevskaia, 13;
[3] MIPT, 141701, Moscow Region, Russia, Dolgoprudny, Institutskiy per., 9;
[4] Lomonosov MSU, 119991, Moscow, Russia, GSP-1, Leninskie Gory;
[5] AIRI, Moscow, Russia, Nizhny Susalny lane, 5, p. 19

Abstract

We introduce two data augmentation techniques, which, used with a Resnet - BiLSTM - CTC network, significantly reduce Word Error Rate and Character Error Rate beyond best-reported results on handwriting text recognition tasks. We apply a novel augmentation that simulates strikethrough text (HandWritten Blots) and a handwritten text generation method based on printed text (StackMix), which proved to be very effective in handwriting text recognition tasks. StackMix uses weakly-supervised framework to get character boundaries. Because these data augmentation techniques are independent of the network used, they could also be applied to enhance the performance of other networks and approaches to handwriting text recognition. Extensive experiments on ten handwritten text datasets show that HandWritten Blots augmentation and StackMix significantly improve the quality of handwriting text recognition models.

Keywords: data augmentation, handwritten text recognition, strikethrough text, computer vision, StackMix, handwritten blots.

Citation: Shonenkov AV, Karachev DK, Novopoltsev MY, Potanin MS, Dimitrov DV, Chertok AV. Handwritten text generation and strikethrough characters augmentation. Computer Optics 2022; 46(3): 455-464. DOI: 10.18287/2412-6179-CO-1049.

Introduction

Handwriting text recognition (HTR) is a vital task. Automation allows for a dramatic reduction in labor costs for processing correspondence and application forms and deciphering historical manuscripts. The main problem with historical documents is the usual small amount of labeled data. However, HTR systems require many examples for training and setting parameters. Optical character recognition (OCR) is an area in which deep learning has proven itself perfectly. The situation in HTR, especially with historical documents, which are much worse. Since there are only a few open datasets, the quality of trained models recognition is much lower.

To improve the state-of-the-art of HTR system, we introduce two ways to increase the volume of training data: augmentation that simulates strikethrough text - HandWritten Blots and a "new text" generation method - StackMix.

The proposed HandWritten Blots algorithm simulates the strikethrough characters as close as possible to the originals. It can change the inclination, size, and transparency of drawing lines that strike out characters. This is illustrated in fig. 1.

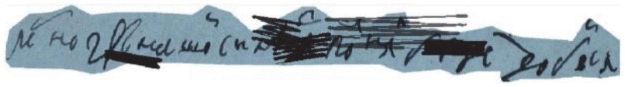
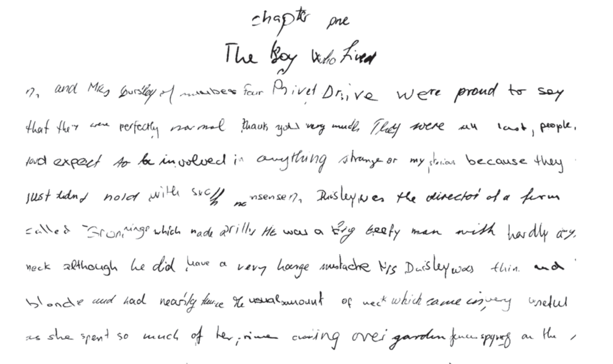


Fig. 1. Sample handwritten text image after using Hand-Written Blots

StackMix generates handwritten text (phrase, string, or entire page) using characters images of the training dataset. We proposed weakly-supervised learning to extract characters boundaries from training images. As an example, we generated pages of texts from different sources. Some paragraphs from the first chapter of a Harry Potter book were generated from IAM datasets style (fig.2). The results suggest that it is possible to generate different texts with different styles and fonts.

These augmentations were designed initially for the system to decipher Peter the Great manuscripts that were first introduced at [1] by using marked-up lines of the text as input.



chapter one
The Boy Who Lived

r, and Mrs Dursley of number four Privet Drive were proud to say
that they were perfectly normal thank you very much, They were the last, people,
you'd expect to be involved in anything strange or mysterious because they
just didn't hold with such nonsense, Dursley was the director of a firm
called Grunnings which made drills He was a big beefy man with hardly any
neck although he did have a very large mustache Mrs Dursley was thin and
blonde and had nearly twice the usual amount of neck which came in very useful
as she spent so much of her time craning over garden fences spying on the
neighbors The Dursleys had a small son called Dudley and in their opinion

Fig. 2. Example first page of Harry Potter created using StackMix from IAM style

We evaluate our augmentation and normalization techniques using a Resnet - BiLSTM - CTC architecture to perform HTR. Our approach achieves low Word Error Rate (WER) and Character Error Rate (CER) on ten different handwritten texts dataset.

Related work

Data augmentation

Data augmentation consists of augmenting the training set with synthetically generated samples. It reduces the tendency to overfit when training models with many parameters and limited labeled data. In data augmentation for image classification problems, the training set is increased by modifying the original images through transformations such as scaling, rotation, or flipping images and generating new images from part of the images of the training dataset.

For example, in the CutMix approach [2], parts of the images are cut from different samples and inserted into a new one. At the same time, the targets are mixed according to the proportions of the original parts of the images. A similar approach is used in SnapMix [3], but the cutting of images is regulated using Class Activation Map. This approach allows for reducing the noise of the cut objects and selecting the most significant parts. Additionally, an exciting approach with mixing objects is presented in MixUp [4] and MWH [5], where images overlap each other with a transparency coefficient and mix the targets with the proportion. Unfortunately, these methods cannot be applied to the optical character recognition and handwritten text recognition tasks because mixing recursively dependent targets makes it very difficult to obtain a correct mapping of image and text. Therefore, we would like to introduce the StackMix approach, which improves the quality and stability of our neural network.

Several authors have proposed specific augmentation techniques for HTR. In [6], the authors introduce a more robust augmentation technique and normalization to model the handwritten text variation of a given author. In [7], they show some affine transformation methods for data augmentation in HTR. [8] and [9] synthesize new lines images by concatenating characters from different datasets. In [8], the authors introduce a new method for matching double-sided historical documents to remove interference caused by handwriting from the reverse side due to ink sipping over long storage periods. This article proposes different strategies for obtaining synthetic handwritten Chinese documents using an existing segmented database at the character level. In [10], the authors improve the performance by augmenting the training set with specially crafted multiscale data.

Common tricks may significantly improve the quality of HTR models. In [11], the authors investigated data augmentation and transfer learning for small historical datasets.

Obtaining a massive corpus of labeled handwriting images for different languages is a cumbersome task. Authors use Generative Adversarial Networks (GAN) (called ScrabbleGAN) to generate training data [12]. ScrabbleGAN follows a semi-supervised approach to synthesize handwritten text images that are versatile both in style and lexicon. It can generate images of varying lengths. The generator can also manipulate the resulting text style, allowing us to decide whether the text must be cursive or how thick/thin the pen stroke should be.

Handwritten text recognition systems

Handwritten text recognition is a long-standing computer vision problem. Early works on handwritten recognition problems suggest using a combination of Hidden Markov Models (HMM) and recurrent neural networks (RNNs) [13, 14]. The disadvantage of these approaches is the impossibility of end-to-end loss function optimization. With the advent of deep learning came tremendous improvements in handwriting recognition.

Poznaski and Wolf [7] perform word-level recognition by employing a fixed-size CNN architecture that evaluates binary lexical attributes over word images, such as whether a given portion of the image contains a certain unigram or bigram (PHOC [15]). The correct transcription is determined by the word in the lexicon closest to this representation. Authors [16] also employ a fixed-size CNN architecture to learn features for the PHOC representation for embedding the text and images into a common subspace.

Another general approach uses RNNs for HTR. These have been widely adopted with the introduction of Connectionist Temporal Classification (CTC). CTC is an algorithm used to deal with tasks like speech recognition and handwriting recognition. The input data and the output transcription are available, but there are no characters boundaries provided. The basic idea was to interpret the network outputs as a probability distribution over all possible label sequences conditioned on a given input sequence. Given this distribution, an objective function can be derived that directly maximizes the probabilities of the correct labelings.

State-of-the-art HTR architectures combine a convolutional neural network (CNN) with Long Short-Term Memory Recurrent Neural Networks (LSTM) cells [17]. This type of network model can deal with sequential data to identify temporal patterns.

Large Multidimensional Long Short-Term Memory Recurrent Neural Networks (MDLSTM) [18] networks use 2D-RNN, which can deal with both axes of an input image. A simple model consists of several convolutional neural network (CNN) and MDLSTM layers, and using connectionist temporal classification (CTC) loss provides excellent metrics for IAM [19] dataset.

However, MDLSTM models have some disadvantages like high computational costs and instability. In works [20] and [21], the authors try to eliminate recurrent layers in CNN-LSTM-CTC to decrease the number of parameters. Their Gated Fully Convolutional Networks show relatively good results, even without a language model. Another alternative to the RCNN-CTC approach is seq2seq models [22]. The encoder extracts features from the input, and the decoder with an attention mechanism emits the output sequentially. OrigamiNet [23] is a module that combines segmentation and text recognition tasks. It can be added to any line-level handwritten text recognition model to make it page-level.

We compared the proposed model with the models described above. The data for comparison is given in section Results. We also found that different authors used various data partitions for train, test, and validation on the IAM dataset (see section Datasets). This made it difficult to compare the models correctly, so we presented our model results for the same data sets used in the original papers.

Proposed augmentation algorithm

Handwritten blot augmentation

The idea of augmentation appeared during the analysis of the Digital Peter dataset [1, 24]. In the process of examining the dataset, we found examples of images in which some characters were crossed out and almost indistinguishable, but they were still present in the markup. Hence, the idea of using the Cutout augmentation [25] emerged since it allows for overlapping of some elements of symbols or entire symbols, which makes the augmentation a bit like crossed-out symbols.

However, in training the models, we decided to implement such an algorithm that would simulate the strikethrough characters as close as possible to the originals. Since we did not find the implementation of such algorithms in open sources, we created it ourselves. We significantly improve the quality of HTR models using Handwritten Blot augmentation than using Cutout augmentation.

To implement the strikethrough effect, we decided to use the Bezier curve construction algorithm, which in our case smoothed the curve transition between points. The Bezier curve is a parametric curve and is a special case of the Bernstein polynomial. Finding basic polynomials of degree n are found by the formula:

$$b_{j.n} = \binom{n}{j} s^j \times (1-s)^{n-j}. \qquad (1)$$

Where $j = 0,..,n$. The definition of a curve as a linear combination is found as:

$$B(s) = \sum_{j=0}^{n} b_{j,n} \times v_j. \qquad (2)$$

Where v_j the point in the space, and $b_{j,n}$ define above. Since the sum of all polynomials must be equal to one, then.

$$b_{0.n} + b_{1.n} + ... + b_{n.n} = (s + (1-s))^n = 1. \qquad (3)$$

Where S is non-negative weights that sum to one. We found the implementation of the algorithm for constructing the Bezier curve [26].

Next, we implemented our algorithm that simulates strikethrough. A graphical description of the algorithm is shown in fig. 3. The main steps were as follows:

a) Determine the coordinates of the strikethrough area.

b) Define areas for generating points to be used for drawing a Bezier curve.

c) Generate points for the Bezier curve with the intensity parameters of the points and their coordinates to simulate the slope. Sometimes, a random point needs to be used several times for the loop to go slightly further from the curve.

d) Draw a curve with specified transparency.

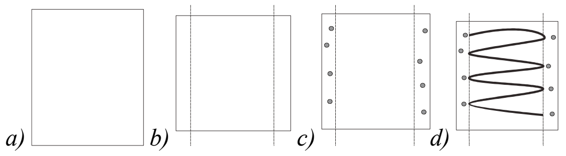


Fig. 3. Graphic description of the algorithm

The implementation of this algorithm can be found here [27]. We empirically selected the parameters for strikethrough (minh = 50, maxh = 100, minw = 10, maxw = 50, incline = 15, intensity = 0.9, transparency = 0.95, count = 1…11, probability = 0.5) and tested it on different datasets. These parameters can change the inclination, transparency, and size of drawing lines that strikeout characters.

The effect of the Blot augmentation on quality metrics is shown in the "blots" row of tab. 5 and fig. 13.

The obtained data suggest that handwritten blot augmentation makes a significant contribution to the quality of training models. Therefore, we recommend using it to train models in handwriting recognition problems.

Handwritten text generation by StackMix algorithm

Proposed Handwritten text generation algorithm StackMix generates synthetic lines of text, given samples of the isolated characters. We used weakly-supervised learning to extract characters boundaries from training images. The algorithm is based on the post-processing of a supervised pretrained neural network via CTC loss. It gets characters boundaries using only weakly-supervised learning without any manual markup (fig. 4). The training does not require character-level annotation and can be applied to cases where only labeled words or entire phrases samples are available. During training, the order of characters is given, and the goal is to separate the input image by these characters, proportional to the size of the output sequence. We use the intersection coordinates of the different characters to get the coordinates of the beginning and end of each character. We can generate any phrase from different datasets style with all characters coordinates in all the lines.

Character width is proportional to k * w / N, where k - number of "cells" with maximum character probability; w - input images width; N - number of samples (fig. 5).

The main idea was to connect the last layer of RNN (after applying SoftMax activation for every symbol from image features) and image width. For training neural network can be used base scheme without any augmentations and tricks. To get high-quality marking of symbols boundaries, a sample from the training stage should be used. Example of the image of symbol segmentation using the weakly-supervised method for IAM dataset is given in fig. 6.

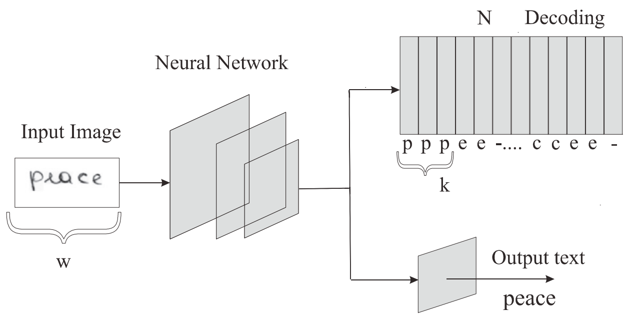


Fig. 4. Post-processing scheme to get the boundaries of symbols. k - number of "cells" with maximum character probability; w - input images width; N - number of samples

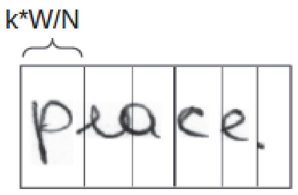


Fig. 5. The boundaries of the symbol

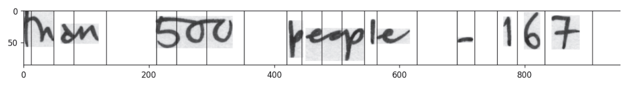


Fig. 6. Example images of symbol segmentation using semi-supervised methods. Images have ids: IAM-a02-082-05

The input for the algorithm is expected to be text from the external corpus, which creates a new image with this text using parts of images of the training dataset. The algorithm from natural language toolkit for python (nltk) [28] MWETokenizer [29] is used for tokenization. It processes tokenized text and merges multi-word expressions into single tokens. Collections of multi-word phrases are obtained from the training dataset, using symbol borders to connect parts of images and MWE tokens, including spaces and punctuation marks. This study used one random MWE tokenizer from six with a token dimension of no more than 3, 4, 5, 6, 7, and 8 with probabilities 0.05, 0.15, 0.2, 0.2, 0.2, and 0.2, respectively. After this, we matched each token with part of an image from the training data. The pieces were stacked (hence the name "StackMix") together into a complete image, maintaining the correct order of the tokens.

The StackMix approach also requires an external text corpus that has the same alphabet as the main dataset. Corpus does not require special marking and only contains allowed symbols. We use different text corpus over thousands of authors and multiple languages written centuries apart corresponding to each dataset in Table 1.

Examples of the StackMix algorithm for various datasets are given in Fig. 8 – 12. Experiments result show that StackMix augmentation improves the quality of training models (Tab. 5, row StackMix). Despite the visible places where tokens were glued together, the algorithm significantly increased the quality of recognition. We tried to increase the realism of the generated text by alignment and selection of samples. However, it did not improve the quality of our experiments.

Tab. 1. Text corpus for different datasets

Dataset	Text Corpus
Bentham, IAM	"Jigsaw Unintended Bias in Toxicity Classification" [30]
Saint Gall	"The Latin Library" [31]
HKR	Russian texts from Wikimedia [32]
Digital Peter	russian texts of the XVII-XVIII centuries

Nevertheless, after specific improvements, this algorithm may be used for the realistic generation of new documents.

Neural network architecture and handwritten texts datasets

Neural network architecture

The neural network underlying the proposed system consists of three parts: a feature generator, RNN to account for the order of the features, and the classifier that outputs the probability of each character (use CTC Loss).

Various network architectures were tested as a feature generator, and the final choice fell on Resnet (fig. 7). We took only three first blocks from Resnet-34 and replaced the stride parameter in the first layer with 1 to increase the "width" of the objects.

After extracted the features, they were averaged through the AdaptiveAvgPool2d layer and fed into the three BiLSTM layers to deal with feature sequences.

As a final classifier, we use two fully connected layers with GELU and dropout between them.

The results achieved using the described architecture without any additional modifications are shown in the "base" row of tab. 5.

Datasets

Ten different datasets were used in the experiments to prove state-of-the-art quality of our model. We have tested our two augmentations techniques over thousands of authors, and multiple languages were written centuries apart.

Bentham manuscripts refers to a large set of documents written by the renowned English philosopher and reformer Jeremy Bentham (1748-1832). Volunteers of the Transcribe Bentham [33] initiative transcribed this collection. Currently, >6000 documents or >25000 pages have been transcribed using this public web platform. We used the BenthamR0 dataset [34], a part of the Bentham manuscripts.

IAM handwriting dataset contains forms of handwritten English text. It consists of 1539 pages of scanned text from 657 different writers. This dataset is widely used for experiments in many papers. However, there is a big problem with uncertainty in data splitting for model training and evaluation. This issue was described in [22]. The IAM dataset has different train/val/test splits that are shown in Table 1. The problem is that none of them are labeled as a standard, so the IAM dataset split differs from paper to paper. However, we should compare results on the same split.

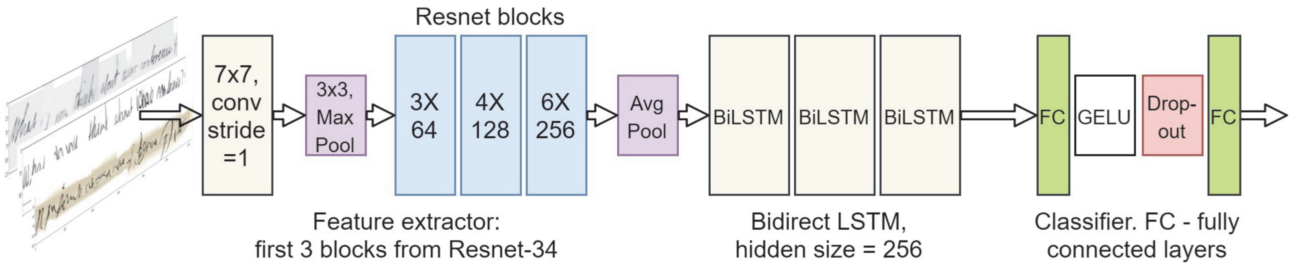

Resnet blocks
7x7, conv stride =1
3x3, Max Pool
3X 64
4X 128
6X 256
Avg Pool
BiLSTM
BiLSTM
BiLSTM
FC
GELU
Drop-out
FC
Feature extractor: first 3 blocks from Resnet-34
Bidirect LSTM, hidden size = 256
Classifier. FC - fully connected layers

Fig. 7. Neural network architecture

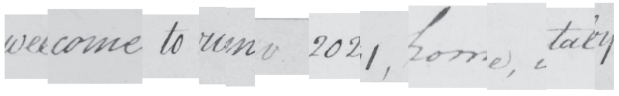


Fig. 8. StackMix example line from the Bentham. "Welcome to ICMV 2021, Rome, Italy"

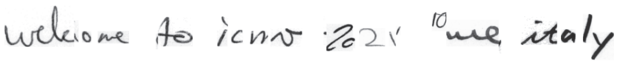


Fig. 9. StackMix example line from the IAM. "Welcome to ICMV 2021, Rome, Italy"

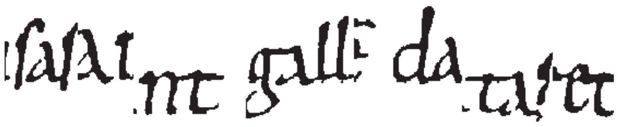


Fig. 10. StackMix example line from the Saint Gall. "Saint Gall"

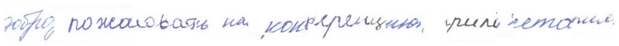


Fig. 11. StackMix example line from the HKR. "Добро пожаловать на конференцию, Рим, Италия"

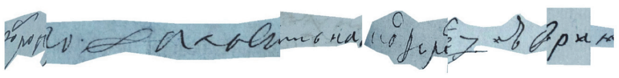


Fig. 12. StackMix example line from the Digital Peter. "Добро пожаловать на конференцию в Рим"

In our experiments, we use the IAM-B [35] and IAM-D partitions. IAM-B was used to compare our model with others. IAM-D was a new partition inspired by the official page of the project [36]. This page contained "unknown","val1" and "val2" split labels. We added "unknown" samples to the train set and combined "val1" and "val2" together.

IAM-B was chosen because many recently published papers used this partition. We used IAM-D because it provides more training samples.

We create a github repository [37] with information and indexes corresponding to each IAM split in Tab. 2. It also contains links to papers that use these splits. We hope this helps researchers choose appropriate IAM partitions and make valid comparisons with other articles.

Saint Gall dataset contains handwritten historical manuscripts written in Latin that date back to the 9th century. It consists of 60 pages, 1410 text lines, and 11597 words.

Tab. 2. IAM splits

Split	Train	Val	Test
IAM-A	6161	966	2915
IAM-B	6482	976	2915
IAM-C	6161	940	1861
IAM-D	9652	1840	1861

HKR [38] is a recently published dataset of modern Russian and Kazakh languages. This database consists of > 1400 filled forms. It contains 64943 lines and > 715699 symbols produced by about 200 different writers. Data are split in the following manner: 45559 lines for a train set, 10009 lines for validation, and 9375 lines for test. We found this data splitting in github [39] of the authors of the HKR_Dataset, but these proportions of train/valid/test were slightly different than those of the original paper [38]. We assumed that the seed was not fixed in a script to get split, which was not a big problem for comparing results. We assumed that the seed was not fixed in a script to get split, which was not a big problem for comparing results.

Digital Peter is an entirelynew dataset of Peter the Great's manuscripts [1]. It consists of 9 694 images and text files corresponding to lines in historical documents. The open machine learning competition Digital Peter was held based on the considered dataset [24]. There are 6 237 lines in the training set, 1 527 lines in the validation set, and 1 930 lines in the testing set.

Specialists of the University of Greifswald created Konzil dataset. It contains manuscripts written in modern German. Train sample consists of 353 lines, validation - 29 lines, and test - 87 lines.

Schiller contains handwritten texts written in modern German. Train sample consists of 244 lines, validation - 21 lines, and test - 63 lines.

Ricordi contains handwritten texts written in Italian. Train sample consists of 295 lines, validation - 19 lines, and test - 69 lines.

Patzig contains handwritten texts written in modern German. Train sample consists of 485 lines, validation - 38 lines, and test - 118 lines.

Schwerin contains handwritten texts written in medieval German. Train sample consists of 793 lines, validation -68 lines, and test -196 lines.

Results and discussions

The four most frequently used in HTR task datasets and Digital Peter dataset were used in the experiments to prove the state-of-the-art quality of our model. Digital Peter is an entirely new dataset [1]. Authors present detailed information about Digital Peter dataset and share links to data.

Proposed augmentations allow for metric improvements. In our experiments, we used StackMix ”on the fly” during training. We added traditional augmentations of CLAHE [40], JpegCompression, Rotate, and our augmentation (simulation of crossed-out letters) - ”HandWritten Blots”. Different combinations of augmentations were grouped in our experiments (Tab. 3,4):

– "Base" - experiments without augmentations, 300 epoch (HKR 100 epoch)
– "Augs" - standart augmentations (CLAHE [40], JpegCompression, Rotate), 300 epoch (HKR 100 epoch).
– "Blots" - using only our HandWrittenBlot augmentation, 500 epoch (HKR 150 epoch).
– "StackMix" - using only our StackMix approach, 1000 epoch (HKR 300 epoch).
– "All" - using all previous augmentations (augs + blots + stackMix), 1000 epoch (HKR 300 epoch).

Models with StackMix were trained during 1000 epoch but were not overfitted. We believe they should be trained more with bigger external text corpora.

Tab. 3. Extended results for all experiments with CER / WER / ACC for valid and test partitions

	BenthamR0					
	Valid CER, %	Valid WER %	Valid ACC %	Test CER %	Test WER %	Test ACC %
base	5.28 ± 0.08	26.1 ± 0.1	24.9 ± 0.3	2.99 ± 0.06	11.8 ± 0.3	52.1 ± 0.8
augs	5.28 ± 0.03	25.9 ± 0.1	25.1 ± 0.4	2.85 ± 0.05	11.3 ± 0.1	52.8 ± 1.0
blots	4.61 ± 0.02	24.4 ± 0.1	26.9 ± 0.3	2.27 ± 0.04	9.5 ± 0.2	57.0 ± 0.4
stackmix	4.51 ± 0.01	24.2 ± 0.1	26.9 ± 0.4	2.08 ± 0.03	9.0 ± 0.1	58.2 ± 0.8
all	**4.20 ± 0.03**	**23.4 ± 0.1**	**28.1 ± 0.5**	**1.73 ± 0.02**	**7.8 ± 0.1**	**61.9 ± 1.1**
	IAM-B					
base	3.67 ± 0.07	13.2 ± 0.2	38.3 ± 0.2	5.80 ± 0.08	18.9 ± 0.2	29.3 ± 0.4
augs	3.50 ± 0.03	12.7 ± 0.2	38.8 ± 0.9	5.43 ± 0.04	17.7 ± 0.1	30.7 ± 0.5
blots	2.93 ± 0.07	10.7 ± 0.3	44.0 ± 0.7	4.59 ± 0.03	15.0 ± 0.1	36.4 ± 0.5
stackmix	3.13 ± 0.03	11.5 ± 0.0	45.4 ± 0.5	4.90 ± 0.07	16.4 ± 0.2	35.2 ± 0.5
all	**2.40 ± 0.05**	**8.9 ± 0.2**	**53.7 ± 1.1**	**3.77 ± 0.06**	**12.8 ± 0.2**	**43.6 ± 0.6**
	IAM-D					
base	3.72 ± 0.03	13.5 ± 0.1	40.3 ± 0.6	4.55 ± 0.06	14.5 ± 0.2	35.5 ± 0.7
augs	3.54 ± 0.03	12.9 ± 0.2	41.9 ± 0.6	4.38 ± 0.04	14.0 ± 0.2	36.3 ± 0.6
blots	2.94 ± 0.09	10.8 ± 0.3	46.8 ± 0.9	3.70 ± 0.03	11.8 ± 0.1	42.3 ± 0.6
stackmix	2.98 ± 0.04	11.1 ± 0.1	48.0 ± 0.3	3.77 ± 0.04	12.3 ± 0.1	42.4 ± 0.8
all	**2.32 ± 0.04**	**8.7 ± 0.1**	**55.1 ± 0.8**	**3.01 ± 0.02**	**9.8 ± 0.1**	**50.7 ± 0.2**
	Saint Gall					
base	4.61 ± 0.08	31.8 ± 0.6	3.0 ± 0.6	4.71 ± 0.05	32.5 ± 0.3	2.2 ± 0.5
augs	4.39 ± 0.09	30.7 ± 0.5	3.8 ± 0.5	4.49 ± 0.11	31.3 ± ± 0.5	3.4 ± 0.7
blots	4.12 ± 0.04	28.6 ± 0.3	5.4 ± 1.1	4.08 ± 0.03	28.1 ± 0.2	4.6 ± 1.1
stackmix	4.11 ± 0.06	29.1 ± 0.4	7.5 ± 0.9	4.06 ± 0.12	28.8 ± 0.8	6.1 ± 0.8
all	**3.73 ± 0.06**	**26.5 ± 0.3**	**12.0 ± 2.5**	**3.65 ± 0.11**	**26.2 ± 0.6**	**11.8 ± 2.0**
	HKR					
base	1.72 ± 0.03	5.8 ± 0.1	91.1 ± 0.1	6.71 ± 0.18	22.5 ± 0.3	71.1 ± 0.5
augs	1.78 ± 0.03	5.8 ± 0.1	91.2 ± 0.1	6.67 ± 0.24	21.5 ± 0.3	72.2 ± 0.1
blots	1.34 ± 0.04	4.1 ± 0.2	93.7 ± 0.3	7.40 ± 0.26	23.0 ± 0.5	72.6 ± 0.4
stackmix	1.72 ± 0.05	6.1 ± 0.1	90.7 ± 0.1	3.69 ± 0.13	14.4 ± 0.4	80.0 ± 0.3
all	**1.32 ± 0.02**	**4.3 ± 0.1**	**93.4 ± 0.1**	**3.49 ± 0.08**	**13.0 ± 0.3**	**82.0 ± 0.5**
	Digital Peter					
base	4.27 ± 0.06	22.5 ± 0.3	47.1 ± 0.3	4.44 ± 0.02	24.3 ± 0.2	43.7 ± 0.4
augs	3.89 ± 0.04	21.2 ± 0.2	49.2 ± 0.5	4.15 ± 0.04	23.0 ± 0.2	45.7 ± 0.4
blots	3.17 ± 0.06	17.3 ± 0.2	56.3 ± 0.7	3.39 ± 0.04	19.3 ± 0.4	51.9 ± 0.4
stackmix	3.19 ± 0.05	17.2 ± 0.2	55.5 ± 0.4	3.40 ± 0.05	19.2 ± 0.3	51.6 ± 0.7
all	**2.42 ± 0.08**	**13.3 ± 0.4**	**64.0 ± 0.8**	**2.50 ± 0.03**	**14.6 ± 0.2**	**60.8 ± 0.8**

A comparison of our results for various datasets (IAM [19], BenthamR0 [32], Digital Peter [1], HKR_Dataset [38], Saint Gall [41]) is presented in tab. 3, 4 and in fig. 13.

The effect of the proposed two augmentations on quality metrics is shown in fig. 13. Train time T_{arb} was measured in arbitrary units and obtained by the formula:

$$T_{arb} = \log_2(T / T_{\min}). \quad (4)$$

Where $T_{min} = 33.6$ ms is the minimum value of train time per one image, the colored lines represent obtained experiment points. Since the quality of approaches grows with increasing train time, some points have no line.

Moreover, we consider 5 more datasets named konzilsprotokolle C (Konzil), Schiller, Ricordi, Patzig and Schwerin. The datasets were written in Italian modern, and medieval German and were first introduced in the ICFHR 2018 Competition over READ dataset to compare the performance of approaches learning with few labeled pages (as all of these datasets are relatively small).

Tab. 4. Extended results for ICFHR 2018 Competition over READ dataset with CER / WER / ACC for valid and test partitions

	Konzil					
	Valid CER, %	Valid WER %	Valid ACC %	Test CER %	Test WER %	Test ACC %
base	9.40 ± 0.34	40.3 ± 2.0	26.9 ± 2.9	10.46 ± 0.48	42.6 ± 1.1	21.8 ± 2.9
augs	8.39 ± 1.29	36.4 ± 3.9	30.3 ± 2.9	10.41 ± 1.49	41.1 ± 3.0	24.4 ± 2.5
blots	5.86 ± 0.47	27.6 ± 1.5	36.5 ± 3.1	7.38 ± 0.70	31.8 ± 2.2	30.3 ± 2.5
stackmix	4.20 ± 0.39	23.9 ± 2.6	41.4 ± 4.2	5.60 ± 0.20	28.7 ± 0.4	32.4 ± 1.5
all	**2.85 ± 0.24**	**16.9 ± 1.0**	**48.3 ± 2.4**	**3.31 ± 0.24**	**17.4 ± 1.4**	**46.7 ± 4.1**
	Ricordi					
base	49.39 ± 12.46	86.3 ± 9.6	2.1 ± 2.9	50.79 ± 12.07	88.6 ± 7.7	0.0 ± 0.0
augs	18.88 ± 3.28	56.9 ± 6.1	2.1 ± 2.9	19.84 ± ± 2.04	58.1 ± 3.3	0.0 ± 0.0
blots	49.77 ± 16.67	84.3 ± 12.1	2.1 ± 2.9	52.59 ± 15.64	88.2 ± 9.8	0.0 ± 0.0
stackmix	10.73 ± 1.55	39.0 ± 4.0	6.3 ± 2.4	12.25 ± 0.90	40.6 ± 2.1	2.3 ± 2.2
all	**9.89 ± 2.75**	**36.6 ± 6.7**	**9.2 ± 5.0**	**11.54 ± 2.00**	**38.4 ± 5.7**	**3.3 ± ± 3.2**
	Schiller					
base	14.60 ± 2.04	45.7 ± 3.9	2.9 ± 2.6	17.31 ± 1.17	58.5 ± 2.5	0.0 ± 0.0
augs	14.51 ± 3.20	44.9 ± 5.7	3.8 ± 4.0	16.35 ± 1.83	54.7 ± 2.9	0.0 ± 0.0
blots	8.98 ± 0.68	32.5 ± ± ± 2.5	13.3 ± ± 4.0	10.98 ± 0.64	41.3 ± 1.5	1.9 ± 1.3
stackmix	6.15 ± 0.29	27.2 ± 1.3	19.1 ± 5.8	8.42 ± 0.23	35.5 ± 1.3	5.4 ± 2.1
all	**3.68 ± 0.39**	**17.5 ± 1.2**	**27.6 ± 2.1**	**5.79 ± 0.31**	**26.1 ± 0.8**	**16.2 ± 2.1**
	Schwerin					
base	7.32 ± 0.00	27.2 ± 0.0	17.6 ± 0.0	8.65 ± 0.00	30.1 ± 0.0	14.3 ± 0.0
augs	16.56 ± 4.87	46.4 ± 12.0	3.2 ± 7.2	17.98 ± 4.47	49.5 ± 9.6	5.1 ± 4.6
blots	2.26 ± 0.00	9.1 ± 0.0	63.2 ± 0.0	3.28 ± 0.00	12.6 ± 0.0	53.6 ± 0.0
stackmix	2.13 ± 0.20	8.6 ± 0.6	61.2 ± 2.2	3.09 ± 0.25	12.8 ± 1.3	52.9 ± 3.4
all	**1.92 ± 0.11**	**7.6 ± 0.3**	**65.0 ± 1.9**	**2.91 ± 0.08**	**12.2 ± 0.3**	**53.5 ± 0.9**
	Patzig					
base	37.15 ± 1.90	77.0 ± 1.9	2.6 ± 0.0	37.92 ± 1.74	79.8 ± 1.4	0.2 ± 0.4
augs	34.76 ± 4.84	75.0 ± 5.4	2.1 ± 1.2	35.32 ± 5.02	76.3 ± 3.9	0.5 ± 0.8
blots	31.25 ± 1.73	71.5 ± 2.1	3.2 ± 1.2	32.70 ± 1.81	73.0 ± 1.8	0.0 ± 0.0
stackmix	14.97 ± 0.86	42.7 ± 1.8	11.6 ± 1.4	13.72 ± 0.61	46.5 ± 1.8	7.3 ± 2.3
all	**12.80 ± 1.25**	**37.3 ± 3.0**	**14. ± 3.4**	**11.34 ± 0.39**	**38.6 ± 0.3**	**13.6 ± 1.0**

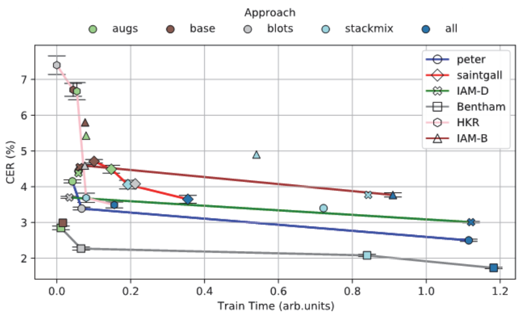


Fig. 13. The relative train time and CER results of experiments for different datasets and approaches

In tab. 4, we provide experimental results for datasets considered above (Konzil, Schiller, Ricordi, Patzig, Schwerin). Despite the small number of samples in each dataset, we succeeded in achieving good metrics. StackMix (and StacMix+Blots+Augmentations which is named "all") leads to significant improvement in recognition quality. This is the main result. StackMix approach helps to generate new training samples, which is vital for few samples datasets. These results show that our model could be applied not only for widespread big datasets but also for less-known small ones. It is enough about twelve translated pages on historical manuscripts to train a good performing model. The "All" approach achieves the best metric value for each dataset. It shows that our augmentations applied to small datasets lead to prosperous and stable training.

Ablation study

In this section, we present the comparison with other models (tab. 5). Our model outperforms other approaches on the BenthamR0, HKR, and IAM-D datasets. It reached 3.77 % of CER on the IAM-B dataset, which is close to the best model published in 2016 that achieved 3.5 % CER. On the Saint Gall dataset, we achieved 5.56 % CER, which is very close to the current best solution of 5.26 % CER.

IAM dataset has two versions because papers use various data splits. We included IAM-B and IAM-D partitions in Table 5 to compare them with other models of the same split.

Authors of the paper [42] provided open code for their model here [43]. We noticed that when evaluating the model, they lowered the characters in predicted and true strings. However, in our experiments, we did not convert the characters to lowercase. For real HTR tasks, it is not essential to track the case of characters.

Tab. 5. Comparison to other models, test set (IAM, Bentham, HKR, Saint Gall, Digital Peter)

Model	IAM-B	
	CER, %	WER, %
[18]	**3.5**	**9.3**
[23]	4.7	–
[45]	4.32	16.24
[20]	7.99	28.61
[46]	7.73	25.22
[47]	6.64	19.6
ours	3.77	12.8
	IAM-D	
[48]	7.8	25.5
ours	**3.01**	**9.8**
	Digital Peter	
[24]	10.5	44.4
ours	**2.50**	**14.6**
	Bentham R0	
[48]	7.1	20.9
[42]	3.98	9.8
ours	**1.73**	**7.9**
	HKR	
[48]	4.5	19.2
ours	**3.49**	**13.0**
	Saint Gall	
[48]	7.25	23.0
[42]	5.26	**21.14**
ours	**3.65**	26.2

We compare our approach on a few labeled datasets (Konzil, Schiller, Ricordi, Patzig, Schwerin) with results from paper [11]. Results are presented in Tab. 6. For our model, we took the best results from Table 4, and the best results from Table 1 in paper [11]. Our model achieves better CER for 3 of 5 datasets. Additionally, we did not use transfer learning from bigger datasets, unlike authors [11].

Tab. 6. Comparison to other models, test CER for ICFHR 2018 Competition over READ dataset

Dataset	Model	
	CNN-BLSTM bidirectional LSTM	ours
Konzil	4.37 [4.24 – 4.54]	**3.31 ± 0.24**
Ricordi	**11.2 [10.11 - 11.23]**	11.54 ± 2.00
Schiller	9.4 [9.31 - 9.46]	**5.79 ± 0.31**
Schwerin	3.5 [3.46 - 3.53]	**2.91 ± 0.08**
Patzig	**10.6 [10.52 - 10.65]**	11.34 ± 0.39

Data that we used for Table 6 are located in "specific_data" folder. Each image and text translation has the index "train [1, 4, 16]" in their file names. So we set "train_1" as a valid index, "train_4" as test index, and "train_16" as train index. We did it this way because "train_1" has the smallest number of samples for each dataset, "train_4" is slightly more extensive, and "train_16" is the largest. So, for the test set in Table 4, we took all files marked with "train_4" for each dataset. But authors of [11] used the test set (with ground truth) for ICFHR 2018 Competition [44], which is hidden for us. Despite the usage of different test sets, we think this comparison is relatively fair.

Conclusion

We have introduced two new data augmentation techniques, strikethrough text algorithm HandWritten Blots and handwritten text generation algorithm StackMix, based on weakly-supervised training. We have demonstrated their use with Resnet - BiLSTM - CTC network to produce the best result among the currently known handwriting recognition systems. These techniques produce the lowest word error rate (WER) and Character Error Rate (CER) to date over hundreds of authors, multiple languages, and thousands of documents, including challenging, medieval, historical documents with noise, ink bleed-through, and faint handwriting.

The presented system can significantly increase the speed of deciphering historical documents. For example, it took a team of 10-15 historians about three months to decipher 662 pages of manuscripts from the Digital Peter dataset. When working on the same dataset on a single Tesla V100, the average decryption speed was 95 lines / s or 380 pages / min, unattainable by historical scientists.

References

[1] Potanin M, Dimitrov D, Shonenkov A, Bataev V, Karachev D, Novopoltsev M. Digital peter: Dataset, competition and handwriting recognition methods. arXiv preprint, 2021. Source: ⟨https://arxiv.org/abs/2103.09354⟩.

[2] Yun S, Han D, Chun S, Oh SJ, Yoo Y, Choe J. CutMix: Regularization strategy to train strong classifiers with localizable features. 2019 IEEE/CVF Int Conf on Computer Vision (ICCV) 2019: 6022-6031.

[3] Huang S, Wang X, Tao D. SnapMix: Semantically proportional mixing for augmenting fine-grained data. Proc AAAI Conf on Artificial Intelligence 2021; 35(2): 1628-1636.

[4] Zhang H, Cisse M, Dauphin YN, Lopez-Paz D. mixup: Beyond empirical risk minimization. Int Conf on Learning Representations 2018.

[5] Yu H, Wang H, Wu J. Mixup without hesitation. arXiv preprint, 2021. Source: ⟨https://arxiv.org/abs/2101.04342⟩.

[6] Wigington C, Stewart S, Davis B, Barrett B, Price B, Cohen S. Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network. 2017 14th IAPR Int Conf on Document Analysis and Recognition (ICDAR) 2017; 1: 639-645.

[7] Poznanski A, Wolf L. Cnn-n-gram for handwriting word recognition. Proc IEEE Conf on Computer Vision and Pattern Recognition 2016: 2305-2314.

[8] Krishnan P, Jawahar C. Matching handwritten document images. Proc European Conf on Computer Vision 2016: 766-782.

[9] Shen X, Messina R. A method of synthesizing handwritten chinese images for data augmentation. 2016 15th Int Conf on Frontiers in Handwriting Recognition (ICFHR) 2015: 114-119.

[10] Chammas E, Mokbel C, Likforman-Sulem L. Handwriting recognition of historical documents with few labeled data. 2018 13th IAPR Int Workshop on Document Analysis Systems (DAS) 2018: 43-48.

[11] Aradillas JC, Murillo-Fuentes JJ, Olmos PM. Boosting offline handwritten text recognition in historical documents with few labeled lines. IEEE Access 2020; 9: 76674-76688.

[12] Fogel S, Averbuch-Elor H, Cohen S, Mazor S, Litman R. Scrabblegan: Semi-supervised varying length handwritten text generation. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition 2020: 4324-4333.

[13] Bengio Y, et al. Markovian models for sequential data. Neural Computing Surveys 1999; 2(199): 129-162.

[14] Bourlard HA, Morgan N. Connnectionist speech recognition: A hybrid approach. Kluwer Academic Publishers; 1994.

[15] Almazán J, Gordo A, Fornés A, Valveny E. Word spotting and recognition with embedded attributes. IEEE Trans Pattern Anal Mach Intell 2014; 36(12): 2552-2566.

[16] Krishnan P, Dutta K, Jawahar C. Deep feature embedding for accurate recognition and retrieval of handwritten text. 15th Int Conf on Frontiers in Handwriting Recognition (ICFHR) 2016: 289-294.

[17] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997; 9(8): 1735-1780.

[18] Voigtlaender P, Doetsch P, Ney H. Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. 15th Int Conf on Frontiers in Handwriting Recognition (ICFHR) 2016: 228-233.

[19] Marti U-V, Bunke H. The IAM-database: an English sentence database for offline handwriting recognition. Int J Doc Anal Recognit 2002; 5(1): 39-46.

[20] Coquenet D, Chatelain C, Paquet T. Recurrence-free unconstrained handwritten text recognition using gated fully convolutional network. 17th Int Conf on Frontiers in Handwriting Recognition (ICFHR) 2020: 19-24.

[21] Ingle RR, Fujii Y, Deselaers T, Baccash J, Popat AC. A scalable handwritten text recognition system. Int Conf on Document Analysis and Recognition (ICDAR) 2019: 17-24.

[22] Michael J, Labahn R, Grüning T, Zöllner J. Evaluating sequence-to-sequence models for handwritten text recognition. Int Conf on Document Analysis and Recognition (ICDAR) 2019: 1286-1293.

[23] Yousef M, Bishop TE. OrigamiNet: Weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. IEEE/CVF Conf on Computer Vision and Pattern Recognition (CVPR) 2020: 14710-14719.

[24] Competition digital peter. 2020. Source: ⟨https://github.com/sberbank-ai/digital_peter_aij2020⟩.

[25] DeVries T, Taylor GW. Improved regularization of convolutional neural networks with cutout. arXiv preprint, 2017. Source: ⟨https://arxiv.org/abs/1708.04552⟩.

[26] Hermes D. Helper for Bézier curves, triangles, and higher order objects. J Open Source Softw 2017; 2(16): 267.

[27] Method implementation (our code). 2021. Source: ⟨https://github.com/TheDenk/augmixations⟩.

[28] Bird S, Loper E, Klein E. Natural language processing with python. O'Reilly Media Inc; 2009.

[29] Malouf R. Multi-word expression tokenizer. Source: ⟨https://www.nltk.org/_modules/nltk/tokenize/mwe.html⟩.

[30] The conversation AI team, T. C. A. Jigsaw unintended bias in toxicity classification. 2018. Source: ⟨https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification⟩.

[31] Credits for the Latin library. Source: ⟨https://www.thelatinlibrary.com/cred.html⟩.

[32] Russian wikimedia downloads. 2021. Source: ⟨https://dumps.wikimedia.org/ruwiki/⟩.

[33] Transcribe Bentham. 2010. Source: ⟨http://transcribe-bentham.ucl.ac.uk/td/TranscribeBentham⟩.

[34] Gatos B, Louloudis G, Causer T, Grint K, Romero V, Sánchez J-A, Toselli A, Vidal E. Ground-truth production in the transcriptorium project. 11th IAPR Int Workshop on Document Analysis Systems 2014: 237-241.

[35] Theodore Bluche. 2002. Source: ⟨http://www.tbluche.com/resources.html⟩.

[36] IAM Handwriting Database. 2002. Source: ⟨https://fki.tic.heia-fr.ch/databases/iam-handwriting-database⟩.

[37] Github repository with various IAM splits. 2021. Source: ⟨https://github.com/shonenkov/IAM-Splitting⟩.

[38] Nurseitov D, Bostanbekov K, Kurmankhojayev D, Alimova A, Abdallah A. HKR for Handwritten Kazakh and Russian database. arXiv preprint, 2020. Source: ⟨https://arxiv.org/abs/2007.03579⟩.

[39] Github with HKR dataset splitting. 2020. Source: ⟨https://github.com/bosskairat/Dataset⟩.

[40] Reza AM. Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement. The Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology 2004; 38(1): 35-44.

[41] Fischer A, Frinken V, Fornés A, Bunke H. Transcription alignment of Latin manuscripts using Hidden Markov Models. Proc 2011 Workshop on Historical Document Imaging and Processing (HIP'11) 2011: 29-36.

[42] de Sousa Neto AF, Bezerra BLD, Toselli AH, Lima EB. HTR-Flor: A deep learning system for offline handwritten text recognition. 33rd SIBGRAPI Conference on Graphics, Patterns and Images 2020: 54-61.

[43] HTR-Flor implementation. 2019. Source: ⟨https://github.com/arthurflor23/handwritten-text-recognition⟩.

[44] Strauss T, Leifert G, Labahn R, Hodel T, Mühlberger G. Icfhr2018 competition on automated text recognition on a read dataset. 16th Int Conf on Frontiers in Handwriting Recognition (ICFHR) 2018: 477-482.

[45] Coquenet D, Chatelain C, Paquet T. End-to-end handwritten paragraph text recognition using a vertical attention network. arXiv preprint, 2020. Source: ⟨https://arxiv.org/abs/2012.03868⟩.

[46] Moysset B, Messina R. Are 2D-LSTM really dead for offline text recognition. Int J Doc Anal Recognit 2019; 22(3): 193-208.

[47] Wang T, Zhu Y, Jin L, Luo C, Chen X, Wu Y, Wang Q, Cai M. Decoupled attention network for text recognition. Proc AAAI Conf on Artificial Intelligence 2020; 34(07): 12216-12224.

Abdallah A, Hamada M, Nurseitov D. Attention-based fully gated CNN-BGRU for Russian handwritten text. J Imaging 2020; 6(12): 141.

Authors' information

Alex Vladimirovich Shonenkov (b. 1994) graduated from the Moscow Institute of Physics and Technology in 2018. Now he works as the ML researcher at Sber AI. He is AI enthusiast at Kaggle. Research interests: deep learning modality fusion, image processing, natural language processing, AI competitions. E-mail: *AVShonenkov@sberbank.ru*.

Denis Konstantinovich Karachev (b. 1993) graduated from the Ural State University of Railway Transport in 2018, majoring in Mechatronics and Robotics. Currently, he works as the leading analytic at the Industry center for the development and implementation of information systems (OCRV). Research interests are computer vision, generative adversarial networks, reinforcement learning. E-mail: *denis.karachev@ocrv.ru*.

Maxim Yurievich Novopoltsev (b. 1983) graduated from the Ufa State Aircraft Technological University in 2006. Now he works as the Data Scientist Team Lead at Sber AI. Research interests: deep learning, image processing, text recognition, and pattern recognition. E-mail: *MYNovopoltsev@sberbank.ru* .

Mark Stanislavovich Potanin (b. 1994) is currently a Ph.D. student at the Moscow Institute of Physics and Technologies with scientific research of neural architecture. Also, he works as a data scientist at Sber AI. Main fields of interest: computer vision, object detection, text recognition. E-mail: *mark.potanin@phystech.edu*.

Denis Valerievich Dimitrov (b. 1994) is currently a Ph.D. student at the Probability Theory department, Mechanics and Mathematics faculty, Lomonosov Moscow State University, under the supervision of Professor Dr. Alexander Bulinski. His current research includes both strictly mathematical issues concerning statistical estimation of the f-divergences and applications such as multivariate inhomogeneities detection, feature selection, handwritten text recognition and generative computer vision models (including big transformers). Senior Lecturer at the AI Academy. Lecturer at the NewProLab, is teaching course on a time series. Now he works as the Managing Director of Data Research at Sber AI. Research interests: deep learning, image processing, generative adversarial networks, inhomogeneities detection, feature selection. E-mail: *denis.dimitrov@math.msu.su*.

Andrey Victorovich Chertok (b. 1987) graduated Ph.D. from Moscow State University in 2012. Now he works as the Managing Director-Head of the Department at Sber AI. Research interests: deep learning, computer vision, natural language processing, artificial general intelligence. E-mail: *AChertok@sberbank.ru*.

Received September 15, 2021. The final version – October 20, 2021.