Handwritten Text Recognition

Dinmukhamed Sagynbay, Ruslan Zhagypar, and Kalamkas Zhagyparova
School of Engineering and Digital Sciences, Nazarbayev University,
Kabanbay Batyr Avenue 53, Nur-Sultan City, Republic of Kazakhstan.
E-mails: {dinmukhamed.sagynbay, ruslan.zhagypar, kalamkas.zhagyparova}@nu.edu.kz

***Abstract*—Handwritten text recognition (HTR) is one of the prominent research areas in computer vision, which allows convertion of handwritten text into its digital format. In this project, HTR system based on individual character recognition, also known as optical character recognition (OCR), is presented. After preprocessing the image with a word, the system classifies the segmented handwritten characters to one of the 35 classes for letters in the English alphabet and digits. The scope of the project considers the following machine learning and deep learning algorithms: convolutional neural networks (CNN), ResNet9 pre-trained model, multilayer perceptron (MLP), linear regression, and logistic regression. The proposed CNN method resulted in a relatively high testing accuracy (88.8%). The paper also contains detailed comparison results between the aforementioned methods.**

***Index Terms*—handwriting text recognition, optical character recognition, classification, languages, feature extraction, deep learning.**

I. Introduction

The handwritten text recognition (HTR) is a system which converts handwritten input text to a machine-encoded format, thus allows to further edit, search and store the document. Taking into account the amount of handwritten documents and texts, HTR systems have an invaluable practical application. Different oganizations are satisfying the needs of digital preservation of historic data, law documents, educational persistence [1] etc through HTR systems. In this project, HTR system based on Optical Character Recognition (OCR) is presented. As the name suggests, OCR is dedicated to recognize single characters, which can be further applied to a word-level recognition. OCR utilizes image processing technologies to convert characters on scanned documents into digital forms. It typically performs well in machine-printed fonts. However, it still poses difficult challenges for machines to recognize handwritten characters, because of the huge variation in individual writing styles. Generally, OCR system includes three main steps: 1) image acquisition and preprocessing, 2) feature extraction and 3) classification [2]. In the first stage, image of text needs to be preprocessed by reducing noise, binarization of input image, dilating the image in order to enhance the image pixels and make them to be detected accurately, and segmentation of characters from input words.

Once a word in an image is segmented to characters, the further recognition step can be performed using machine learning and artificial neural network (ANN) algorithms. Authors of [3] compare the performance of multilayer perceptron (MLP) or support vector machines (SVM) classifiers on the basis of the Chars74K dataset. It has been highlighted that SVM outperforms MLP reaching classification accuracy of 62.93% on test data set. However, it should be noted that SVM needs the features to be extracted separately. The extracted features include fourier descriptors such as fourier angle and fourier magnitude [3]. On the other hand, there is no separate feature extraction step present with neural networks. The application of convolution neural networks (CNN) is discussed in [2], [4]. Specifically, Tang et al. proposed adding an adaptation layer in CNN via transfer learning, which resulted in an improvement in performance for historical Chinese character recognition tasks [5]. Three network architectures are described in [2], which are: 1) 10-layer CNN (77.0% accuracy); 2) 5-layer CNN (80.0% accuracy); and 3) inception V3 model (90.6% accuracy). As to our best knowledge, this is the top performing model for handwriting text recognition, thus, CNN was selected as the state-of-the-art method.

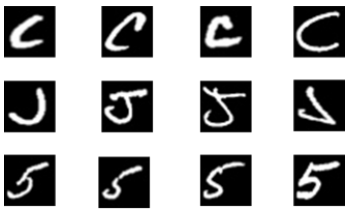

Fig. 1. Samples from dataset for letterc C and J, and digit 5.

The remaining of the paper is organized as follows. The preprocessing step, applied dataset and the proposed recognition model is described in Section II. Section III describes and discusses our results and experimental evaluation. Section V concludes the paper.

II. Methods

A. Datasets

1) Dataset for Character Recognition: The dataset for character recognition [6] contains all the English alphabets (small and caps) and digits. The images are 32 by 32 pixel black and white images There are total 35 categories 26 for Alphabets (small and capital letters are combined to create a single class of each character) and 9 for Digits (i.e. 1 to 9). There are total 856,560 images in this dataset, however, we used 4000 images from each class during training, and 1000 images from each class during validation. Samples from the dataset are represented in Fig. 1.

Fig. 2. Samples from dataset for handwritten text.

2) Dataset for Word Recognition: This dataset for handwritten word recognition consists of more than 400,000 handwritten names in English collected through charity projects [7]. There are 206,799 first names and 207,024 surnames in total. It should be noted that characters of the words are not connected, and are written separately from each other. This dataset was used to test the overall performance of the system, thus, we used it only for illustration purposes. Samples from the dataset are represented in Fig. 2.

B. Preprocessing

Preprocessing step incorporates the dilation operation, segmentation and sorting. Dilation is a morphological operation, which increases the object (character) area by applying a structuring element to an input image. Dilation is used to accentuate features of the image, in our case, it makes the detection of contours (the boundaries of a shape with same intensity) more accurate. The next step is a segmentation of discrete characters from the word in an image, which was performed via OpenCV functions for contour detection. Since the characters in a word are not connected, it is possible to detect the contour of each character, and thus separate them by comparing their contour areas. Finally, the separated contour is sent to OCR model to get its digital counterpart.

Post processing step includes the sorting and merging recognized letters into a word. The obtained contours are sorted to get the correct order of individual recognized characters for correct output extraction. In this case, for extracting a single word, a left to right sorting of individual characters was implemented. The individual word is then obtained by fetching the list of letters in a correct order. Fig. 3 represents the flowchart of the whole HTR system.

C. Architecture

1) Backbone of the CNN architecture: When dealing with image data convolutional neural networks (CNN) have shown to be extremely efficient. Unlike regular fully connected neural networks layers in CNN span in three dimensions: (C, H, W). C - represents the number of channels in each layer, whereas height(H) and width (W) represent the dimensions of the ”sheets” that make up the layer. Each channel employs a filter with learnable parameters, and during forward pass a the input is convolved with the filter resulting in a 2D activation map. The output is represents those activation maps stacked together. Each convolutional layer has 4 main hyperparameters: a) Number of channels; b) Kernel size (i.e. kxk filter); c) Filter stride which is the step at which the filter traverses over the input; d) Padding - decides how to handle the edges of the input, if the parameter is set to zero, unless the kernel size is 1, the output will be cropped.

The architecture of the CNN that was used for this particular problem is depicted in Fig. 4. It consists of three convolutional layers with kernel size 3 and dimensions as highlighted in the figure. After each of these layers RelU activation function is utilized, which is followed by maxpooling (kernel size - 2, stride - 2). The output of the convolutional layers is flattened, and then fed into 2 fully connected NN layers with 128 and 35 neurons, respectively.

2) Combatting Overfitting: When dealing with complex models the problem of overfitting becomes increasingly relevant. To prevent this in our project, we employed three strategies. Firstly, use as many samples training and validation samples as possible. As outlined previously, the dataset is immense, therefore due to technical limitations we could not make use of all of them. Therefore, it was decided to use an optimal number of samples, so that hardware load is relieved, but at the same time, the performance of the model does not suffer. Secondly, use dropout technique, where at selected layers certain channels (or neurons in 1D case) get randomly zeroed out, although with a pre-set probability. This method tackles overfitting as it ensures that the model will be rigid and will not depend on the output of every element in the model. This, in turn, makes the generalization error more acceptable. Finally, batch normalization is added to further optimize the performance. The method ensures that the data stays normalized not only at the input but also at certain selected layers.

D. Model Selection

The model selection problem is regarded in terms of the following two sub-tasks: 1) selection of the best learning method; and 2) selection of the best set of hyperparameters in the proposed model. We do not need to perform cross-validation technique as there is enough data for training, validation, and test purposes. The project considers several models for the purpose of OCR: 1) proposed CNN model; 2) pre-trained ResNet9 model; 3) multilayer perceptron (MLP); and 4) linear and logistic regressions. The architecture of MLP is represented in Fig. 5. It consists of three hidden dense layers with 1000 neurons, which have rectified linear unit as an activation function. The last layer has 35 neurons corresponding to the number of classes and the softmax function to have probabilities as output. The model was trained with the traning data of size 4000 , batch size of 128, and total of 20 epochs. Scikit-learn library was leveraged for application of linear regression, logistic regression, and SVM algorithms. In logistic regression, 'saga' solver was used as it allows to work with large datasets. It should be noted that no concrete features were separately extracted for the training phase of all

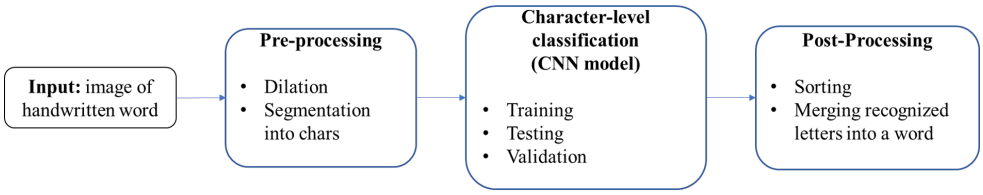

Fig. 3. The flowchart of the HTR model

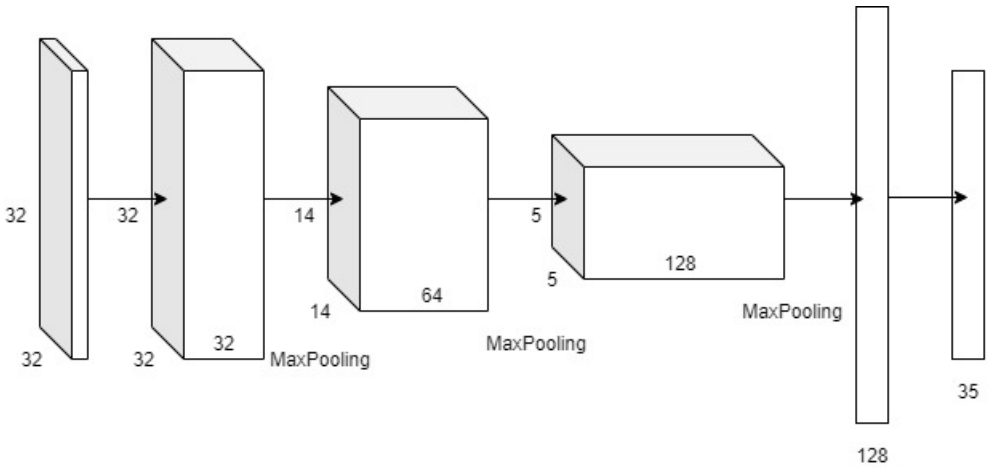

Fig. 4. Employed CNN Architecture

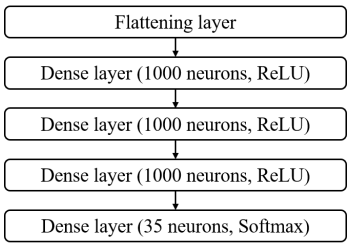

Fig. 5. The architecture of the MLP model

algorithms. Moreover, the labels had to be encoded for the purpose of training.

III. Results

The results regarding the CNN architecture can be seen in the figures below (Fig. 7, 8, 9, 10, 11). It can be clearly seen that even without the dropout and batch normalization methods being employed the validation accuracy of 87.82% is achieved. As we employ the aforementioned techniques the model performance gets better, however, only negligibly. The best validation accuracy is observed while using dropouts. To assess the performance in the context of existing solutions ResNet9 architecture was implemented for our problem. This model showed highest accuracy, however, not by a great margin, with 90 %. The results of testing with test data of size 1000 for the aforementioned methods are given Table I. The proposed method shows validation accuracy of 88.8%, which is a similar result to that of the pre-trained ResNet9

TABLE I
Testing accuracy results of the methods

Methods	Test Accuracy
Proposed CNN	88.8%
ResNet9	90.0%
MLP	84.9%
Logistic Regression	16.9%
Linear Regression	33.8%

(90.0%). Third highest accuracy belongs to MLP that reaches an accuracy of 84.9%. The plot of training and validation accuracy against the epoch number for MLP is shown in Fig. 6. It is visible that the highest validation accuracy is attained at 18^{th} epoch. One can notice that the linear regression algorithms, which is built in scikit-learn library, fail to show high results for this task (33.8%). The validation plot for the logistic regression is illustrated in Fig. 12, which shows the validation accuracy of 57.0%. The factors that deteriorate its performance could be a large size of the feature vector and a high number of classes.

IV. Discussion

One of the **limitations** of the project is the dependence of the recognition part on the preprocessing stage. In this project, since we rely on the segmentation based on detected contours, the text can be recognized only if it's letters are not connected. And if the OpenCV library is not able to find the character contour, then this method will fail. Nevertheless, it is possible to apply other character segmentation methods for connected

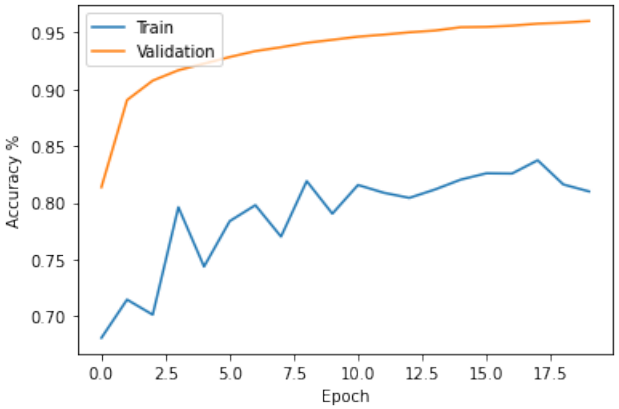

Fig. 6. The accuracy plot of the MLP model (highest validation accuracy of 83.7% at 18th epoch)

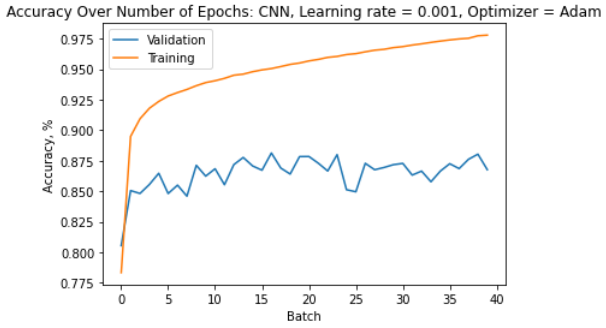

Fig. 7. The accuracy plot of the CNN model (highest validation accuracy of 87.82%)

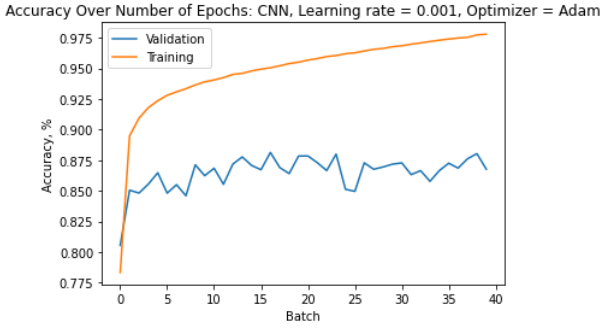

Fig. 8. The accuracy plot of the CNN model (highest validation accuracy of 88.13%)

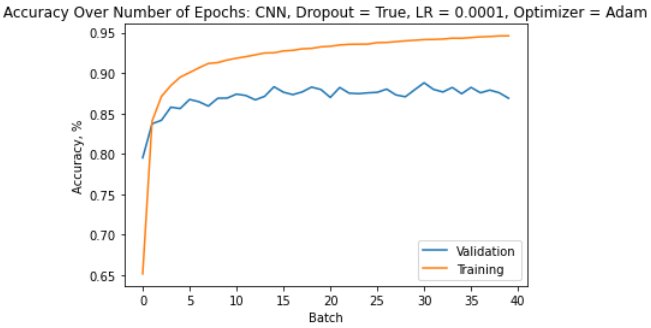

Fig. 9. The accuracy plot of the CNN model (highest validation accuracy of 88.8%)

letters within a word to increase the range of recognizeable input text.

There could be a lot of variation in a single handwritten letter in terms of writing style, therefore a lot more examples are needed for training this model.

V. Team Members and Contribution

All three members contributed equally to the project's implementation. At initial stages, each member conducted a literature review on OCR and HTR, and tried to reimplement the publicly available codes. After discussing the state-of-the-art methods for character recognition, and finalising the dataset to be used, each member chose additional models to try on. Dinmukhamed was responsible for CNN and ResNet9 architectures, Ruslan implemented SVM and MLP and also tried Linear and Logistic Regression models. Kalamkas tried GMM and helped to tune parameters, and was responsible for pre-processing and post-processing stages. The report was also divided according to the conducted work.

VI. Conclusion

As a final course project, handwritten text recognition was implemented based on character recognition model. Firstly, the image with the handwritten word is segmented into individual characters, which are then taken as the input to the machine learning and deep learning algorithms. The scope of the project considered CNN, ResNet9, MLP, Linear and Logistic Regression models to classify the handwritten character into one of the 35 categories of English letters and digits. The proposed CNN method resulted in a relatively high testing accuracy (**88.79%**), which coincides with the result of top performing method from the literature review. Although the proposed method is limited to classify words with unconnected letters, this merely depends on the preprocessing step, which can be further improved by considering more advanced segmentation methods.

References

[1] J. Memon, M. Sami, R. A. Khan and M. Uddin, "Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)", in *IEEE Access*, vol. 8, pp. 142642-142668, 2020.

[2] T. C. Wei, U. U. Sheikh and A. A. A. Rahman, "Improved optical character recognition with deep neural network," in *Proc. IEEE 14th International Colloquium on Signal Processing Its Applications (CSPA)*, pp. 245-249, 2018.

[3] A. Gupta, M. Srivastava and C. Mahanta, "Offline handwritten character recognition using neural network," in *IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, pp. 102-107, 2011.

[4] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks," in *Proc. of the 2014 Computer Vision and Pattern Recognition (CVPR)*, pp. 1717–1724, 2014.

[5] Y. Tang, L. Peng, Q. Xu, Y. Wang, and A. Furuhata, "CNN Based Transfer Learning for Historical Chinese Character Recognition," in *Proceedings of the 12th IAPR Workshop on Document Analysis Systems (DAS)*, pp. 25–29, 2016.

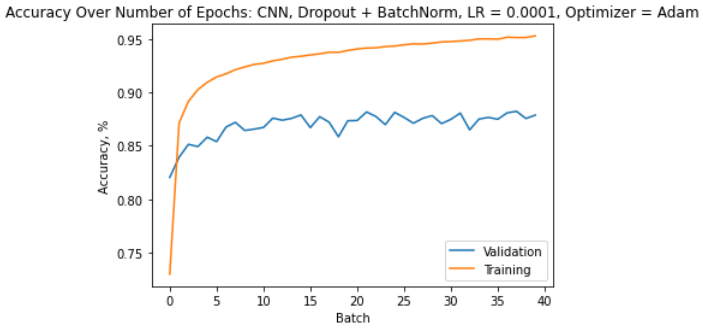

Fig. 10. The accuracy plot of the CNN model (highest validation accuracy of 88.2%)

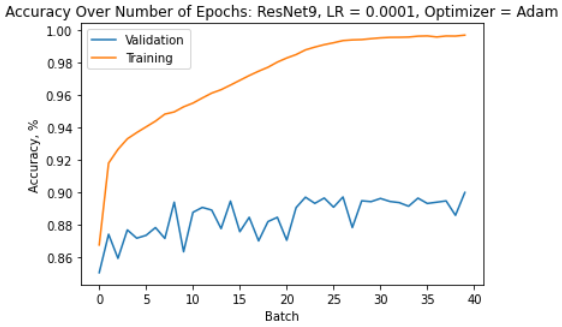

Fig. 11. The accuracy plot of the ResNet9 model (highest validation accuracy of 90.0%)

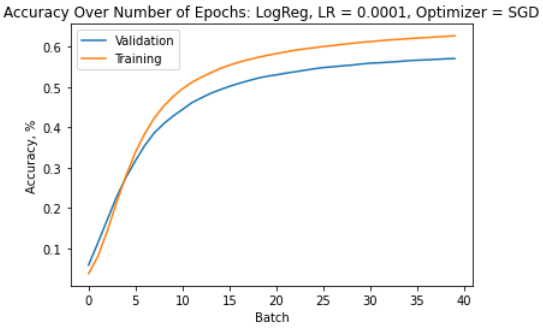

Fig. 12. The accuracy plot of the logistic regression model (highest validation accuracy of 57.0%)

[6] V. Khamgaonkar, "Handwritten characters" https://www.kaggle.com/vaibhao/handwritten-characters
[7] V. Saini, "Handwriting recognition" https://www.kaggle.com/landlord/handwriting-recognition