합성곱 신경망을 사용한 임베디드 시스템에서의 실시간 손글씨 인식

*세피데사닷 **이상훈 ***조남익
서울대학교
*sepid@snu.ac.kr

Real-Time Handwritten Letters Recognition On An Embedded Computer Using ConvNets

*Hosseini, Sepidehsadat **Lee, Sang-Hoon ***Cho, Nam-Ik
Seoul National University

Abstract

Handwritten letter recognition is important for numerous real-world applications and many topics like human-machine interaction, education, entertainment, and more. This paper describes the implementation of a real-time handwritten letters recognition system on a common embedded computer. Recognition is performed using a customized convolutional neural network, which was designed to work with low computational resources such as the Raspberry Pi platform. The experimental results show that the proposed real-time system achieves an outstanding performance in the accuracy rate and the response time for recognition of twenty-six handwritten letters.

1. Introduction

Real-world tasks such as control of touchless screens, interaction with robots, digitization of texts, and more, directly or indirectly depends on efficient handwritten letter recognize systems. For this reason, recognition of handwritten characters has been heavily studied during the last decades. But few works developed handwritten letters recognition on embedded computers, usually works related to this regard focus on handwritten digits [1, 9] and not on handwritten letters since these are composed by more characters than digits.

Early works on handwritten character recognition are based on hand-crafted extracted features [10] and then using the SVM classifier. For instance, [6] uses Fourier descriptors and achieve an accuracy of 86.6 % on the Chars74K dataset. However, in the last years, Deep Learning techniques have been shown excellent results on characters recognition. As evidence of this, [12] accomplishes an accuracy of 96.87% on a CASIA-OLHWDB1.1 dataset for Chinese character recognition using convolutional neural networks (ConvNets) based on domain-specific knowledge. Even, more recent works [6, 7] propose the application of Gabor filters before training in order to achieve better results.

In this work, we propose a real-time handwritten recognition system to work with low computational resources and consuming low power. In order to achieve these requirements, we implemented a customized ConvNet with few layers and few parameters. This architecture is of considerable importance since power consumption, time response and recognition accuracy depends on it. In the following sections, we will explain in detail this architecture as well as the whole system.

2. Methodology

A. Overview

The proposed system for handwritten letters recognition works in real-time with color RGB images captured from a camera. It was designed to be executed on embedded computers with low computational resources without GPU support and using a low power consumption. Computational demands in this system depend highly on the ConvNet, therefore, this is designed with a customized architecture in order to satisfy the goals explained before.

The embedded platform selected for this recognition system was the Raspberry Pi 3 since its versatility and its low cost make of this a perfect option. The system works on this embedded computer as follow. First, the camera captures RGB images of 400x300 pixels at 120 frames per second. Then, the image is transformed to a binary image thresholding the strokes of the handwritten letter. After, the region that contains the handwritten letter is extracted and then adjusted in order to be taken as input of the ConvNet. Finally, the result of the ConvNet is shown in the screen. Fig. 1 depicts the system.

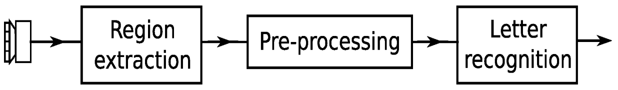


Fig. 1. Block Diagram of proposed recognition system.

B. Region Extraction And Pre-Processing

Separate the handwritten shapes from the background is the first and an important step for the proposed system. In this sense, we apply several image processing techniques that will be described as follow. First, we extracted the handwritten lines based on color thresholding and therefore the whole image is converted to the binary image. At this point, the square region that contains the handwritten letter is extracted. Then, morphological opening operation (erode and then dilate) is applied to the handwritten lines in order to remove small objects from the foreground. Next, morphological closing operation (dilate and then erode) is applied to remove small holes from the foreground. After these operations, the moments of the transformed images are calculated. Finally, based on these moments, the small areas are not considered since these are mainly produced by noise.

C. Handwritten Letters Recognition

The design and implementation of the recognition system have two steps: training phase and inference phase. Both steps will be explained in detail in the next paragraphs.
In the training phase, the model learns from a dataset based on the back-propagation algorithm [8], which is a method used to calculate a gradient that is needed in the calculation of the weights to be used in the network. In this regard, the deep learning framework used to deployed and train our model was Caffe [4]. This is a framework made with expression, speed, and modularity, these features make Caffe a good option for our project. Usually, training step is performed on computers with powerful GPUs. In this project, training step was performed on a computer with the following computational resources: Intel Core 7 Octa-Core CPU @3.8 GHz, 12 GB RAM, and GeForce GTX 1050 Ti GPU. After training step, we obtained the .caffemodel Caffe file with the trained weights of the designed model.
Afer training phase, inference step is performed. This means that the trained parameters are used to perform classification in real-time. Therefore, we use the .caffemodel file into our real-time recognition system in order to identify different handwritten letters. Since our project focuses on obtaining a fast time response using low computational resources, this system was fully implemented in C++ and using OpenCV libraries. Furthermore, the hardware used to perform the inference phase was the Raspberry Pi 3 platform, which has the following computational resources: ARM Cortex A53 Quad-Core CPU @1.2 GHz, 1 GB RAM.
Separate the handwritten shapes from the background is the first and an important step for the proposed system. In this sense, we apply several image processing techniques that will be described as follow. First, we extracted the handwritten lines based on color thresholding and therefore the whole image is converted to the binary image. At this point, the square region that contains the handwritten letter is extracted. Then, morphological opening operation (erode and then dilate) is applied to the handwritten lines in order to remove small objects from the foreground. Next, morphological closing operation (dilate and then erode) is applied to remove small holes from the foreground. After these operations, the moments of the transformed images are calculated. Finally, based on these moments, the small areas are not considered since these are mainly produced by noise.

D. Handwritten Letters Dataset

The EMNIST Letters Dataset [2] was used for training and testing the deep neural model. This dataset is part of the EMNIST Dataset [2] and is composed by handwritten character digits derived from the NIST Special Database [3] and converted to a 28x28 pixel image format. The EMNIST Letters Dataset has a total of 145,600 uppercase and lowercase letters divided into 26 balanced classes. In order to train an test our proposed model, we used the 86% of the whole EMNIST Letters Dataset for training and 14% for testing. It means that 124,800 images were used for training and 20,800 images for testing. It means that 124,800 images were used for training and 20,800 images for testing. It means that 124,800 images were used for training and 20,800 images for testing.

E. Convolutional Nueral Netwrok

As explained in the introduction and overview of this paper, we focused on a small ConvNet in order to work on embedded computers with low computational resources and using low power consumption. Since each handwritten letter is composed of strongly differentiated strokes, recognition doesn't remind large images and a complex ConvNet. Therefore, we used binary images of 28x28 pixels and a small deep neural network wich few layers.
The proposed ConvNet is formed by two convolutional layers with kernels of size 5x5 size each one, a nonlinearity (ReLU) activation function and a max-pooling layer after each convolutional layer, and one full-connected layer with 500 neurons of length followed by a ReLU activation function and with a final 26-way softmax. Furthermore, this ConvNet is composed by only 60K learnable

parameters out of 345,308 connections. This number of parameters is significantly less than the AlexNet network (60M learnable parameters and 650,000 neurons) [5] and the GoogleNet network (6.8M learnable parameters)[11]. The architecture used for handwritten letter recognition is presented in Fig. 2.

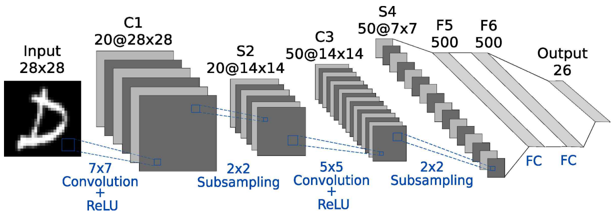


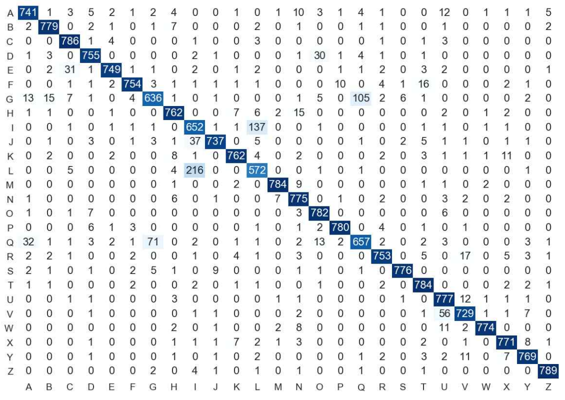
Fig. 2. Architecture of proposed Convnet.

F. Experiment and Result

After the careful design of the ConvNet and training on the EMNIST Letter Dataset, we evaluated its performance using several continuous iterations and analyzing the confusion matrix. These metrics will be explained as follow.

First of all, since each training process of the ConvNet initializes with random weights, the accuracy of each training iteration has a different value. Therefore, in order to have a precise performance metric of the designed ConvNet, 10 iterations has been evaluated After these 10 iterations, the architecture for handwritten recognition shows an outstanding maximum accuracy of 93.2%, an average accuracy rate of 92.9% and a standard deviation of 0.25.

A useful metrics of the network performance is the confusion matrix. This gives a visualization of the misclassified classes and helps to add more training images in order to improve the model. In this regard, the confusion matrix of our ConvNet, see the Fig. 3, discloses which letters are misclassified by the ConvNet. For instance, letter 'L' is sometimes confused with 'I', 'Q' with 'G', and 'D' with 'O'. In fact, these pairs of letters mentioned above are difficult to discern, even for a human.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	741	1	3	5	2	1	2	4	0	0	1	0	1	10	3	1	4	1	0	0	12	0	1	1	1	5
B	2	779	0	2	1	0	1	7	0	0	0	2	0	1	0	0	1	1	0	0	1	0	0	0	0	2
C	0	0	786	1	4	0	0	0	1	0	0	3	0	0	0	0	0	1	0	1	3	0	0	0	0	0
D	1	3	0	755	0	0	0	0	2	1	0	0	0	1	30	1	4	1	0	1	0	0	0	0	0	0
E	0	2	31	1	749	1	1	0	2	0	1	2	0	0	0	1	1	2	0	3	2	0	0	0	0	1
F	0	0	1	1	2	754	3	1	1	1	1	1	0	0	0	10	0	4	1	16	0	0	0	2	1	0
G	13	15	7	1	0	4	636	1	0	1	0	0	0	1	5	0	105	2	6	1	0	0	0	0	2	0
H	1	1	0	0	1	0	0	762	0	0	7	6	2	15	0	0	0	0	0	0	2	0	1	2	0	0
I	0	0	1	0	1	1	1	0	652	1	1	137	0	0	1	0	0	0	0	1	1	0	1	1	0	0
J	0	1	0	3	0	1	3	1	37	737	0	5	0	0	0	0	1	0	2	5	1	1	0	1	1	0
K	0	2	0	0	2	0	0	8	1	0	762	4	0	2	0	0	0	2	0	3	1	1	1	11	0	0
L	0	0	5	0	0	0	0	4	216	0	0	572	0	0	1	0	1	0	0	0	0	0	0	0	1	0
M	0	0	0	0	0	0	0	1	0	0	2	0	784	9	0	0	0	0	0	1	1	0	2	0	0	0
N	1	0	0	0	0	0	0	6	0	1	0	0	7	775	0	1	0	2	0	0	3	2	0	2	0	0
O	1	0	1	7	0	0	0	0	0	0	0	0	0	3	782	0	0	0	0	0	6	0	0	0	0	0
P	0	0	0	6	1	3	0	0	0	0	0	1	0	1	2	780	0	4	0	1	0	1	0	0	0	0
Q	32	1	1	3	2	1	71	0	2	0	1	1	0	2	13	2	657	2	0	2	3	0	0	0	3	1
R	2	2	1	0	0	2	0	0	1	0	4	1	0	3	0	0	0	753	0	5	0	17	0	5	3	1
S	2	1	0	1	0	2	5	1	0	9	0	0	0	1	1	0	1	0	776	0	0	0	0	0	0	0
T	1	1	0	0	0	2	0	0	2	0	1	1	0	0	1	0	0	2	0	784	0	0	0	2	2	1
U	0	0	1	1	0	0	0	3	0	0	0	0	1	1	0	0	0	0	1	0	777	12	1	1	1	0
V	0	0	0	1	0	0	0	0	1	1	0	0	0	2	0	0	0	0	0	1	56	729	1	1	7	0
W	0	0	0	0	0	0	0	2	0	1	0	0	2	8	0	0	0	0	0	0	11	2	774	0	0	0
X	0	0	0	1	0	0	0	1	1	1	7	2	1	3	0	0	0	0	0	2	0	1	0	771	8	1
Y	0	0	0	1	0	0	0	1	0	1	0	2	0	0	0	0	1	2	0	3	2	11	0	7	769	0
Z	0	0	0	0	0	0	2	0	4	1	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	789

Fig. 3 . Confusion Matrix.

G. System Result

Once the trained ConvNet is obtained using the Caffe framework, we get the architecture and learned weights on a .caffemodel Caffe file. Next, this file with the useful information of our trained model is used into the real-time recognition system, which was implemented on the Raspberry Pi 3 using C++ language. Finally, the overall handwritten letters recognition system is evaluated in real-time under different conditions and perturbations.

The implementation of the proposed system on a common personal computer has no problems since its high computational resources. However, when our ConvNet-based system is implemented on an embedded computer like the Raspberry Pi 3 we have two major pitfalls working against us: restricted memory (only 1GB) and limited processor speed (four ARM Cortex-A53 core running at 1.2GHz).

In spite of the computer-processing limitations mentioned above, our embedded real-time handwritten recognition system shows promising results. Fig. 4 depicts its performance in real environments. As you can observe, the system correctly recognizes different uppercase and lowercase handwritten letters despite distortions, different light conditions, and different sizes and shapes of the handwritten letters. In addition to this, after we evaluated 10 consecutive inference iterations we obtain an outstanding system response time of about 22.8 milliseconds to classify a single handwritten letter.

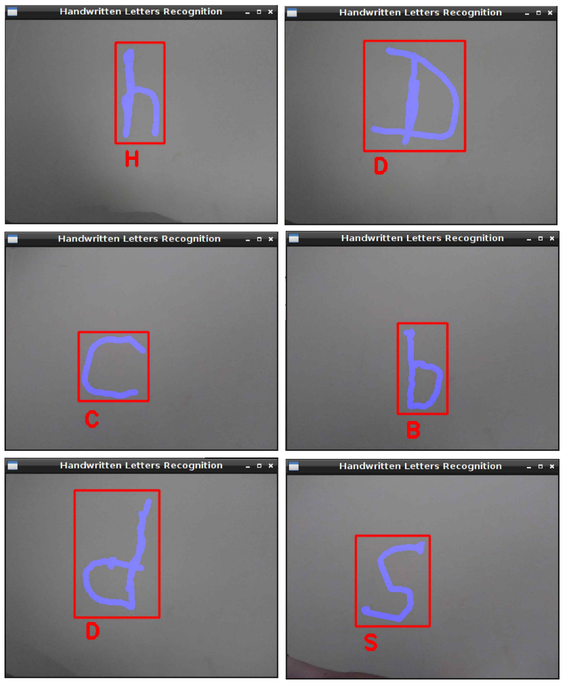

Handwritten Letters Recognition
H
Handwritten Letters Recognition
D
Handwritten Letters Recognition
C
Handwritten Letters Recognition
B
Handwritten Letters Recognition
D
Handwritten Letters Recognition
S

Fig. 4. Results of the handwritten letters recognition on a Raspberry Pi 3.

3. Reference

[1] E. Bouvett, O. Casha, I. Grech, et al., ''An FPGA embedded system architecture for handwritten symbol recognition'', 16th IEEE Mediterranean Electrotechnical Conference, Tunisia, 2012.
[2] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, ''EMNIST: Extending MNIST to handwritten letters'', 2017 International Joint Conference on Neural Networks (IJCNN), USA, 2017.
[3] P. J. Grother, K. K. Hanaoka, ''NIST Special Database 19 Handprinted Forms and Characters Database'', National Institute of Standards and Technology, USA, 2016.
[4] Y. Jia, E. Shelhamer, J. Donahue, et al., ''Caffe: Convolutional architecture for fast feature embedding'', ACM Int. Conf. on Multim., 2014.
[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ''ImageNet Classification with Deep Convolutional Neural Networks'', Advances in neural information processing systems, USA, 2012.
[6] A. Mehta, M. Srivastava, Ch. Mahanta, "Offline handwritten character recognition using neural network", IEEE International conference on computer applications and Industrial Electronics, Malaysia, 2011.
[7] D. Nunez, and B. Kwolek, ''Hand Posture Recognition Using Convolutional Neural Network'', 22nd Iberoamerican Congress on Pattern Recognition, Chile, November 2017.
[8] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, ''Learning representations by back-propagating errors'', Nature, USA, 1986.
[9] L. B. Saldanha, and Ch. Bobda, ''An embedded system for handwritten digit recognition'', Journal of Systems Architecture, USA, 2015.
[10] N. Sharma, T. Patnaik, B. Kumar, ''Recognition for Handwritten English Letters: A Review'', International Journal of Engineering and Innovative Technology (IJEIT), India, 2013.
[11] Ch. Szegedy, W. Liu , Y. Jia, et al., ''Going Deeper with Convolutions'', IEEE Conference on Computer Vision and Pattern Recognition (CVPR), USA, 2015.
[12] W. Yang, L. Jin, Z. Xie, and Z. Feng, ''Improved Deep Convolutional Neural Network For Online Handwritten Chinese Character Recognition using Domain-Specific Knowledge'', 13th International Conference on Document Analysis and Recognition (ICDAR), Tunisia, 2015.