Vietnamese character recognition based on CNN model with reduced character classes

Thi Ha Phan[1], Duc Chung Tran[2], Mohd Fadzil Hassan[3]
[1]Information Technology Department, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam
[2]Computing Fundamental Department, FPT University, Hanoi, Vietnam
[3]Department of Computer and Information Sciences, CERDAS, Universiti Teknologi Petronas, Perak, Malaysia

Article Info

Article history:

Received Oct 28, 2020
Revised Jan 18, 2021
Accepted Feb 11, 2021

Keywords:

Character
CNN
Deep learning
Recognition

ABSTRACT

This article will detail the steps to build and train the convolutional neural network (CNN) model for Vietnamese character recognition in educational books. Based on this model, a mobile application for extracting text content from images in Vietnamese textbooks was built using OpenCV and Canny edge detection algorithm. There are 178 characters classes in Vietnamese with accents. However, within the scope of Vietnamese character recognition in textbooks, some classes of characters only differ in terms of actual sizes, such as “c” and “C”, “o” and “O”. Therefore, the authors built the classification model for 138 Vietnamese character classes after filtering out similar character classes to increase the model's effectiveness.

Corresponding Author:

Thi Ha Phan
Information Technology Department
Posts and Telecommunications Institute of Technology
Hanoi, Vietnam
Email: hapt@ptit.edu.vn, hapt27@fe.edu.vn

1. INTRODUCTION

Text recognition is the process of converting the text in one or more images into documents such as those written on a computer [1-3]. Currently, the word recognition problem has been studied and widely used in the automation of office operations in many languages like English [4, 5], Chinese [6, 7], Japanese [8]. In Vietnam, there are some identification systems such as VietOCR software [9] and VNDOCR [10]. with accuracy of about 90-95% [9, 10]. In general, the recognition of Vietnamese text has been a matter of concern for many scientific researchers based on the different models and techniques such as using HANDS-VNOnDB [11], utilizing RetinaNet for text detection and Inception-v3 CNN network for feature extraction, passing through bidirectional long short-term memory (Bi-LSTM)-based RNN for text recognition [12], Bi-LSTM combined with Conditional Random Field (CRF) [13], SSD Mobilenet V2 for text detection and Attention OCR (AOCR) for text recognition [14], CNN, a well-known for image processing applications [15-18], integrated with grammatical features for emotion detection [16].

Unlike the mentioned works, in this work, the authors focused on the printed word recognition from the Vietnamese educational textbook with a simplified set of several characters: 138 character classes instead of 178 character classes since some characters are different in sizes only.

2. RESEARCH METHOD

2.1. Overview model for Vietnamese character recognition based on CNN

Figure 1 shows an overview of the proposed CNN-based reduced-character-class-model. In Figure 1, in this work, the raw data are pictures taken by phone from textbook pages in which each page is a color image and a snapshot of the entire text in it. In order to make the data usable, they need to be pre-processed to form two data sets for CNN model training (training data and testing data).

The CNN model's input is the pre-processed data (in this case, gray images of accented characters with a fixed size of 28x28 pixels), and the output is the images' corresponding labels. Here, it should be noted that the selected image size is significant for use in pattern recognition. There are 178 characters classes in Vietnamese that have accents, but within this article, based on analysis of words in Vietnamese textbooks of grade 1, there are some classes of characters that differ only in actual sizes, for example, “c” and “C”, “o” and “O”. Therefore, the authors only built identification models for 138 characters classes after filtering out similar characters. This is to increase the effectiveness of the developed model.

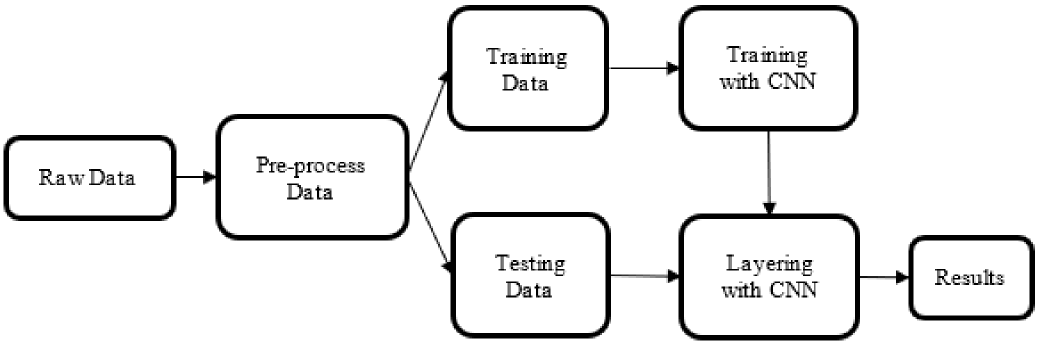


Figure 1. Overview of CNN-based reduced-character-class model for text recognition

2.2. Pre-processing data

Given an input image taken from the phone (RGB color system), the authors performed the pre-processing data steps as: converting the image to grayscale and binary images, finding the contour of tokens in the image, finding the contour of each character in each token, cutting each character and saving it. These pre-processing steps were performed in Java language in combination with the OpenCV library. The following details procedures for each step.

a. Step 1: Converting to grayscale and binary images
   OpenCV [19-21] is an open-source library built for image processing. There are functions in the library that convert color images to grayscale and binary one based on the Otsu algorithm [22]. Different from [23] which used localization thresholding technique, this work utilized the existing functions from the library.
b. Step 2: Finding the contour of each token
   To find the contour of each character, the authors built a function to perform the following:
   Perform closing operation to create a seamless connection of characters in a token; each token forms a block. However, the authors need to retain the original binary image to cut each character into pictures for the next step.
   - Next, implement the Canny [24-28] algorithm to find the contour of each word in the image.
   - Not only does the image contains text, but also other physical objects, so the boundaries found include those objects. These objects are retained, but when the identification step is taken, they are ignored.
c. Step 3: Finding the contour of each character
   In this paper, a function has been built to accomplish this task:
   - The input of the function is images of Vietnamese tokens (including the boundary connections), the output is photos of the characters, including accents.

   This step again uses the Canny algorithm on a one-token binary image to find the contour of each token's character images. Here, the boundaries of the found images will be characters (without accents). There may be additional images of individual accents (for example, the 'ế' has three (3) contours, the contour of the word 'e', '^', and an acute accent). To find the contours of a fully accented character image, this paper rearranges the rectangular blocks containing these elements, thereby combining the elements together along the same y-axis forming an image of the character to be searched.

d. Step 4: Cutting and saving the gray image of each character with the same accent, if any
   From the above steps, the area of every character with an accent image is found. The algorithm continues to cut each character from grayscale image, then resizes into 28x28 pixels to get the complete Vietnamese character image saves it as a file with format .jpg. Figure 2 presents some of the results from the processed image.

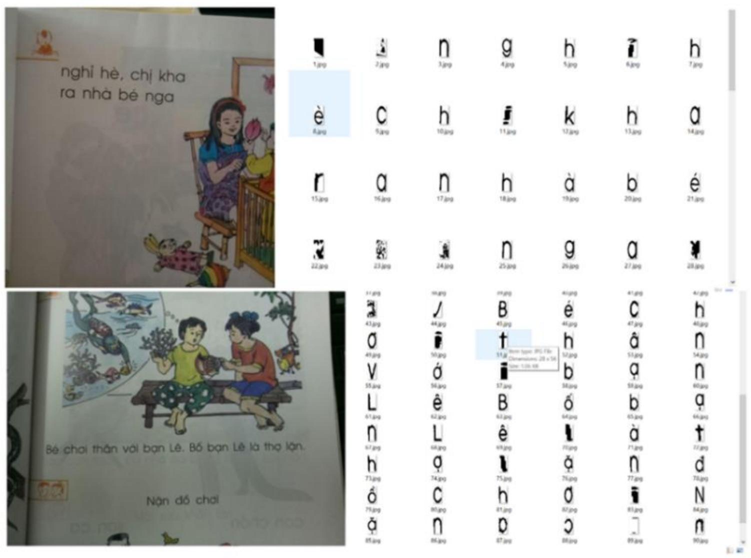


Figure 2. Image processing results

2.3. Building training and testing data

From the raw data of 2,000 photos from Vietnamese textbooks, the dataset consists of 86,544 samples and is labelled manually after pre-processing. This data set is then divided into two parts: testing and training. The testing dataset consists of 50 images per character class for a total of 6,900 samples, while the training data set is the remaining one, which is detailed in Table 1. Figure 3 presents are some images of the "Ợ" character class retrieved from the training dataset:

Table 1. Training dataset statistics

No.	Char	No. Sam ples	No.	Char	No. Sam ples	No.	Char	No. Sam ples	No.	Char	No. Sam ples	No.	Char	No. Sam ples	No.	Char	No. Sam ples
1	a	499	24	è	536	47	o	524	70	u	465	93	Ã	620	116	Ẹ	630
2	à	494	25	ẻ	477	48	ò	526	71	ù	377	94	Á	648	117	Ê	673
3	ả	515	26	ẽ	531	49	ỏ	516	72	ủ	454	95	Ạ	988	118	Ề	778
4	ã	488	27	é	543	58	õ	536	73	Ũ	400	96	Ă	702	119	Ể	612
5	Á	503	28	ẹ	545	51	ó	531	74	Ú	417	97	Ằ	523	120	Ễ	583
6	ạ	504	29	ê	520	52	ọ	553	75	ụ	427	98	Ẳ	994	121	Ế	748
7	Ă	471	30	ề	478	53	ô	554	76	Ư	402	99	Ẵ	956	122	Ệ	684
8	ằ	521	31	ể	524	54	ồ	582	77	ừ	392	100	Ắ	1047	123	G	630
9	ẳ	486	32	ễ	519	55	ổ	527	78	ử	430	101	Ặ	985	124	H	630
10	ẵ	463	33	ế	539	56	ỗ	546	79	ữ	440	102	Â	531	125	K	957
11	ắ	520	34	ệ	507	57	ố	538	80	ứ	422	103	Ầ	622	126	L	818
12	ặ	513	35	g	492	58	ộ	518	81	ự	476	104	Ẩ	639	127	M	573
13	Â	513	36	h	418	59	ơ	502	82	V	499	105	Ẫ	575	128	N	659
14	ầ	526	37	i	543	60	ờ	501	83	X	577	106	Ấ	658	129	P	600
15	ẩ	520	38	ì	472	61	ở	511	84	Y	527	107	Ậ	639	130	Q	763
16	ẫ	493	39	ỉ	515	62	ỡ	489	85	ỳ	539	108	B	594	131	R	817
17	ấ	509	40	ĩ	542	63	ớ	511	86	ỷ	584	109	D	607	132	T	1002
18	ậ	503	41	í	558	64	ợ	492	87	ỹ	577	110	Đ	619	133	Y	750
19	B	473	42	ị	521	65	p	484	88	Ý	524	111	E	699	134	Ỳ	640
20	C	500	43	k	531	66	q	512	89	ỵ	524	112	È	565	135	Ỷ	606
21	d	469	44	l	568	67	r	558	90	A	527	113	Ẻ	777	136	Ỹ	652
22	Đ	496	45	m	378	68	s	548	91	À	619	114	Ẽ	796	137	Ý	660
23	E	501	46	n	448	69	t	539	92	Ả	936	115	É	841	138	Ỵ	736

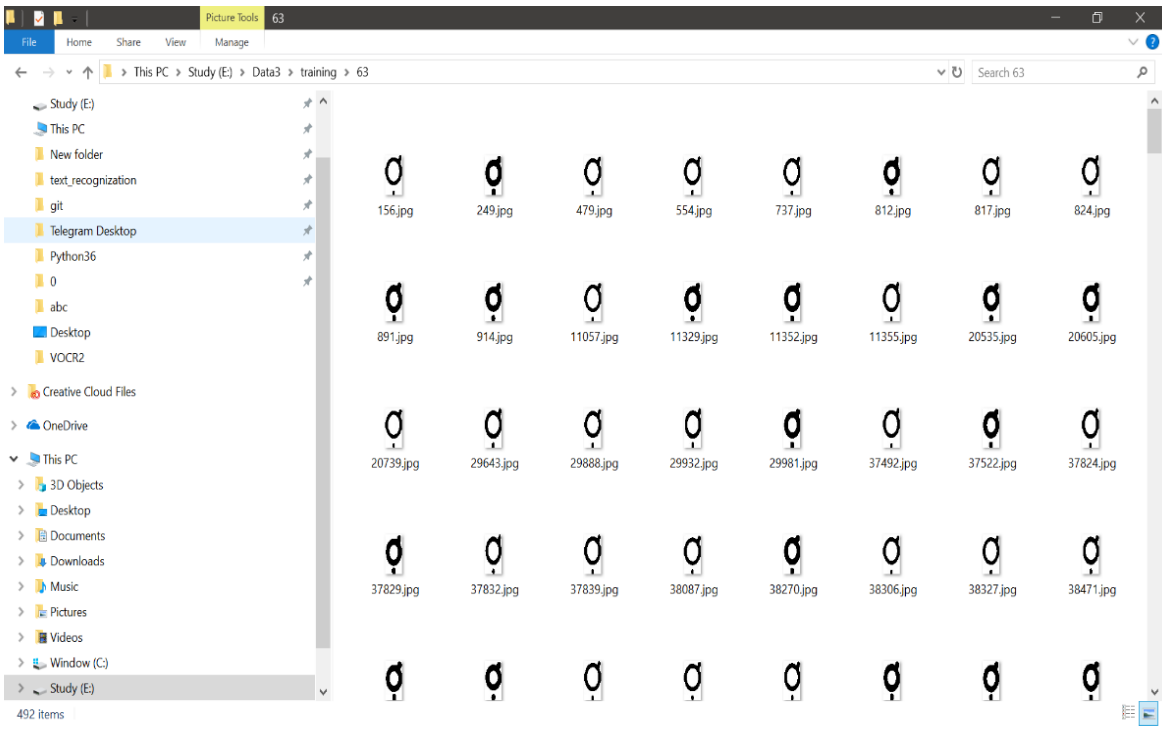


Figure 3. Training data of "Ợ" character class

2.4. Building a CNN model for Vietnamese printed character recognition and evaluating identification results

The proposed CNN model consists of four main classes: Convolution class, Relu class, Pooling class, Fully-connected class. This paper presents the proposed CNN model in Figure 4.

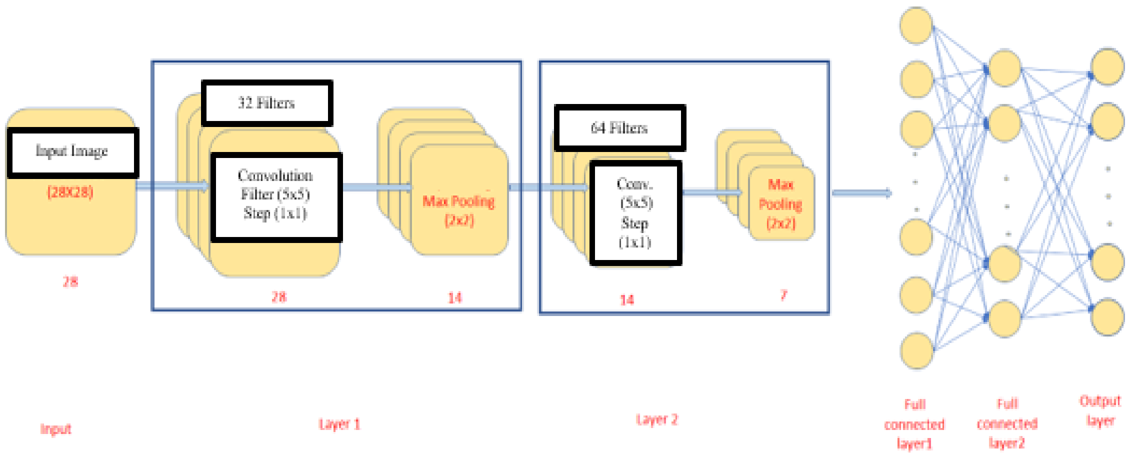


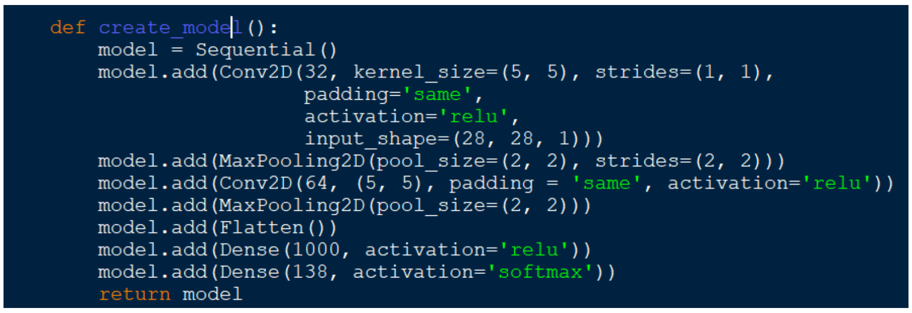
Figure 4. CNN model recognizing Vietnamese characters

Based on the above figure, a CNN model was built. The model has an Input layer which uses grey images of 28x28 pixels. Next, it has 2 Convolution layers followed by 2 Max Pooling layers correspondingly. Next are 2 Fully-connected layers. Finally, there is an Output layer having 138 outputs.

In Convolution layer 1, the input image rows and columns are padded with 0s. Because of this, when performing the convolution multiplication on the input image, it's size will remain 28x28 pixels. In this work, 32 random filters to get 32 output channels. After convolution, the next layer is Max pooling with a size of 2x2, step size of 2x2. After this step, the dimension of the image is 14x14.

In the second Convolution layer, the first layer's image is also added with rows and columns of 0 to keep the dimension number 14x14. This layer will give 64 output channels. After convolution, the image is passed through the Max pooling layer with a size of 2x2, step size of 2x2, and the dimension of the image is 7x7. After two layers of Convolution and Max pooling, we get a layer of 7x7x64 = 3,164 neurons. Next, we add another fully-connected layer of 1,000 neurons, and finally, the output layer consists of 138 outputs.

The above model is built in the Keras library [29] with Python language, as shown in Figure 5.



```
def create_model():
    model = Sequential()
    model.add(Conv2D(32, kernel_size=(5, 5), strides=(1, 1),
                     padding='same',
                     activation='relu',
                     input_shape=(28, 28, 1)))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
    model.add(Conv2D(64, (5, 5), padding = 'same', activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Flatten())
    model.add(Dense(1000, activation='relu'))
    model.add(Dense(138, activation='softmax'))
    return model
```

Figure 5. CNN model built on Keras library [29-31]

3. RESULTS AND DISCUSSION

3.1. CNN model training and evaluation results

With the training and testing data set prepared, the authors conducted model training in 2 cases:

- Case 1: Maintaining the entire testing set of 138 * 50 = 6,900 samples, the training uses 200 samples per character class, for a total of 138 * 200 = 27,600 samples. After a training time of about 2 hours, we get the results in Figure 6, Figure 7.

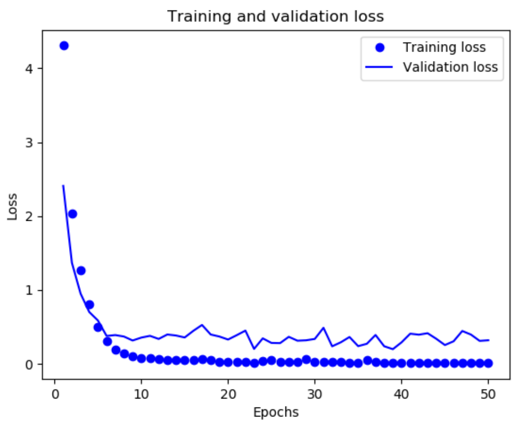


Figure 6. Loss Function chart with a training set of 200 samples per class

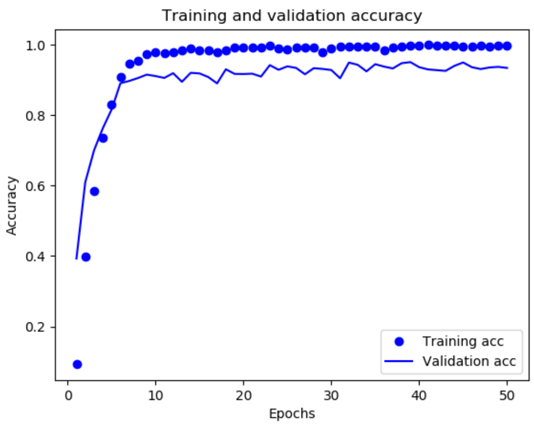


Figure 7. Accuracy chart with a training set of 200 samples per class

In case 1, the accuracy of the training set improves as time goes and the loss of the function goes closer to zero. The model results had low accuracy and instability for testing data since loss function was still large. This is the main reason why the authors conducted further training with a larger dataset. In case 2, the results were better with the full training dataset and gave greater accuracy on the testing dataset. After two training sessions, the authors found that the number of 50 training sessions for each training was sufficient because the accuracy had almost no significant change in the last period. The loss function also reached a threshold that was small enough. Summarizing the two cases, we get the results shown in Table 2.

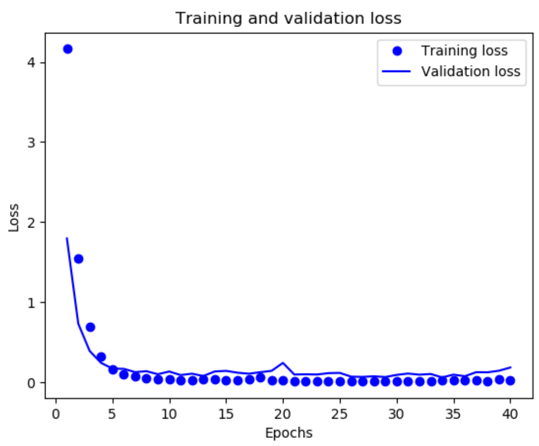


Figure 8. Loss Function diagram with complete data set

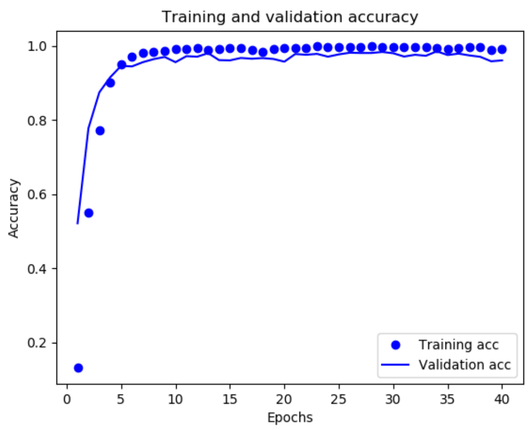


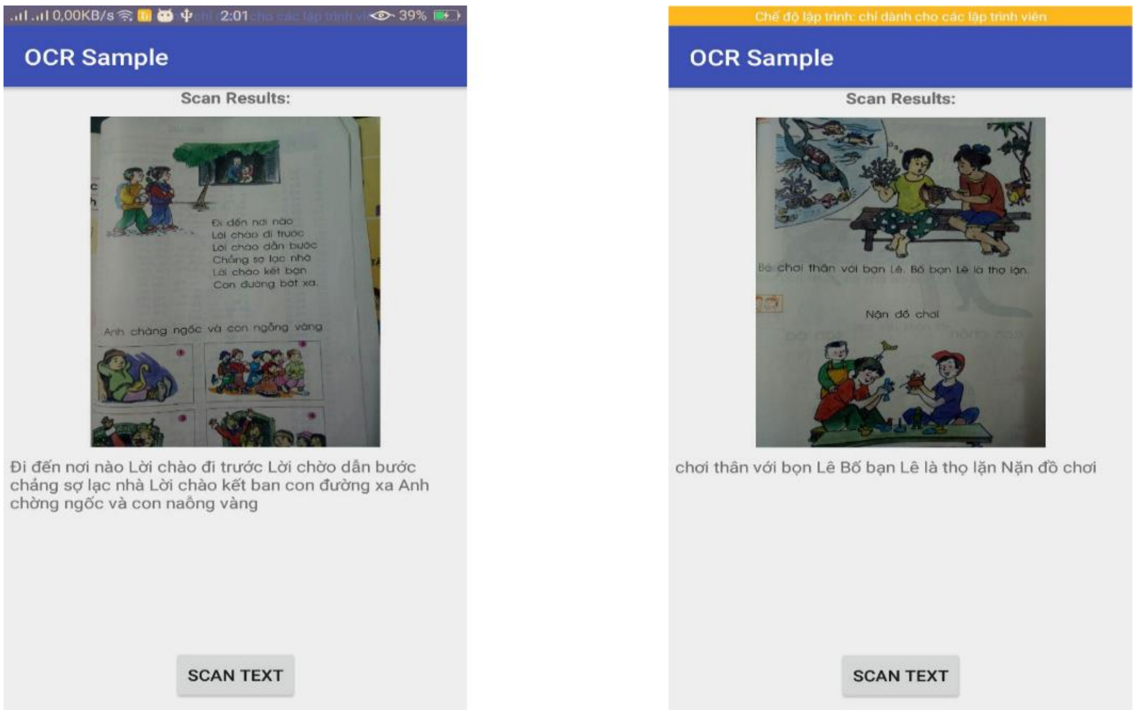
Figure 9. Accuracy chart with complete data set

Table 2. Table of CNN training model results

Training Times	No. of Training Samples	No. of Testing Samples	Accuracy
1	27,600	6,900	91.23%
2	79,644	6,900	97.14%

3.2. Token identification in Vietnamese textbooks based on the developed CNN model

A mobile demo application was built on the Android platform to test the developed CNN model's accuracy and practicality. The application's input is an image of each page in the textbook, and the output will be a sequence of tokens in text form. Here, the application took a picture, then performed pre-processing steps outlined in Section 2, and finally used the trained CNN model to produce the main result, a text string in the textbook page that has potential for auto reading application. Below are some demo images of the application shown in Figure 10.



Đi đến nơi nào Lời chào đi trước Lời chờo dẫn bước chảng sợ lạc nhà Lời chào kết ban con đường xa Anh chờng ngốc và con naổng vàng

chơi thân với bọn Lê Bố bạn Lê là thọ lặn Nặn đồ chơi

Figure 10. Images of Vietnamese token recognition in Android-based demo application developed based on the built CNN model

The photography had some issues with brightness, angle of shooting. Thus, it made the application difficult in pre-processing steps and, therefore, not well text recognition. It is also tricky to recognize uppercase or lowercase letters because there are similar faces in uppercase and lowercase letters.

4. CONCLUSION

Through research and experiment, the author group has achieved the main results: studying in detail the deep learning model to build a reduced-character-class CNN model for Vietnamese printed character recognition. The CNN model's training and testing datasets include 2,000 raw images, 79,644 training samples, and 6,900 testing samples. This work demonstrated that the developed CNN model is useful in image recognition since relatively satisfactory results were achieved. The developed model can be used in Vietnamese spelling application for learning Vietnamese. The authors also built and installed the CNN model on mobile to solve the problem of Vietnamese token identification on textbooks for learning and spelling Vietnamese.

REFERENCES

[1] Y. Liu, L. Jin and C. Fang, "Arbitrarily Shaped Scene Text Detection with a Mask Tightness Text Detector," *IEEE Transactions on Image Processing*, vol. 29, pp. 2918-2930, 2020, doi: 10.1109/TIP.2019.2954218.

[2] T. Zheng, X. Wang, and X. Xu, "A Novel Method of Detecting Chinese Rendered Text on Tilted Screen of Mobile Devices," *IEEE Access*, vol. 8, pp. 25840–25847, 2020, doi: 10.1109/ACCESS.2020.2971617.

[3] J. Guo, R. You, and L. Huang, "Mixed Vertical-and-Horizontal-Text Traffic Sign Detection and Recognition for Street-Level Scene," *IEEE Access*, vol. 8, pp. 69413–69425, 2020, doi: 10.1109/ACCESS.2020.2986500.

[4] R. Tavoli and M. Keyvanpour, "A Method for Handwritten Word Spotting Based on Particle Swarm Optimisation and Multi-Layer Perceptron," *IET Software*, vol. 12, no. 2, pp. 152–159, Apr. 2018, doi: 10.1049/iet-sen.2017.0071.

[5] C. Su, X. Huang, F. Fukumoto, J. Li, R. Wang, and Z. Chen, "English and Chinese Neural Metonymy Recognition Based on Semantic Priority Interruption Theory," *IEEE Access*, vol. 8, pp. 30060–30068, 2020, doi: 10.1109/ACCESS.2020.2972379.

[6] X. Wu, Q. Chen, J. You, and Y. Xiao, "Unconstrained Offline Handwritten Word Recognition by Position Embedding Integrated ResNets Model," *IEEE Signal Processing Letters*, vol. 26, no. 4, pp. 597–601, Apr. 2019, doi: 10.1109/LSP.2019.2895967.

[7] G. Wu, G. Tang, Z. Wang, Z. Zhang, and Z. Wang, "An Attention-Based BiLSTM-CRF Model for Chinese Clinic Named Entity Recognition," *IEEE Access*, vol. 7, pp. 113942–113949, 2019, doi: 10.1109/ACCESS.2019.2935223.

[8] A. D. Le, T. Clanuwat, and A. Kitamoto, "A Human-Inspired Recognition System for Pre-Modern Japanese Historical Documents," *IEEE Access*, vol. 7, pp. 84163–84169, 2019, doi: 10.1109/ACCESS.2019.2924449.

[9] VietOCR, "VietOCR," 2020. [Online]. Available: http://vietocr.sourceforge.net/.

[10] IoIT, "Institute of Information Technology - VnDOCR," 2020. [Online]. Available: http://www.vast.ac.vn/gioi-thieu-chung/co-cau-to-chuc/cac-vien-nghien-cuu/297-vien-cong-nghe-thong-tin.

[11] H. T. Nguyen, C. T. Nguyen and M. Nakagawa, "ICFHR 2018 – Competition on Vietnamese Online Handwritten Text Recognition using HANDS-VNOnDB (VOHTR2018)," *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 494-499, 2018, doi: 10.1109/ICFHR-2018.2018.00092.

[12] H. D. Liem, N. D. Minh, N. B. Trung, H. T. Duc, P. H. Hiep, D. V. Dung, D. H. Vu, "FVI: An End-to-end Vietnamese Identification Card Detection and Recognition in Images," 2*018 5th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 338-340, 2018, doi: 10.1109/NICS.2018.8606831.

[13] N. C. Lê, N. Nguyen, A. Trinh and H. Vu, "On the Vietnamese Name Entity Recognition: A Deep Learning Method Approach," *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, pp. 1-5, 2020, doi: 10.1109/RIVF48685.2020.9140754.

[14] H. T. Viet, Q. Hieu Dang and T. A. Vu, "A Robust End-To-End Information Extraction System for Vietnamese Identity Cards," *2019 6th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 483-488, 2019, doi: 10.1109/NICS48868.2019.9023853.

[15] A. Qayyum, C. K. Ang, S. Sridevi, M. K. A. A. Khan, L. W. Hong, M. Mazher, T. D. Chung, "Hybrid 3D-ResNet Deep Learning Model for Automatic Segmentation of Thoracic Organs at Risk in CT Images," *2020 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, pp. 1-5, 2020, doi: 10.1109/ICIEAM48468.2020.9111950.

[16] T. Huynh and A. Le, "Integrating Grammatical Features into CNN Model for Emotion Classification," *2018 5th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 243-249, 2018, doi: 10.1109/NICS.2018.8606875.

[17] Y. Peng, M. Liao, Y. Song, Z. Liu, H. He, H. Deng, Y. Wang, "FB-CNN: Feature Fusion-Based Bilinear CNN for Classification of Fruit Fly Image," *IEEE Access*, vol. 8, pp. 3987-3995, 2020, doi: 10.1109/ACCESS.2019.2961767.

[18] W. Zhao, L. Jiao, W. Ma, Jiaqi Zhao, Jin Zhao, H. Liu, X. Cao, S. Yang, "Superpixel-Based Multiple Local CNN for Panchromatic and Multispectral Image Classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 4141-4156, July 2017, doi: 10.1109/TGRS.2017.2689018.

[19] J. Huang, A. Huang and L. Wang, "Intelligent Video Surveillance of Tourist Attractions Based on Virtual Reality Technology," *IEEE Access*, vol. 8, pp. 159220-159233, 2020, doi: 10.1109/ACCESS.2020.3020637.

[20] O. Green, "Efficient Scalable Median Filtering Using Histogram-Based Operations," *IEEE Transactions on Image*

Processing, vol. 27, no. 5, pp. 2217-2228, May 2018, doi: 10.1109/TIP.2017.2781375.

[21] OpenCV, "OpenCV," 2020. [Online]. Available: https://opencv.org/.

[22] T. D. Chung and M. K. A. A. Khan, "Watershed-based Real-time Image Processing for Multi-Potholes Detection on Asphalt Road," *2019 IEEE 9th International Conference on System Engineering and Technology (ICSET)*, pp. 268-272, 2019, doi: 10.1109/ICSEngT.2019.8906371.

[23] Likun Xia, Tran Duc Chung and K. A. A. Kassim, "An automobile Detection Algorithm Development for Automated Emergency Braking System," *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1-6, 2014, doi: 10.1145/2593069.2593083.

[24] J. Liu, D. Chen, Y. Wu, R. Chen, P. Yang and H. Zhang, "Image Edge Recognition of Virtual Reality Scene Based on Multi-Operator Dynamic Weight Detection," *IEEE Access*, vol. 8, pp. 111289-111302, 2020, doi: 10.1109/ACCESS.2020.3001386.

[25] M. Kalbasi and H. Nikmehr, "Noise-Robust, Reconfigurable Canny Edge Detection and its Hardware Realization," *IEEE Access*, vol. 8, pp. 39934-39945, 2020, doi: 10.1109/ACCESS.2020.2976860.

[26] J. Lee, H. Tang and J. Park, "Energy Efficient Canny Edge Detector for Advanced Mobile Vision Applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 4, pp. 1037-1046, April 2018, doi: 10.1109/TCSVT.2016.2640038.

[27] K. A. Nemer Pelliza, M. A. Pucheta and A. G. Flesia, "Optimal Canny's Parameters Regressions for Coastal Line Detection in Satellite-Based SAR Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 1, pp. 82-86, Jan. 2020, doi: 10.1109/LGRS.2019.2916225.

[28] X. Li, X. Wang, A. Yang and M. Rong, "Partial Discharge Source Localization in GIS Based on Image Edge Detection and Support Vector Machine," *IEEE Transactions on Power Delivery*, vol. 34, no. 4, pp. 1795-1802, Aug. 2019, doi: 10.1109/TPWRD.2019.2925034.

[29] Keras, "Keras: the Python deep Learning API," 2020, [Online]. Available: https://keras.io/.

[30] G. Horng, M. Liu and C. Chen, "The Smart Image Recognition Mechanism for Crop Harvesting System in Intelligent Agriculture," *IEEE Sensors Journal*, vol. 20, no. 5, pp. 2766-2781, Mar. 2020, doi: 10.1109/JSEN.2019.2954287.

[31] J. Civit-Masot, F. Luna-Perejón, S. Vicente-Díaz, J. M. Rodríguez Corral and A. Civit, "TPU Cloud-Based Generalized U-Net for Eye Fundus Image Segmentation," *IEEE Access*, vol. 7, pp. 142379-142387, 2019, doi: 10.1109/ACCESS.2019.2944692.