Tampereen ammattikorkeakoulu

5G00FT12-3001 AI and Machine Learning

Student(s):
- Toan Tan (2004068 - toan.tan@tuni.fi)
- Minh Dang (2004376 - minh.dang@tuni.fi)

Option:
Option 1 (using neural network)

Title of the work:
Training CNN model to predict Vietnamese letters

Used algorithm(s):
- CNN
- Otsu's Binarization

TABLE OF CONTENTS

1. Description of the work 3

2. Data preparation for the training 3

3. Neural networks training 3

4. Relevant metrics for the case(s) 4

5. Conclusions of the results 4

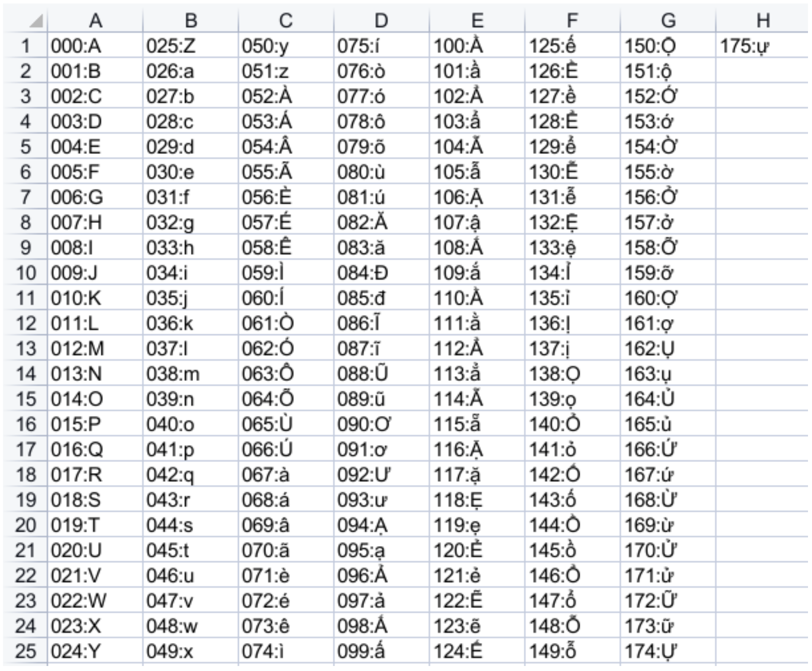
1. Description of the work

- Training CNN model to predict Vietnamese letters

2. Data preparation for the training

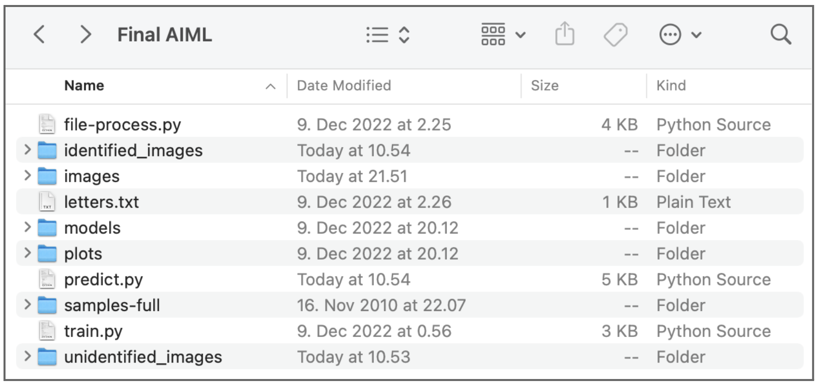
- The dataset includes 16.214 samples of 124 students of the Faculty of Information Technology, Faculty of Science and Technology, Hanoi, Vietnam (code.google.com/archive/p/sapphire-ocr)
- At the beginning, put all the images in the **samples-full** folder.
- Run the file **file-process.py** to move the images into the respective **images/[test, train]/[000, 002, 003,...173, 174, 175]** folder with 15% testing files and 85% training files.

FIGURE 2.a Table of vietnamese characters and index binding



	A	B	C	D	E	F	G	H
1	000:A	025:Z	050:y	075:í	100:Ầ	125:ế	150:Ộ	175:ự
2	001:B	026:a	051:z	076:ò	101:ầ	126:Ề	151:ộ	
3	002:C	027:b	052:À	077:ó	102:Ẩ	127:ề	152:Ớ	
4	003:D	028:c	053:Á	078:ô	103:ẩ	128:Ể	153:ớ	
5	004:E	029:d	054:Â	079:õ	104:Ẫ	129:ể	154:Ờ	
6	005:F	030:e	055:Ã	080:ù	105:ẫ	130:Ễ	155:ờ	
7	006:G	031:f	056:È	081:ú	106:Ậ	131:ễ	156:Ở	
8	007:H	032:g	057:É	082:Ă	107:ậ	132:Ệ	157:ở	
9	008:I	033:h	058:Ê	083:ă	108:Ắ	133:ệ	158:Ỡ	
10	009:J	034:i	059:Ì	084:Đ	109:ắ	134:Ỉ	159:ỡ	
11	010:K	035:j	060:Í	085:đ	110:Ằ	135:ỉ	160:Ợ	
12	011:L	036:k	061:Ò	086:Ĩ	111:ằ	136:Ị	161:ợ	
13	012:M	037:l	062:Ó	087:ĩ	112:Ẳ	137:ị	162:Ụ	
14	013:N	038:m	063:Ô	088:Ũ	113:ẳ	138:Ọ	163:ụ	
15	014:O	039:n	064:Õ	089:ũ	114:Ẵ	139:ọ	164:Ủ	
16	015:P	040:o	065:Ù	090:Ơ	115:ẵ	140:Ỏ	165:ủ	
17	016:Q	041:p	066:Ú	091:ơ	116:Ặ	141:ỏ	166:Ứ	
18	017:R	042:q	067:à	092:Ư	117:ặ	142:Ố	167:ứ	
19	018:S	043:r	068:á	093:ư	118:Ẹ	143:ố	168:Ừ	
20	019:T	044:s	069:â	094:Ạ	119:ẹ	144:Ồ	169:ừ	
21	020:U	045:t	070:ã	095:ạ	120:Ẻ	145:ồ	170:Ử	
22	021:V	046:u	071:è	096:Ả	121:ẻ	146:Ổ	171:ử	
23	022:W	047:v	072:é	097:ả	122:Ẽ	147:ổ	172:Ữ	
24	023:X	048:w	073:ê	098:Ấ	123:ẽ	148:Ỗ	173:ữ	
25	024:Y	049:x	074:ì	099:ấ	124:Ế	149:ỗ	174:Ự	

PICTURE 2.b Project files



Name	Date Modified	Size	Kind
file-process.py	9. Dec 2022 at 2.25	4 KB	Python Source
identified_images	Today at 10.54	--	Folder
images	Today at 21.51	--	Folder
letters.txt	9. Dec 2022 at 2.26	1 KB	Plain Text
models	9. Dec 2022 at 20.12	--	Folder
plots	9. Dec 2022 at 20.12	--	Folder
predict.py	Today at 10.54	5 KB	Python Source
samples-full	16. Nov 2010 at 22.07	--	Folder
train.py	9. Dec 2022 at 0.56	3 KB	Python Source
unidentified_images	Today at 10.53	--	Folder

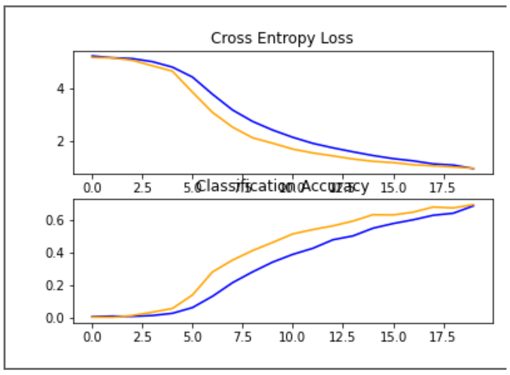
3. Neural networks training

- Refer from Offline Handwriting Recognition CNN on kaggle (kaggle.com/code/tejasreddy/offline-handwriting-recognition-cnn/notebook)

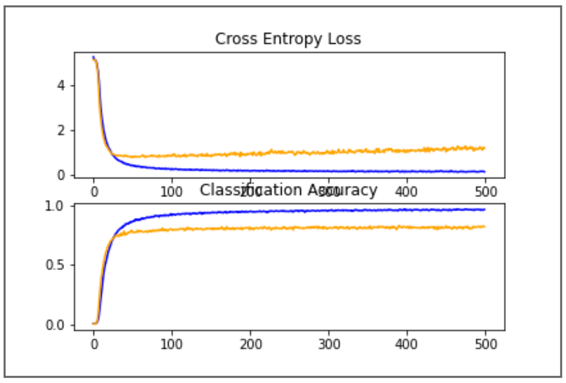
- Use **CNN** algorithm.
- Use the number of classes as the number of outputs in the output layer.
- Use **categorical_crossentropy** as the loss function.
- Read the images as files use **categorical** as class mode.
- With prediction the result, select the class with the **highest probability** as the predicted class.
- Train with **3 layers and dropouts**
- Train with differents epochs **(20-100-500-1000-2000)**
- Each epoch training will create two plots of Cross Entropy Loss and Classification Accuracy

4. Relevant metrics for the case(s)

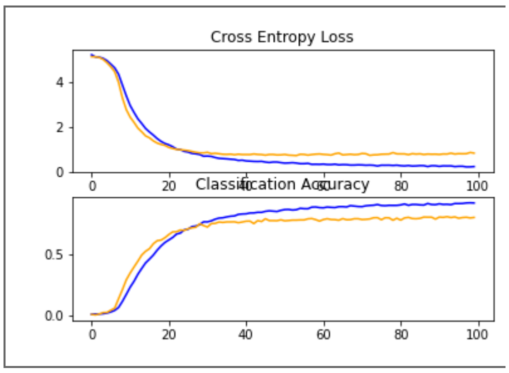
- This training will use Cross Entropy Loss and Classification Accuracy as analysis metrics



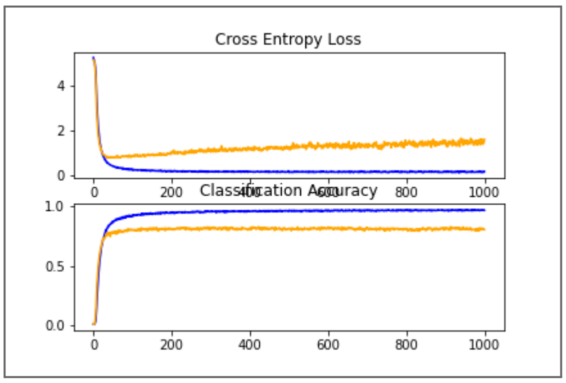
(epochs=20)



(epochs=500)



(epochs=100)



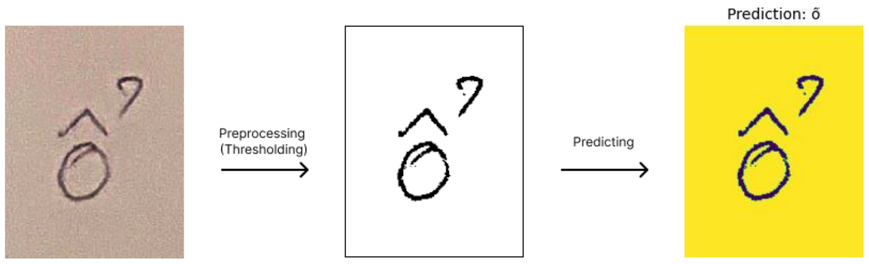
(epochs=1000)

5. Conclusions of the results

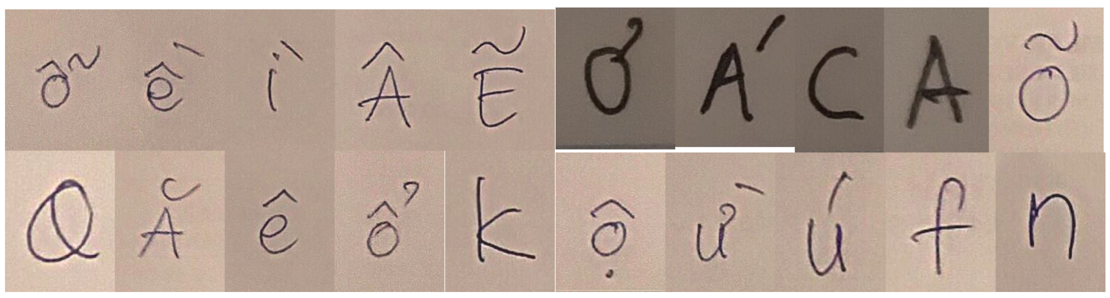
First and foremost, there are 4 models have been trained in this practical work, with the same model but different epoch values (20-100-500-1000). As result, the accuracy scores of four models are 69.89, 80.96, 82.2, 80.56 respectively.

To test this model, some images had to be opted randomly by following some robust rules, such as, it has to contain at least one capitalize and/or one lowercase letter, it should be mixed with letters having tones (à, á, ả, ã, ạ). All the unidentified images will be put inside the **unidentified_images** folder. Run the file **predict-images.py** to perform the prediction, all pictures will have pre-processing to thresholding images. Finally, the resulting images are labeled and appear in the **identified_images** folder.

PICTURE 5.a A process of prediction handwritten letter (147:**ổ**) with Otsu's Binarization and CNN

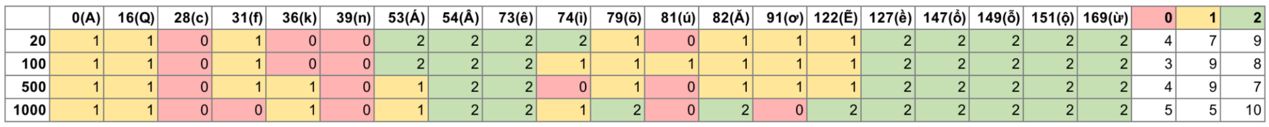


PICTURE 5.b selected images



After opting for some new handwritten letters for this prediction, to evaluate the model correctly, there is a scoring rule for correct prediction. For instance, if there is only a letter or a tone is correct, the model will have 1 point, if both letter and tone are correct, it will gain 2 points. Hence, the results of four models are:

FIGURE 5.c Table of results



	0(A)	16(Q)	28(c)	31(f)	36(k)	39(n)	53(Á)	54(Â)	73(ê)	74(ì)	79(õ)	81(ú)	82(Ă)	91(ơ)	122(Ẽ)	127(è)	147(ổ)	149(ỗ)	151(ộ)	169(ừ)	0	1	2
20	1	1	0	1	0	0	2	2	2	2	1	0	1	1	1	2	2	2	2	2	4	7	9
100	1	1	0	1	0	0	2	2	2	1	1	1	1	1	1	2	2	2	2	2	3	9	8
500	1	1	0	1	1	0	1	2	2	0	1	0	1	1	1	2	2	2	2	2	4	9	7
1000	1	1	0	0	1	0	1	2	2	1	2	0	2	0	2	2	2	2	2	2	5	5	10

In conclusion, the model achieved an overwhelming performance for predicting handwritten letters while Vietnamese letters are extremely complex and the lack of images in the dataset. Furthermore, this trained model is a tiny first step for building OCR (Optical Character Recognition) engine. In real scenario, engineers usually combine CNN with RNN (Recurrent Neural Network) and CTC (Connectionist Temporal Classification) to enhance the prediction accuracy of the system.