Green Intelligent Systems and Applications

Research Article
Volume 4(1), 2024, 41–53
https://doi.org/10.53623/gisa.v4i1.417

Transcribing Handwritten Medical Prescription using Convolutional Neural Network AlexNet Architecture and Canny Edge Detection

Ralph Andrei A. Benitez[1], Donata D. Acula[1,2]*, Anton Oliver M. Bondoc[1], Angelo Louis L. Hizon[1], Aaron Joseph D. Santos[1]

[1]Department of Computer Science, College of Information and Computing Sciences, University of Santo Tomas, Manila, Philippines
[2]Research Center for Natural and Applied Sciences, University of Santo Tomas, Manila, Philippines

SUBMITTED: 1 March 2024 ; REVISED: 25 May 2024; ACCEPTED: 28 May 2024

ABSTRACT: Misinterpreted medical prescriptions had led to casualties due to the illegible cursive handwriting of medical practitioners. Many studies focused on this problem. However, the accuracy was unsatisfactory and needed improvement. The study evaluated the performance of the Canny edge detection with other preprocessing methods, including RGB to Grayscale Conversion, Binarization, and Inversion, which was used to process the images of cursive handwritten medical prescriptions using Alexnet Convolutional Recurrent Neural Network (ACoRNN). The CRNN model developed by previous researchers was used as the basis for comparison, and the researchers created a faster and more accurate model. The best combination of preprocessing methods for ACoRNN was with RGB to Grayscale Conversion, Binarization, Canny edge detection, and Inversion. The researchers' model had faster preprocessing and testing time and achieved 90.76% average accuracy through five trials.

KEYWORDS: Machine learning; handwritten text recognition; convolutional neural network; canny edge detection; medical prescription; AlexNet

1. Introduction

Doctors are often attributed together with their illegible cursive handwriting in medical prescriptions. An article published by the Philippine Council for Health Research and Development (PCHRD) concluded that 28% of Filipino patients could not read their prescriptions. Cerio, Mallare, and Tolentino [1] and Carino, Divinagracia, Reyes, Sia, and Sydiongco [2] stated long exhaustive hours, stress, and the rush of peak hours forced doctors to work fast and get fatigued, which caused some doctors to write illegible prescriptions without any intention of harming their patients.

Although without any malicious intentions, misinterpretation of prescriptions was caused by illegible cursive handwriting, which could lead to improper dosage of medicine and cause fatal injuries or death [1, 3]. Pasco [4] conducted a study at the Philippine General Hospital, which concluded that the most common errors across medical departments are prescribing

errors wherein illegible handwriting leads to incorrect drug selection and dosage due to the misinterpretation of the patient's prescription.

Studies have analyzed cursive handwriting in medical prescriptions using CNN and BLSTM as their feature extractor and sequence labeler, which produced an accuracy of 72% [5] and 79.05% [6]. The difference between the two studies is the CNN architecture and its preprocessing model, wherein Cabais et al. (2021) used a smaller architecture and added another preprocessing method, edge detection.

The study's main objective is to develop an offline cursive handwriting recognition system for handwritten medical prescriptions using Convolutional Neural Network (CNN) AlexNet architecture and Canny Edge Detection. Specifically, the study aims to evaluate the accuracy and efficiency of the handwriting recognition system in training and testing periods where CNN AlexNet architecture and Canny Edge Detection are used, determine the optimal parameters needed for Canny Edge Detection and CNN AlexNet architecture to achieve the best performance of the system, and to test if there is a significant improvement in the performance of the system when CNN AlexNet architecture and Canny Edge Detection are implemented compared to CNN-LSTM and Compass Edge Detection [6].

With this information in mind, the researchers explored an offline cursive handwriting text recognition (HTR) system for transcribing medical prescriptions using CNN AlexNet and Canny Edge Detection. This AI-driven transcription of handwritten medical prescriptions decreases paper waste and boosts healthcare efficiency. This aligns with green technology principles by reducing resource use and improving accessibility while contributing to environmental sustainability.

2. Related Works

2.1. Cursive handwritten text recognition.

Cursive handwriting still proved challenging for many years because of the variability of an individual's writing [7, 8]. However, this does not mean recognition systems can't recognize cursive words. Instead, recognition systems are still able to consistently produce accurate results for cursive text inputs that are consistent and structured, such as using fonts. A study conducted by Mirza and Siddiqi [9] on various news channels containing Urdu script that uses fonts produced a character recognition rate of 97.63% on 40,000 text lines. Another study conducted by Srivastava, Priyadarshini, Gopal, Gupta, and Dayal [10] using the standard alphabet and digit dataset on bank checks produced 98% accuracy for numerical digits, 97% for letters, and 95.71% for words. When cursive words are written or styled consistently, models and algorithms detect patterns and features effectively and efficiently.

2.2. AlexNet.

AlexNet was designed by Krizhevsky, Sutskever, and Hinton [11]; where it is known for its similarities to LeNet by Yann LeCun et al., except AlexNet has more depth, more convolutional layers and more filters applied in comparison to LeNet. It extracts features from an image through 8 layers, specifically through 5 convolutional layers and 3 fully-connected layers. Convolutional layers serve as a filter smaller than the input image; features are then extracted via convolution, which are matrices with weight. Max-pooling layers reduce the input

dimension by extracting the most prominent features within the matrix; in the case of max-pooling, it gathers the highest or lowest value [12, 13].

2.3. Convolutional recurrent neural network.

Convolutional Recurrent Neural Network (CRNN) models or architectures were the basis for the researchers' creation of their CRNN architecture to recognize illegible cursive handwriting of doctors' medical prescriptions. Studies conducted by [6, 15], and [5] produced various CRNN architectures for text recognition systems. Also, the researchers will use the results of the mentioned studies to compare and contrast the future results of their CRNN architecture. In the study of [6], they utilized CRNN to develop a system that recognizes the illegible handwriting of doctors. The structure of their model is similar to the previous model [5], but the main differences between these models are the preprocessing methods, the RNN structure used, and the number of CNN layers [6]. utilized. Cabais et al.'s system architecture consists of 1.) Preprocessing methods, namely RGB to Grayscale, Noise Removal, Binarization, and Compass Edge Detection; 2.) The CNN of 5 layers is used for feature extraction; 3.) The RNN structure, BLSTM, and lastly, 4.) The CTC predicts and connects the sequence of inputs that the CRNN produces. The system developed by [6] yielded an overall accuracy of 79.05% on the testing dataset, better than the base system of [5].

3. Methodology

3.1. Dataset.

The data collected are the cursive handwritten medical prescriptions written by medical professionals. As such, the researchers have set conditions for collecting and deciding if the data collected from respondents are valid. These conditions are as follows: (1) Handwriting must be cursive; (2) The medical professionals have to write the 12 preset prescriptions; and (3) Handwriting must be incomprehensible to readers. To increase the data collected from each respondent, the researchers implemented the same data collection used by previous studies [5, 6]. The researchers set three (3) trials, with the 3rd trial being optional for the respondents to write each medical prescription. The three trials (3) are as follows: (1) Writing normally with no pressure; (2) Writing each prescription with a 10-second timer two times. The collected data will then be scrutinized and added to the Cabais dataset to increase the size of the existing dataset. Additionally, the data must be resized to the same dimensions as the Cabais' dataset, with 1024x128 pixel dimensions. The whole dataset will then be split into 60, 10, and 30 percent for training, validation, and testing datasets. To avoid an uneven split of medical prescriptions per dataset, the researchers implemented stratified random sampling.

3.2. Hypotheses.

To provide more details to support the results of the experiment, the following hypotheses were developed and tested: (1) There is a significant improvement in the performance of Canny Edge Detection compared to Compass Edge Detection in classifying cursive handwritten medical prescriptions, (2) There is a significant improvement in the performance of CRNN using AlexNet architecture with Canny Edge Detection compared to the CRNN model by [6] with Compass Edge Detection in classifying cursive handwritten medical prescriptions, and (3) There is a significant improvement in the performance of the proposed AlexNet CNN

architecture compared to the CNN architecture used by [6] using the same edge detection algorithm for classifying cursive handwritten medical prescriptions.

3.3. Preprocessing.

To achieve the best possible result of training, testing, and validating the models with the combined dataset of Cabais et al. and the researchers' collected data, the combination of preprocessing methods enumerated in Table 1 was used. The preprocessing methods that were utilized in this study were the following: (1) RGB to Grayscale Conversion, (2) Binarization, (3) Inverse, (4) Canny Edge Detection, and (4) Compass Edge Detection. By using the mentioned preprocessing methods, the seven combinations of preprocessing methods were formed, namely, All Preprocessing (Black) Canny Edge Detection, All Preprocessing (White), No RGB to Grayscale Conversion, Canny Edge Detection, No Binarization, Canny Edge Detection, No Canny Edge Detection, Inverse Binarization, and Compass Edge Detection.

Table 1. Preprocessing combinations.

Preprocessing methods	Preprocessing Techniques Used
(1) All Preprocessing (Black) Canny Edge Detection	RGB to Grayscale Conversion, Binarization, Canny Edge Detection
(2) All Preprocessing (White)	RGB to Grayscale Conversion, Binarization, Canny Edge Detection, Inverse
(3) No RGB to Grayscale Conversion, Canny Edge Detection	Binarization
(4) No Binarization, Canny Edge Detection	RGB to Grayscale Conversion
(5) No Canny Edge Detection	RGB to Grayscale Conversion, Binarization
(6) Inverse Binarization	RGB to Grayscale Conversion, Binarization, Inverse
(7) Compass Edge Detection	RGB to Grayscale Conversion, Binarization, Compass Edge Detection

3.4. System architecture.

The researchers created an improved Cursive Handwritten Text Recognition system developed using NumPy, Keras, and TensorFlow libraries. The system architecture above shows the three (3) phases of the system, namely (a) the Input phase, (b) the Preprocessing phase, and (c) the Recognition phase. As indicated by the broken lines in Figure 1, it can be seen that the researchers intend to use a different approach to the Preprocessing phase and feature extraction compared to what was used in Cabais et al.'s (2021) recognition system.

In the input phase, the researchers manually cropped and resized the images before splitting them into three (3) datasets: (a) training, (b) testing, and (c) validation datasets. After splitting the whole dataset, the training dataset was the only one that had undergone data augmentation, which increased its size. The three (3) datasets were moved on to the Preprocessing phase. In the Preprocessing phase, the images inside the three (3) datasets the researchers used different preprocessing combinations. The preprocessing methods that were used are the following: RGB to Grayscale Conversion, Binarization, Canny Edge Detection, Inversion, and Compass Edge Detection. The stated preprocessing methods were similar to the preceding researchers, where Cabais et al. used RGB to Grayscale Conversion, Binarization, Canny Edge Detection, Compass Edge Detection, and Skeletonization.

After the images had undergone preprocessing, the preprocessed datasets were moved on to the Recognition phase, which used the preprocessed images to train and test the model,

which consists of the CNN, BLSTM, and CTC layers. The researchers used AlexNet as the feature extractor instead of creating one from scratch.

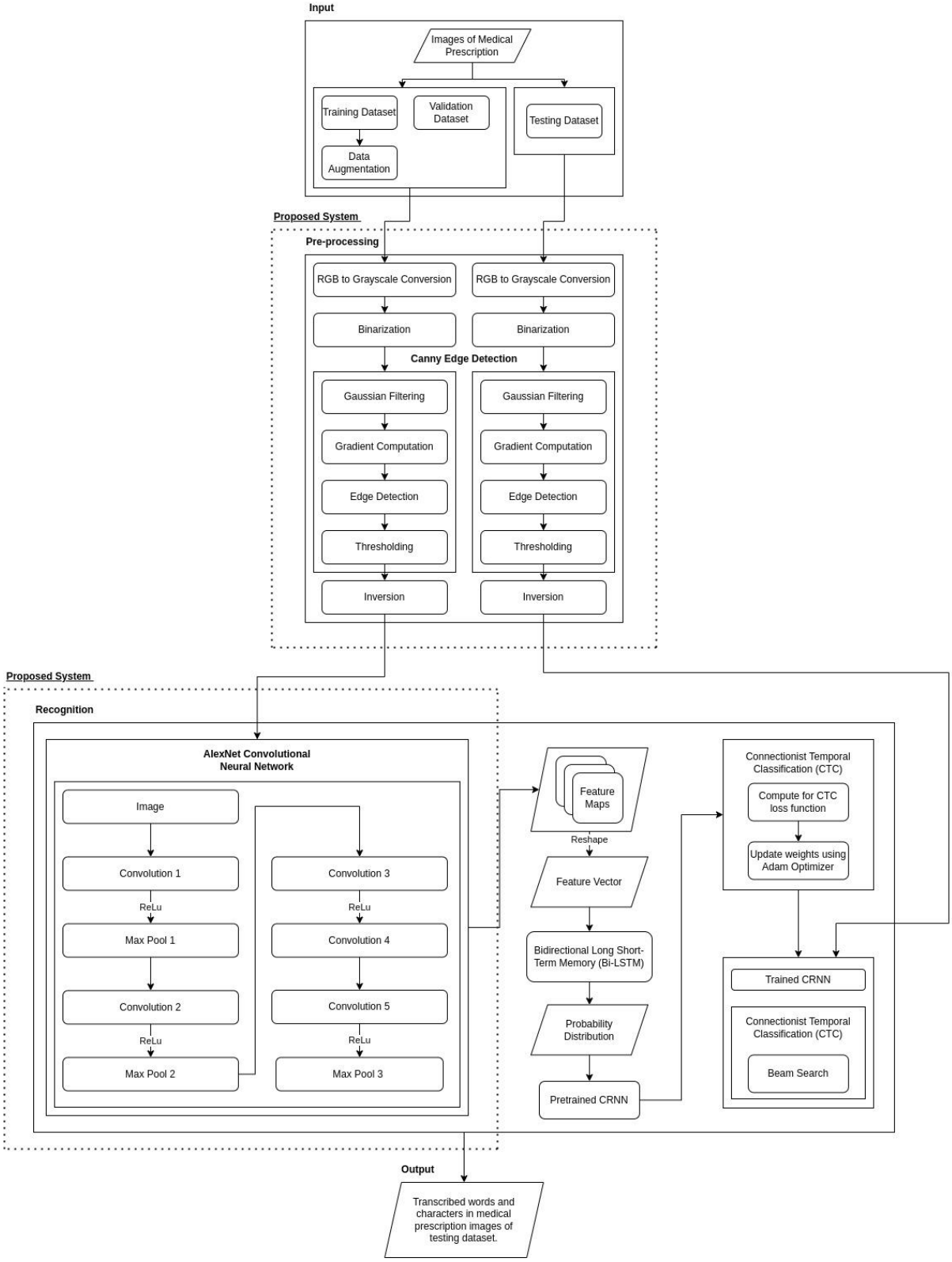

Figure 1. System architecture of the study.

3.5. Statistical metrics and tools.

To obtain a fair comparison between the performances of the Cabais model and the ACoRNN model, several statistical techniques were used to obtain numerical results for comparison. The researchers used the following statistical tools to obtain these metrics: (1) Python and (2) R Studio. The statistical metrics that were used were the following:

$$\text{Character Error Rate (CER)} = \left[\frac{(i+s+d)}{n}\right] * 100 \quad (1)$$

$$\text{Word Error Rate (WER)} = \left[\frac{(i+s+d)}{n}\right] * 100 \quad (2)$$

Where s = No. of misspelled character/s or words; d = No. of missing character/s or words; i = No. of incorrect character/s or words; n = No. of characters in reference text

$$\text{F1 Score} = 2 * \frac{precision * recall}{precision + recall} \quad (3)$$

$$\text{Precision} = \frac{\#\,of\ correct\ predictions}{\#\,of\ all\ recognized\ words} \quad (4)$$

$$\text{Recall} = \frac{\#\,of\ correct\ predictions}{\#\,of\ all\ expected\ words} \quad (5)$$

$$\text{Accuracy} = \frac{\#\,of\ correct\ predictions}{\#\,test\ instances} \quad (6)$$

Additionally, this research used other statistical techniques to test the hypotheses, to either accept or reject the null hypothesis for each case. These statistical techniques are the Paired Sample T-Test, the GESD Test, the Shapiro Wilk Test for Normality, and the Wilcoxon Signed Rank Test.

4. Results and Discussion

4.1. Presentation of training results.

The researchers used Google Colaboratory to train the four (4) models with the given training data. The Cabais dataset sums up to 2,064 images with augmentation for training, 48 for validation, and 144 for testing. The Cabais and Benitez dataset sums up to 4,128 images with augmentation divided into 3,744 images for training, 96 for validation, and 288 for testing. Table 2 shows the split and distribution of the two dataset collections of the twelve (12) prescriptions.

Table 2. Image distribution and dataset split for Cabais and Benitez datasets.

Data Collection	Number of Training Images	Number of Validation	Number of Testing
Cabais Dataset	1872	48	144
Cabais and Benitez Dataset	3744	96	288

The images were split with a ratio of 60:10:30, which were training, validation, and testing, respectively. The data was split first before augmenting the training dataset. The general training configuration of hyperparameters of the four architectures was displayed above, ACoRNN, also known as the Alexnet Convolutional Recurrent Neural Network, and CCoRNN, also known as Cabais Convolutional Recurrent Neural Network. The researchers initially had the same hyperparameters as Cabais et al. (2021). After some trial and error, the researchers settled for 1000 epochs and increased the epochs of the EarlyStop and

ReduceLearningRate triggers resulting in 20 and 15, respectively, in training ACoRNN architectures. CCoRNN architectures were trained following their previous training hyperparameters, as seen in Table 3. EarlyStopping is a function from Tensorflow.Keras API dictates whether the model should stop training after a set number of epochs if there is no improvement of validation loss; this prevents the model from overtraining, thus reducing the chances of overfitting. ReduceLearnRate formally known as ReduceLROnPlateau on Tensorflow.Keras API reduces the learning rate of the model while training after a set amount of epochs if there is no improvement in validation loss. The learning rate is reduced by 0.001 every time the function is triggered.

Table 3. Training hyperparameters.

Hyperparameters	CCoRNN Architecture	ACoRNN Architecture
Input Size	1024 x 128	
Batch Size	16	
Training Epochs	200	
Initial Learning Rate	0.01	
Optimizer	Adam	
ReduceLearnRate trigger	After 10 epochs	After 15 epochs
EarlyStopping trigger	After 15 epochs	After 20 epochs

The training results were obtained from averaging the five trials done on the 7 versions of the dataset. (1) All Preprocessing (Black), (2) All Preprocessing (White), (3) No RGB to Grayscale Conversion, Canny Edge Detection, (4) No Binarization, Canny Edge Detection, (5) No Canny Edge Detection, (6) Inverse Binarization, and (7) Compass Edge Detection, which was Cabais' best combination of preprocessing methods.

Table 4. Summary of training results of ACoRNN and CCoRNN models.

Preprocessing Method (No.)	Ave. Train Loss		Ave. Val Loss		Ave. Train Time (hh:mm:ss)	
	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN
1	3.34	1.07	6.08	5.77	0:21:29	0:22:39
2	0.64	0.69	3.74	5.58	0:24:03	0:18:41
3	1.53	0.349	6.30	5.04	0:19:40	0:22:41
4	2.15	0.345	5.03	4.67	0:21:15	0:19:57
5	0.91	0.54	5.10	5.25	0:21:13	0:15:42
6	1.25	0.346	4.10	4.64	0:18:52	0:21:04
7	1.72	0.81	3.98	4.25	0:18:57	0:19:49

As shown in Tables 4, the training results vary from the combination of preprocessing methods used and the model that it was trained on. For the ACoRNN model, it generated the lowest train and validation loss using the (2) All Preprocessing (White) dataset. On the other hand, the CCoRNN model achieved its lowest losses on different combinations of preprocessing methods.

4.2. Presentation of testing results.

The testing results presentation was obtained with the same means as the training results. The presentation of the data was similar to maintain consistency. From Table 5, the testing results vary from the combination of preprocessing methods used and the model that it was trained on. For the ACoRNN model, it generated the best overall testing performance using the (2) All Preprocessing (White) dataset. On the other hand, the CCoRNN model achieved its best overall performance using the (3) No RGB to Grayscale Conversion, Canny Edge Detection dataset.

Table 5. Summary of testing results of ACoRNN model.

Preprocessing Methods (No.)	CER (%)		WER (%)		F1-Score (%)		Precision (%)		Recall (%)		Accuracy (%)	
	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN
1	2.97	3.62	5.85	8.85	94.16	90.99	94.05	91.22	94.3	90.78	83.89	73.12
2	1.79	3.90	3.61	9.30	96.27	90.58	96.12	90.63	96.4	90.55	90.76	70.9
3	2.33	2.87	4.96	7.25	94.95	92.62	94.83	92.78	95.1	92.47	85.56	76.1
4	2.11	3.54	4.01	8.81	95.88	91.05	95.77	91.32	96.01	90.81	89.52	70.56
5	2.59	3.13	5.02	7.73	94.89	92.22	94.74	92.41	95.04	92.05	86.60	74.79
6	2.93	2.90	5.66	7.33	94.29	92.60	94.16	92.74	94.43	92.48	85.90	75.56
7	2.51	3.20	4.85	8.11	95.09	91.81	94.94	91.71	95.2	91.95	87.64	75.28

4.3. Statistical results.

To calculate the accuracy, the researcher gathered the result and transformed the prediction into numbers where it can have the values of 0 or 1. The value of zero (0) indicates that the prediction of the model is incorrect while one (1) is a correct prediction. The total running time of the two models will be separated into three where the first one is the preprocessing time, second is the training time, and the third one is the testing time. To compute the difference between the accuracy and running time there is a 5% level of significance and 95% confidence in all tests. The first hypothesis is the comparison between preprocessing methods where it includes the comparison of the following: all preprocessing white with fine tuned ACoRNN, compass edge detection with fine tuned ACoRNN, all preprocessing white with fine tuned CCoRNN, and Compass edge detection with fine tuned CCoRNN. The researchers used Shapiro Wilk's Test that is available in scipy.stats in Python to determine whether the data is normally distributed or not. Based on the result as seen in Table 6, the researchers should reject the null hypothesis. which leads to the conclusion that GESD is not applicable for this scenario. Since the first dataset is not normally distributed and has outliers then it does not meet the requirements for Paired Sample T-Test therefore the researchers used Wilcoxon Signed Rank Test. The accuracy of the compass edge detection as base model and all preprocessing white as proposed model both with Fine Tuned ACoRNN is tested whether there is a significant improvement.

Table 6. Shapiro Wilk and Wilcoxon test for all preprocessing white and compass edge detection with fine tuned ACoRNN and CCoRNN.

Test and Results	Shapiro Wilk Test		Wilcoxon Sign Rank Test	
	ACoRNN	CCoRNN	ACoRNN	CCoRNN
Alpha	0.05	0.05	0.05	0.05
P-Value	6.9979e-22	8.7393e-15	0.0012	0.9998
Result	6.9979e-22 < 0.05	8.7393e-15 < 0.05	0.0012 < 0.05	0.9998 > 0.05

Proceeding with the second half of the first hypothesis the one that will be compared is compass edge detection and all preprocessing white with fine tuned CCoRNN model. The same hypothesis and process will be followed. Likewise, the null hypothesis is rejected meaning the dataset is not normally distributed. With the compass and all preprocessing white with fine tuned CCoRNN dataset not passing the requirements for Paired Sample T-Test then Wilcoxon-Signed Rank Test will be used. The hypothesis were:

H_{o}: All Preprocessing White ≤ Compass Edge Detection

H_{a}: All Preprocessing White > Compass Edge Detection

With the result given then the decision of the researchers is to reject the null hypothesis furthering support the fact that Compass Edge Detection with Fine Tuned ACoRNN has a respectable average accuracy of 87.64% while all preprocessing white with fine tuned ACoRNN has an average accuracy of 90.76% which was shown in Table 6. In addition, the researchers do not reject the null hypothesis because the average accuracy rate rate of Compass Edge Detection with fine tuned CCoRNN is 75.28% whereas all preprocessing white has average accuracy of 70.906% which was presented in the same table above.

Table 7. Compass edge detection with fine tuned CCoRNN and all preprocessing white with fined tune .ACoRNN- Cabais and benitez dataset.

Result	Trial 1		Trial 2		Trial 3		Trial 4		Trial 5		Sum	
	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN	CCoRNN	ACoRNN
1	212	255	225	277	220	263	225	258	192	254	1074	1307
0	76	33	63	11	68	25	63	30	96	34	366	133

The second hypothesis is the head-to-head comparison between all preprocessing white with fine tuned ACoRNN and Compass Edge Detection with fine tuned CCoRNN. The researchers added the number of times the model correctly and incorrectly predicted a medical prescription, as seen in Table 7. Through the Shapiro Wilk test, it can be concluded that the result is not normally distributed as shown in Table 8. Thus, Wilcoxon Signed Rank Test was used to test the hypotheses:

H_{o}: All Preprocessing White with Fine Tuned ACoRNN ≤ Compass with Fine Tuned CCoRNN

H_{a}: All Preprocessing White with Fine Tuned ACoRNN > Compass with Fine Tuned CCoRNN

Table 8. Shapiro Wilk and Wilcoxon test results for all peprocessing white with fine tuned ACoRNN and compass edge detection with fine tuned CCoRNN.

Test and Results	Shapiro-Wilk	Wilcoxon
Alpha	0.05	0.05
P-Value	5.25450e-18	1.1670e-19
Result	5.2550-18 < 0.05	1.1670 > 0.05

Based on the same table, the null hypothesis was rejected since the p-value is less than alpha. The third hypothesis is the comparison between the same preprocessing methods but different models for handwritten text recognition (HTR). The following combinations include the following: all preprocessing white with fine tuned ACoRNN, all preprocessing white with fine tuned CCoRNN, Compass Edge with fine tuned ACoRNN, and Compass Edge with fine tuned CCoRNN. First half of this hypothesis will compare all preprocessing white with Fine Tuned ACoRNN and all preprocessing white with Fine Tuned CCoRNN. The decision for Shapiro Wilk Test is to reject the null hypothesis and conclude that the data is not normally distributed then using Wilcoxon Signed Rank Test will be used to validate the following hypotheses:

H_o: Fine Tuned ACoRNN ≤ Fine Tuned CCoRNN

H_a: Fine Tuned ACoRNN > Fine Tuned CCoRNN

The decision is rejecting the null hypothesis since the P-Value is extremely close to the value of 0 wherein it showcases the huge difference between accuracies of the base model with an average accuracy of 70.906% and proposed model with an accuracy of 90.76% using the same all preprocessing white dataset. Second half of the third hypothesis will cover the comparison between compass edge detection with Fine Tuned ACoRNN and fine tuned CCoRNN as shown also in Table 9.

Table 9. Shapiro Wilk and Wilcoxon signed rank test for (a) all preprocessing white with fine tuned ACoRNN and all preprocessing white with fine tuned CCoRNN (b) compass edge detection with fine tuned ACoRNN and compass edge detection with fine tuned CCoRNN.

Test and Results	All Preprocessing White with Fine Tuned ACoRNN and All Preprocessing White with Fine Tuned CCoRNN		Compass Edge Detection with Fine Tuned ACoRNN and Compass Edge Detection with Fine Tuned CCoRNN	
	Shapiro-Wilk	**Wilcoxon**	**Shapiro-Wilk**	**Wilcoxon**
Alpha	0.05	0.05	0.05	0.05
P-Value	5.25450e-18	4.0661e-25	1.6235e-16	1.8062e-15
Result	5.2550-18 < 0.05	4.0661 > 0.05	1.6235e-16 < 0.05	1.8062e-15 > 0.05

H_o: Compass with Fine Tuned ACoRNN ≤ Compass with Fine Tuned CCoRNN

H_a: Compass with Fine Tuned ACoRNN > Compass with Fine Tuned CCoRNN

Since the p-value is less than alpha = 0.05, the null hypothesis was rejected. Supporting the statistical analysis that the average accuracy rate of Compass Edge Detection with fine tuned ACoRNN is 87.64% whereas Compass Edge Detection with fine tuned CCoRNN is 75.28%. Based on the result in Shapiro Wilk Test for normality preprocessing time and training time both have P-Values greater than 5% therefore it is considered as normally distributed. A possible reason why the difference between the dataset is normally distributed is because of its small size. However, before proceeding with Paired Sample T-Test the researchers must check if there are outliers and there is a way which is through Generalized Extreme Studentized Deviate (GESD or Grub's Test) where this can be accessed through a library known as PyAstronomy in Python (Table 10; Table 11).

Table 10. Shapiro Wilk and Wilcoxon signed rank test for time.

Test and Results	Shapiro Wilk			Wilcoxon Signed Rank		
	Preprocessing Time	**Training Time**	**Testing Time**	**Preprocessing Time**	**Training Time**	**Testing Time**
Alpha	0.05	0.05	0.05	0.05	0.05	0.05
P-Value	0.34290	0.74814	0.01730	0.03125	0.74814	0.03125
Result	0.34290 > 0.05	0.74814 > 0.05	0.01730 < 0.05	0.03125 < 0.05	0.74814 > 0.05	0.03125 < 0.05

Table 11. GESD for time.

Running Time	Number of Outliers	Outlier
Preprocessing Time	3	Trials 1,2,3
Training Time	3	Trials 1,2,3

Although preprocessing time and training time are normally distributed but both have outliers which means it does not pass the requirements of Paired Sample T-Test and will join testing time with Wilcoxon Signed Rank Test. The hypotheses were:

H_o: Compass with Fine Tuned CCoRNN ≤ All Preprocessing White with Fine Tuned ACoRNN

H_a: Compass with Fine Tuned CCoRNN > All Preprocessing White with Fine Tuned ACoRNN

The null hypothesis indicates that the compass has spent less running time in comparison to all preprocessing white and the opposite is true for the alternative hypothesis. Based on the results the researchers should reject the null hypothesis of preprocessing time and testing time. As for training time, the null hypothesis was not rejected given that the p-value is greater than the alpha = 0.05.

5. Conclusions

This research implemented a CRNN using the AlexNet CNN architecture along with Canny edge detection to develop a cursive handwritten recognition system for medical prescriptions and was able to address previous issues in preprocessing and training, and thus resulted in the following conclusions. After fine tuning the ACoRNN model, the optimal number of parameters in the model mentioned was 1,611,938. These configurations of the parameters and hyperparameters are adjusted according to the training losses, validation losses, and testing accuracies of the model. Thus, the comparison of the total parameters of the fine tuned models and the base models is the lower the parameters of a model, the better it fits the preprocessed dataset. The overall performance of fine tuned ACoRNN proved to be significant in accuracy compared to fine tuned CCoRNN. Fine tuned CCoRNN was slightly better in training time compared to fine tuned ACoRNN by 1 minute and 22 seconds. Comparing the performance from the study by Cabais et al. (2021), the proposed model of the researchers was 11.71% more accurate than the results of the previous study (90.76% > 79.05%). However, comparing the fine tuned CCoRNN model implementation to the researchers' model (ACoRNN) was 15.48% more accurate (90.76% > 75.28%). Based on the findings, Canny edge detection outperformed Compass edge detection in terms of accuracy, preprocessing time, and testing time regardless of CRNN architecture. However, Compass edge detection performed better in terms of training time. As for the recommendation, it is accepted that cursive handwriting recognition still remains a challenge as different writing styles of medical professionals affect the accuracy and performance of recognition systems. Although the researchers had developed a recognition system which followed the recommendation of past researchers, there was still room for improvements for future studies.

Acknowledgments

The authors would like to acknowledge the authors of the previous research paper entitled ”Analyzing Handwritten Texts in Medical Prescriptions using Compass Edge Detection and Convolutional Recurrent Neural Network” sharing their dataset which was used in the testing phase of the model. The authors would like to thank all the teaching academic staff of the Computer Science Department, University of Santo Tomas for reviewing and scrutinizing the content of this paper.

Competing Interest

Authors declare that no potential conflict of interest exists related to this article.

References

[1] Cerio, A.A.P.; Mallare, N.A.L.B.; Tolentino, R.M.S. (2015). Assessment of the Legibility of the Handwriting in Medical Prescriptions of Doctors from Public and Private Hospitals in Quezon City, Philippines. *Procedia Manufacturing*, *3*, 90–97. https://doi.org/10.1016/j.promfg.2015.07.112.

[2] Factors that Contribute to Medication Errors in the Philippine Heart Center. (accessed on 25 May 2023) Available online: https://www.phc.gov.ph/Images/articles/Factors%20that%20Contribute%20to%20Medication%20Errors.pdf.

[3] 1 in 3 misdiagnoses results in serious injury or death: study. *Fierce Healthcare*. (accessed on 25 May 2023) Available online: https://www.fiercehealthcare.com/hospitals-health-systems/jhu-1-3-misdiagnoses-results-serious-injury-or-death#:~:text=An%20estimated%2040%2C000%20to%2080%2C000.

[4] Pasco, P.M.D.; Caro, R.M.; Cruz, C.L.; Dando, N.M.; Isip-Tan, I.T.C.; Panganiban, L.R.; Pascua, L.P.; Ricalde, R.R.; Sison, A.C. (2017). Prevalence of Medication Errors in Admitted Patients at the Philippine General Hospital. *Acta Medica Philippina*, *51*, 2. https://doi.org/10.47895/amp.v51i2.577.

[5] Fajardo, L.J.; Sorillo, N.J.; Garlit, J.; Tomines, C.D.; Abisado, M.B.; Imperial, J.M.R.; Rodriguez, R.L.; Fabito, B.S. (2019). Doctor’s Cursive Handwriting Recognition System Using Deep Learning. 2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICE). https://doi.org/10.1109/hnicem48295.2019.9073521.

[6] Thavayee, A.; Vijayakumar, T. (2024). Analyzing Medical Manuscripts through Neural Networks and Handwriting Recognition Algorithms. *International Journal of Emerging Technologies and Innovative Research*, *11*, f52-f57.

[7] Sadaf, F.; Raju, S.M.T.U.; Muntakim, A. (2021). Offline Bangla Handwritten Text Recognition: A Comprehensive Study of Various Deep Learning Approaches. *IEEE Xplore*, 153–156. https://doi.org/10.1109/ICEEE54059.2021.9718890.

[8] Hassan, S.; Irfan, A.; Mirza, A.; Siddiqi, I. (2019). Cursive Handwritten Text Recognition using Bi-Directional LSTMs: A Case Study on Urdu Handwriting. *IEEE Xplor*e, 67–72. https://doi.org/10.1109/Deep-ML.2019.00021.

[9] Mirza, A.; Siddiqi, I. (2020). Recognition of cursive video text using a deep learning framework. *IET Image Processing*, *14*, 3444–3455. https://doi.org/10.1049/iet-ipr.2019.1070.

[10] Srivastava, S.; Priyadarshini, J.; Gopal, S.; Gupta, S.; Dayal, H.S. (2018). Optical Character Recognition on Bank Cheques Using 2D Convolution Neural Network. *Advances in Intelligent Systems and Computing*, 589–596. https://doi.org/10.1007/978-981-13-1822-1_55.

[11] Krizhevsky, A.; Sutskever, I.; Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks. *Communications of the ACM, 60*, 84–90. https://doi.org/10.1145/3065386.

[12] AlexNet: The Architecture that Challenged CNNs. *Medium*. (accessed on 25 May 2023) Available online: https://towardsdatascience.com/alexnet-the-architecture-that-challenged-cnns-e406d5297951.

[13] CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more. Medium. (accessed on 25 May 2023) Available online: https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5.

[14] Hassan, S.; Irfan, A.; Mirza, A.; Siddiqi, I. (2019). Cursive Handwritten Text Recognition using Bi-Directional LSTMs: A Case Study on Urdu Handwriting. *IEEE Xplore*, 67–72. https://doi.org/10.1109/Deep-ML.2019.00021.

[15] Chandio, A.A.; Asikuzzaman, M.D; Pickering, M.R.; Leghari, M. (2022). Cursive Text Recognition in Natural Scene Images Using Deep Convolutional Recurrent Neural Network. *IEEE Access*, *10*, 10062–10078. https://doi.org/10.1109/access.2022.3144844.

[16] Islam, M.H.; Tara, K.; Rahman, H.U.; Sarkar, A.K. (2021). An Approach to Assess ACh and NE Secretion inside Heart's Myocardial Cell during R-peak Formation. 2021 3rd International Conference on Electrical & Electronic Engineering (ICEEE), pp. 57-60. https://doi.org/10.1109/ICEEE54059.2021.9718857.