Sri Lankan Journal of Technology
Research

SLJoT
2024 Issue 01 13-24
ISSN-2773-6970

Efficiency and Accuracy in Sinhala Handwritten Character Recognition: A Gabor-initialized CNN Perspective

M.L. Karunarathne[1], C.P. Wijesiriwardana[1], K.M.I. Nishantha[2], and W.G.C.W. Kumara[3*]

[1]Department of Information Technology, Faculty of Information Technology, University of Moratuwa, Sri Lanka
[2]Department of Interdisciplinary Studies, Faculty of Engineering, South Eastern University of Sri Lanka, Sri Lanka
[3]Department of Computer Science and Engineering, Faculty of Engineering, South Eastern University of Sri Lanka, Sri Lanka

[*]Corresponding Author: chinthakawk@seu.ac.lk || ORCID: 0000-0002-4613-275X

Received: 29-12-2023. * Accepted: 18-04-2024 * Published Online: 30-06-2024

***Abstract*--**The Sinhala language is used as a national language only in Sri Lanka. The 60 characters in the Sinhala alphabet are complex in shape in comparison to a language like English. There is no solid solution yet for Sinhala handwritten character recognition in the pattern recognition domain due to the excessive shapes and many perplexing characters available in the Sinhala language. Performance is low for real-world practical applications with the current best character recognizers. Two methods were experimented with in this paper, namely, a Convolutional Neural Network (CNN) and a Gabor-initialized CNN (GCNN). Further, the effect of introducing the dropout layer in both the proposed network architectures is investigated. Gabor filter parameters are examined individually to understand their impact of using GCNN for Sinhala handwritten character recognition. We determine the parameter values of the introduced GCNN architecture considering their impact on the model. 96.33 % training accuracy and 90.14% testing accuracy achieved in CNN model 1 with a 0.5 dropout value is the highest accuracy for 60 characters in Sinhala as per the current literature. Training and testing accuracy of the GCNN architecture are 95.15% and 80.00% consecutively. GCNN converges faster than the CNN model 1 saving computational time and cost even though the training accuracy of GCNN is nearly 1% less than the CNN model 1. Hence, considering the accuracy and the processing speed the optimum method could be used for a live implementation of the work Sinhala character recognition application.

Keywords—handwritten Sinhala character recognition, Convolutional Neural Networks, Gabor filter bank, Gabor initialized CNN

I. INTRODUCTION

Automatic handwritten character recognition has recently become a challenging research area in pattern recognition, providing both educational and

Commercial value. The primary difficulty in recognising handwritten characters is handling different handwriting styles of individuals in various languages. In languages like Japanese, Laos, and Thai, isolated characters from each other are used. In contrast, in languages like Sinhala, English, and Bengali, characters are curved, complex, and sometimes the characters are connected. Hence, automatic recognition of handwritten characters is more challenging than printed characters. Nevertheless, character recognition is worsened by factors like various shapes of the characters, overlapping nature, and consecutive character overlapping.

Sinhala alphabet contains 60 letters with vowels and consonants (**Error! Reference source not found.**). Most Sinhala characters are circular and have minimal vertical or horizontal lines. Even though each Sinhala character is unique in shape, some characters are different from the similar ones with minor variations. Every letter can be combined with 15 modifiers in the Sinhala resulting in more than 1,000 letters with different pronunciations. There are no simple and capital letter variations in Sinhala like in English. The script is written from left to right and the writing system is syllabic. Characters in a word are written separately from each other in a non-cursive manner. Recognition of characters in Sinhala is a difficult task for regular simple character recognition methods due to the availability of many characters with a similar shape in Sinhala. Currently, many use cases demand highly accurate Sinhala handwritten character recognition such as post office automation, banking automation, and many more.

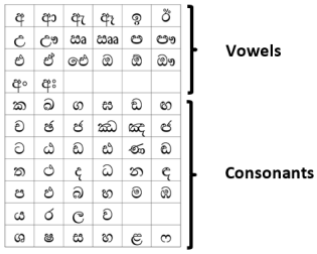


Figure 1. Sinhala alphabet

Handwritten character recognition (HCR) is a process that facilitates the conversion of different kinds of handwritten documents into analyzable, editable, and searchable documents. The invention of the HCR mechanism has obtained notable attention from many researchers over the past, and many significant achievements have been made in this domain (Hewavitharana et al., 2002). However, HCR is still challenging due to the variety of handwriting styles, similar characters, and different character categories (Rajapakse et al., 1995).

CNN is used in this study due to its well-defined architecture and in-built feature extraction. CNN utilizes the convolution of images and filters to create invariant features that are transferred to the next layer and work well on image data. The computation cost of CNN is low due to minimal data preprocessing.

Gabor filters are used in image processing for pre-processing, edge detection, feature extraction, texture analysis, and feature modification (Petkov and Wieling, 2008). Gabor filters behave like human vision systems and are good for texture analysis tasks, separating texture information from images, hence applied to many applications such as face, speech, and vein recognition. Although CNN filters can extract higher-dimensional complex features, it has the drawbacks of overfitting and hindering generalization capability for a relatively low number of training samples. Since texture features can be well obtained a Gabor filter bank is used in this research instead of a trainable convolutional layer as the initial layer of the CNN. The use of a non-trainable Gabor filter bank as the first layer in CNN provides better accuracy on small datasets with no overfitting and better generalization on unseen data (Yuan et al., 2022). Further, Gabor filters with CNN provide accurate results and significant improvement in convergence for handwritten character recognition applications (Gu et al., 2018) and the GCNN model increased the performance in terms of specificity and sensitivity (Petkov and Wieling, 2008). In comparison to different initialization methods, GCNN significantly outperformed even with noisy data and small training data size.

This research aims to develop a model to recognize Sinhala handwritten text using deep learning techniques. CNN is examined with dropout (*D*) and Gabor filters to recognize the handwritten Sinhala character and a comprehensive comparison of the above approaches is performed to find out the best.

The rest of the paper is organized as follows. Chapter 2 summarizes the literature review, chapter 3 explains the proposed methodology, and chapter 4 presents the results and discussion.

II. LITERATURE REVIEW

Handwritten character recognition is a branch of pattern recognition. Handwritten character recognition in languages like Sinhala is difficult compared to English due to the character curviness, size, and shape which depends on the individual writing speed, age, literacy, and writing style (Alom et al., 2018) (Wang and Yu, 2019).

Since 90's several approaches have been used for Sinhala handwritten character recognition, such as CNN (Wasalthilake and Kartheeswaran, 2020), Neural Networks (Rajapakse et al., 1995), Hidden Markov Model (Hewavitharana et al., 2002), Projection file methods (Nilaweera et al., 2007), and Contour tracing method (Silva et al., 2015).

Most of the previous researchers used their datasets and had limitations such as a limited number of characters, algorithm complexity, less accuracy, and not being able to recognize touching characters. The proposed research hopes to address these limitations with higher accuracy.

A. CNN

CNN can detect image features and patterns and is suitable for image classification applications in computer vision. In 1980, Fukushima used CNN first (O'Shea and Nash, 2015) and LeCun et al. in the 1990s obtained successful results by using CNN with a gradient-based learning algorithm (O'Shea and Nash, 2015). CNN can extract and abstract two-dimensional features. Shape variations are well captured by the CNN max-pooling layers. The diminishing gradient problem can be solved by minimizing errors while using a gradient-based learning algorithm with CNN. Image spatial dependencies associated with image content are acquired by CNN for image recognition. The input image to feature vector transformation during training updates parameters, weights, and biases to better understand the image's nature (Albawi et al., 2017).

i. Convolution

Feature enhancement and sharpening are two examples of image manipulation which can be achieved with convolution. In convolution, a matrix (kernel or filter) passes through each pixel of an image and the pixel value is changed accordingly (Gu et al., 2018). The sub-matrix surrounded at each pixel is multiplied by the kernel and the original pixel value is replaced by the summation of the multiplication (Albawi et al., 2017). This operation should be carried out continuously for the whole image. Convolution

acquires low-level features such as color, border, orientation, and gradient and eliminates high-level features such as edges. Hence, this method reduces input image dimensionality (Wu, 2017).

ii. Pooling

Pooling is the second important CNN component. Pooling gradually decreases computational complexity within the network by increasing the spatial scale of the image description. Maximum, minimum, average, and adaptive pooling are several non-linear functions used in pooling implementation with maximum pooling (max-pooling) being the most common approach. Max-pooling reduces dimensionality resulting in low computing power while retaining important rotational and translational variants.

Max-pooling extracts the maximum from the image portion encompassed by the kernel while average-pooling better suppresses noise. It discards noisy activations and performs de-noising along with dimensionality reduction (Gu et al., 2018). The pooling layer controls overfitting by reducing parameter count, memory usage, and computation. In CNN, a pooling layer is used between consecutive convolutional layers with an activation function. CNN Pooling does not facilitate global translation invariance, instead contributes to a local translation invariance in the form of global pooling (Albawi et al., 2017). The pooling resizes input spatially by independently applying on every depth or slice of the input.

iii. Fully connected layer

A fully connected layer is used at the end of CNN. CNN weights are calculated during training. Based on the convolution and pooling layer results fully connected layer determines the best-matched label that represents the original image. Hence, the connection of the image feature vector with the class of the image is decided by the fully connected layer. Convolution or pooling layer outputs are multiplied by the network weights. Finally, the result is passed through an activation function (O'Shea and Nash, 2015).

B. Dropout for regularizing

Deep learning neural networks are expected to overfit quickly as fully connected layers engage in most of the parameters. Ensembles of neural networks with various model configurations are facilitated to decrease overfitting, but it needs extra computational effort (Jain, 2018). Dropouts are randomly dropping out nodes during training which provides an efficient and effective regularization reducing overfitting and improving generalization error in DNNs. By preventing the training of all nodes on the training data set, dropout decreases overfitting. This method remarkably upgrades training speed. The dropout technique helps to reduce inter-node interactions by facilitating the learning of more powerful features that better generalize to new data fields (Anello, 2021). Each neuron in layers behaves independently in comparison to others due to the application of the dropout effect in the model. Even though the increased training time is disadvantageous, the significant increase in accuracy, especially for larger datasets is more advantageous.

C. Gabor filters

Gabor filter analyzes the image to capture any specific frequency content in specific directions in a localized region around the point of analysis (Gabor, 1946). Gabor filters are formulated using a Gaussian kernel function modulated by a sinusoidal wave. The Gabor filter's complex equation is,

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\exp\left(i\left(2\pi\frac{x'}{\lambda}+\psi\right)\right), \quad (1)$$

where, its real and imaginary parts are,

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda}+\psi\right), \text{ and} \quad (2)$$

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\sin\left(2\pi\frac{x'}{\lambda}+\psi\right), \quad (3)$$

where, λ is the wavelength of the sinusoidal factor, θ is the orientation of the normal to the parallel stripes of a Gabor function, and ψ is the phase offset, σ is the standard deviation of the Gaussian envelope, and γ is the spatial aspect ratio and specifies the ellipticity of the Gabor function. Equations (4) and (5) represent what controls the center frequency of the Gabor filter representing the filter's highest response.

$$x' = x\cos\theta + y\sin\theta,$$
$$y' = -x\sin\theta + y\cos\theta.$$

The Gabor filter's power spectrum is formulated by two sine wave impulses and a Gaussian. As we know, frequency domain convolution is equal to the spatial domain multiplication. However, since the uncertainty principle reveals the impossibility of precisely determining both the frequency of a particle and a snapshot of time, accurate measurement of its counterpart becomes challenging (Alekseev and Bobe, 2019). In the application of analyzing texture or obtaining features from an image, a bank of Gabor filters with several different orientations should be applied (Wu, 2021). When Gabor filters are applied to

an image, it acts similarly to conventional filters. When a Gabor filter is used on an image, it provides the highest extraction at edges and points where texture changes. Gabor filters react to edges and texture changes in the image which means the filter has a distinguishing value at the spatial position of that feature.

III. METHODOLOGY

Dataset creation and the three models used namely, CNN model 1, CNN model 2, and GCNN model are explained in this section. The proposed method includes two basic steps namely, pre-processing (labeling images and resizing to 28 × 28) and CNN model (converting to grayscale, feature extraction, classification, and recognition) as shown in Figure 2. However, the performance of the CNN model is dependent on the parameter values of the architecture. Therefore, creating the model with optimum parameter values is required by fine-tuning and testing the model at various times. In this research, the trial and error method is followed to find out the optimal parameter set for CNN models.

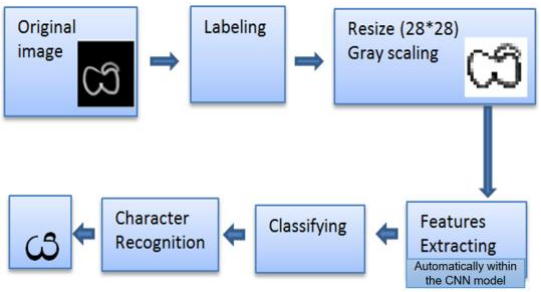


Figure 2. The proposed architecture.

D. Dataset

Since the unavailability of a proper Sinhala handwritten character image dataset, it was required to construct a dataset using appropriate mechanisms. The constructed image dataset contains all 60 Sinhala characters in the alphabet. The previous research regarding Sinhala character recognition using deep learning approaches was not carried out for all 60 Sinhala characters (Wasalthilake and Kartheeswaran, 2020), (Nilaweera et al., 2007). The constructed dataset contained a total of 6,000 black and white images in .png format. These handwritten character images were gathered using the MS Paint application. To collect images of Sinhala characters, twenty-five numbers of different writers were selected with age groups between 6 to 60 years, including both gender types. One writer provided four samples of one Sinhala letter for all 60 characters creating a total of 6,000 samples of images. The collection of these Sinhala handwritten characters was used as the dataset needed to train the deep neural network and test.

E. CNN model 1

The architecture of CNN model 1 is as follows (Figure 3 top). A convolutional layer with 32 Rectified Linear Unit (ReLU) activated filters of size 5 × 5 is the first layer. This is the most significant layer in the proposed GCNN architecture as it modifies the first layer initialization to the Gabor filter from the default Xavier (Glorot) initialization. The second and third layers are convolutional layers with 32 and 64 filters consecutively of size 3 × 3 and ReLU activation. The data then pass to a 2×2 max-pooling layer with a stride of 2, a ReLU-activated third convolutional layer of 128 number of 3×3 filters, and a 2×2 max-pooling layer with a stride of 2. Then it goes through a fourth convolution layer which has ReLU activated 3×3 256 filters. Next, the data is passed to a 2×2 max-pooling layer with a stride of 2. The output is then fed into two fully connected layers, where the final fully connected layer uses a softmax activation for classification. The dropout effect was applied during training before the softmax activation and after the first fully connected layer. Four dropout values $D \in \{0.0, 0.25, 0.5, 0.8\}$ have been used to obtain the optimal dropout value for the models. In this model, all parameters of the convolutional layers are Xavier initialized and all biases are zero-initialized. CNN model 1's optimization algorithm was the stochastic gradient descent (SGD). The selected hyperparameters for learning rate, learning rate decay, and momentum are 0.01, 0.0005, and 0.9 respectively. The categorical cross-entropy is used as the loss function

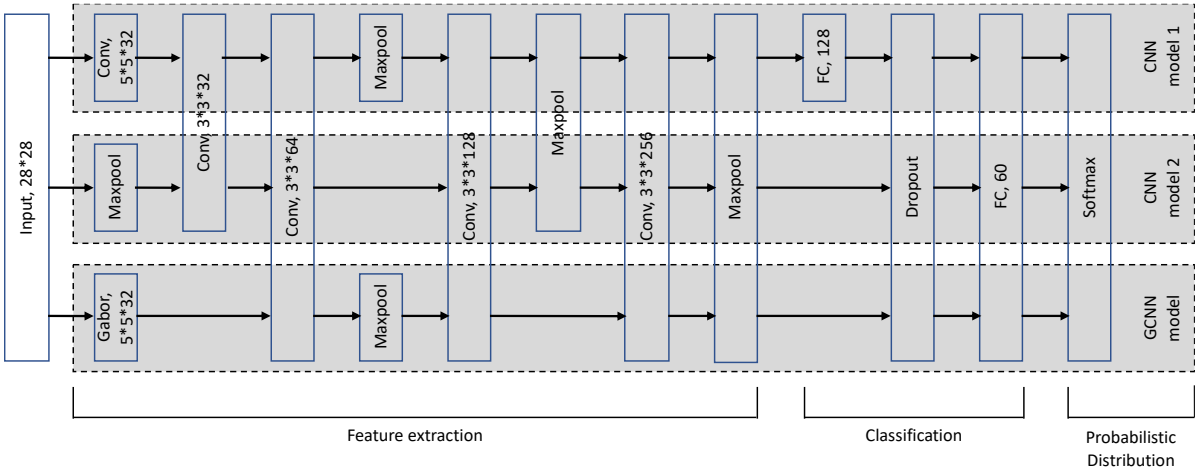


Figure 3. CNN architectures, top: CNN model 1, middle: CNN model 2, and bottom: GCNN model.

F. CNN model 2

As discussed in subsection **Error! Reference source not found.**, the CNN performance depends on the parameter values. Therefore, it is required to design the model with optimum parameter values by tuning and testing the model numerous times. Typically, the pooling layers are used after the convolution layers of the model. Having a pooling layer after the input layer increases accuracy by 4% (Wasalthilake and Kartheeswaran, 2020). Hence a max-pooling layer is inserted after the input layer in the CNN model 2.

The CNN model 2 architecture is as follows (Figure 3 middle). The first layer is a max-pooling layer of size 2 × 2 with a stride of 2 and it is applied directly after the input layer. The next layer is a convolutional layer that contains ReLU-activated 32, 3×3 filters. The third layer is a convolutional layer with ReLU-activated 64, 3×3 filters. Then data passes to a convolutional layer of ReLU activated 128, 3×3 filters, and a 2×2 max-pooling layer with a stride of 2. Then it goes through a convolution layer that has ReLU activated 256, 3×3 filters. The data was then passed to a 2×2 max-pooling layer with a stride of 2. The output is then fed into two fully connected layers, where the final fully connected layer uses a softmax activation for classification. The dropout effect was applied during training before the softmax activation and after the first fully connected layer. Four dropout values D = {0.0, 0.25, 0.5, 0.8} have been used to obtain the optimal dropout value for the model. In this model, all convolutional layers Xavier initiated. All biases are zero-initiated. CNN model 2 uses the SGD optimization. The selected hyperparameters for learning rate, learning rate decay, and momentum are 0.01, 0.0005, and 0.9 respectively. The loss function is formed on the Categorical cross-entropy.

G. GCNN model

The GCNN model is similar to the CNN model 1 in sub-section 1E, except the Xavier in the first layer is replaced by a Gabor filter bank (Figure 3 bottom). The first layer is a convolutional layer that contains Gabor-initiated, ReLU-activated 32, 5×5 filters. The second layer is a convolutional layer with ReLU-activated 64, 3×3 filters. Then data goes through a 3×3 max-pooling layer with a stride of 2. Next, the data passed to a third convolutional layer of ReLU-activated 128, 3×3 filters, and a fourth convolutional layer, which has ReLU-activated 256, 3×3 filters. Then output data passed through another 3×3 max-pooling layer with a stride of 2. After that, the output is fed into two fully connected layers. The final fully connected layer is implemented using a softmax activation for classification. The dropout function is used during training before the softmax activation with four different values D = {0.0, 0.25, 0.50, 0.80}. The most important aspect is that all convolutional layers beside the first layer are initialized with the Xavier for the parameters. All biases are zero-initialized and the RMSprop optimizer was used. The preferred hyperparameters for learning rate, momentum, and learning rate decay are 0.01, 0.9, and 0.0005 respectively. The Categorical cross-entropy was used as the loss function.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. CNN models

Training and testing sizes for all experiments were 3,600 and 2,400 images respectively. Both CNN architectures were tested for 30 epochs and were trained using a learning rate of 0.001 and implemented with an SGD optimizer. All CNN architectures are experimented with four different

dropout values to determine how the dropout effect improves the performance of the CNN models.

The results of CNN models 1 and 2 are listed in Table 1 and plotted in Figure 4. Both CNN models were trained and tested with different dropout values $D \in \{0.0, 0.25, 0.5, \text{and } 0.8\}$ for 50 epochs. The highest training and testing accuracies were 0.9994 when $D = 0.0$ and 0.9014 when $D = 0.5$ for model 1; and 0.9997 when $D = 0.0$ and 0.7942 when $D = 0.5$ for model 2.

Table 1. Results of CNN models 1 and 2.

Dropout	CNN model 1				CNN model 2			
	Training		Testing		Training		Testing	
	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
0.00	**0.9994**	0.0025	0.8407	0.7662	**0.9997**	0.0013	0.7693	1.3529
0.25	0.9888	0.0327	0.8502	0.8360	0.9891	0.0334	0.7706	1.2502
0.50	0.9633	0.0976	**0.9014**	0.4514	0.9417	0.1543	**0.7942**	1.0058
0.80	0.6203	1.1180	0.8485	0.4735	0.3430	2.1192	0.5807	1.4657

As per Figure 4, for both CNN models, training accuracy reduces with dropout and rapidly decreases when $D > 0.5$. But testing accuracy slightly increased till $D = 0.5$ and then reduced. Training and testing loss of both CNN models increased with dropout and it rapidly increases for higher dropout values. In addition, the training and testing accuracy of CNN model 1 is greater than that of model 2. So, introducing a max-pooling layer between the data input layer and the convolution layer does not improve the training and testing accuracy for the constructed Sinhala character dataset.

Even though research conducted by Wasalthilake (Wasalthilake and Kartheeswaran, 2020) claimed that the accuracy could be improved by feeding input images directly to the max-pooling layer before the convolution layer, this research could not achieve such improvement in the performance of CNN for the constructed character dataset. The CNN model 1 achieved 0.9633 training accuracy and 0.9014 testing accuracy with $D = 0.5$ for 60 classes of characters.

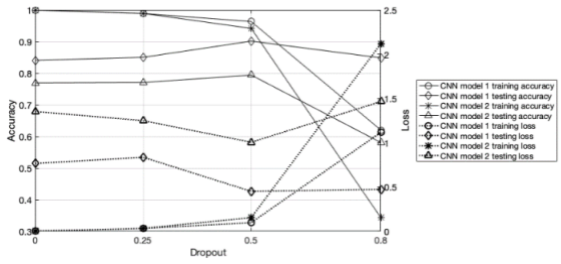


Figure 4. Accuracy and loss vs dropout for CNN models 1 and 2.

B. GCNN model

This experiment was carried out to investigate the behavior of each parameter used in the Gabor filter and to determine the significance of each parameter over the GCNN model concerning the Sinhala character dataset. In addition, the effect of dropout is also investigated under four test cases. Gabor filters of the first layer are initiated using the grid search algorithm.

The main objective of the GCNN was to achieve reasonable energy saving in computation time by decreasing the complexity of the CNN network by utilizing error resilience. In the GCNN, trained layers of CNN are replaced by fixed Gabor kernels and hence CNN can eliminate these expensive computation methods within the layers with the same or higher accuracy levels.

i. *Behavior of Gabor parameters*

The Gabor filter bank was applied to initialize the first layer to determine the influence of each parameter on the overall GCNN model. The Gabor filters used are the same for all tests where each parameter is tested and changed once per test. An example parameter list is $k = 5\times5$, $\sigma = \sigma'$, $\lambda = 50$, $\theta = 0$, $\gamma = 150$, $\psi = 0$, where k is the filter size, σ' is a static constant number in the closed interval [2, 22] for σ parameter. All filters in the Gabor filter bank are the same. Each test was trained for 30 epochs and the filter size was 5 × 5 for all test cases in this experiment at 32 filters in the initial layer.

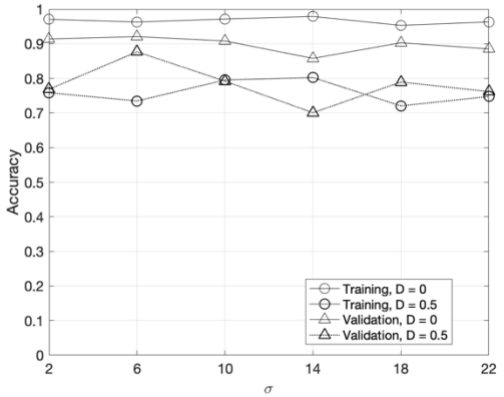
As described above, the Gabor filter bank is kept static for each test, and the training and generated Gabor filters allow learning through training. The parameter values selected for the test cases were based on previous research (Özbulak and Ekenel, 2018). Tunable Gabor filter parameters are explained below. Furthermore, the experiment was also designed to determine the impact of the dropout effect on the quality of results. Here, two different dropout values of $D \in \{0, 0.50\}$ were applied. Each test contains 30 epochs. The number of Gabor filters generated is constant at 32 filters. Each test is conducted once per parameter change for all parameters. Consider one test run, $\sigma = 2$, while other variables ($k, \sigma, \lambda, \theta, \gamma, \psi$) in the Gabor filter are kept constant and in the next test case, σ is a fixed increment to $\sigma = 4$ and the other variables ($k, \sigma, \lambda, \theta, \gamma, \psi$) are fixed. Each Gabor parameter contained six test cases with different ranges and increments as illustrated in Table 2. Therefore, for one given Gabor parameter, a total of twelve tests have been conducted considering the two dropout values. θ and ψ values of 0 and 360 provided the same result, so one value is considered.

Table 2. Gabor parameter test cases

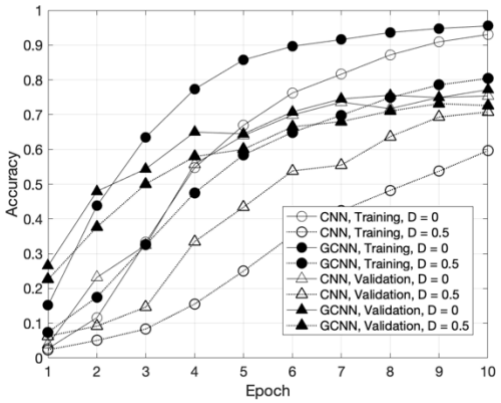
Parameter	Dropout	k	σ	λ	θ	γ	ψ
σ	{0, 0.5}	5 × 5	[2, 22] 2, 6, 10, 14, 18, 22	10	0	150	0
λ			4	[0, 100] 0, 20, 40, 60, 80, 100			
θ				10	[0, 300]		

					0, 60, 120, 180, 240, 300		
γ					0	[0, 300] 0, 60, 120, 180, 240, 300	
ψ						150	[0, 300] 0, 60, 120, 180, 240, 300

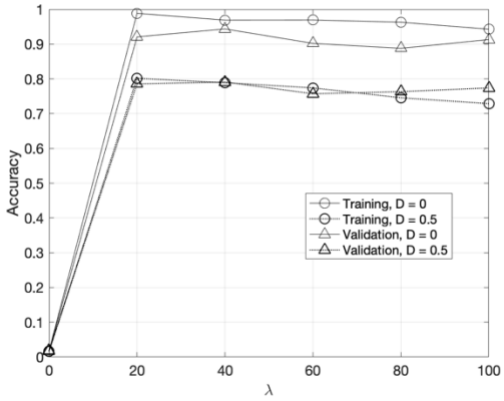
In the σ test, a static first layer was implemented using the Gabor filter bank by changing the $\sigma \in \{2, 6, 10, 14, 18, 22\}$ and $D \in \{0.0, 0.5\}$. Training and validation accuracy achieved are illustrated in **Error! Reference source not found.**(a1). In the λ test, a static first layer was developed using the Gabor filter bank by changing the $\lambda \in \{0, 20, 40, 60, 80, 100\}$ and $D \in \{0.0, 0.5\}$. Training and validation accuracy achieved are illustrated in **Error! Reference source not found.**(b1). For the θ test, a static first layer was implemented using the Gabor filter bank by only changing the $\theta \in \{0, 60, 120, 180, 240, 300\}$ and $D \in \{0.0, 0.5\}$. θ calculating at 0 and 360 provided the same result, so the 360 test case was neglected. Training and validation accuracies achieved are illustrated in **Error! Reference source not found.**(c1). For the γ test, a static first layer was implemented using the Gabor filter bank by changing the $\gamma \in \{0, 60, 120, 180, 240, 300\}$ and $D \in \{0.0, 0.5\}$. Training and validation accuracies achieved are illustrated in **Error! Reference source not found.**(d1). In the ψ test, a static first layer was implemented using the Gabor filter bank by changing the $\psi \in \{0, 60, 120, 180, 240, 300\}$ and $D \in \{0.0, 0.5\}$. ψ calculating at 0 and 360 provided the same result, so the 360 test case was neglected. Training and validation accuracy are illustrated in **Error! Reference source not found.**(e1).



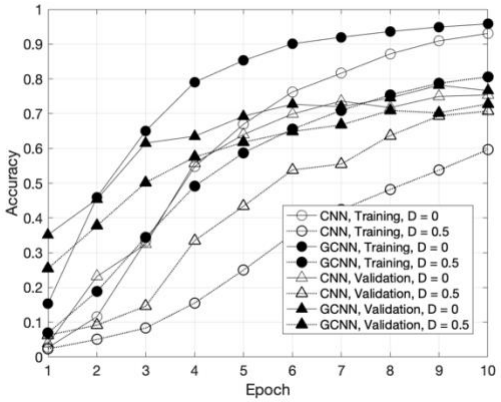
(a1)



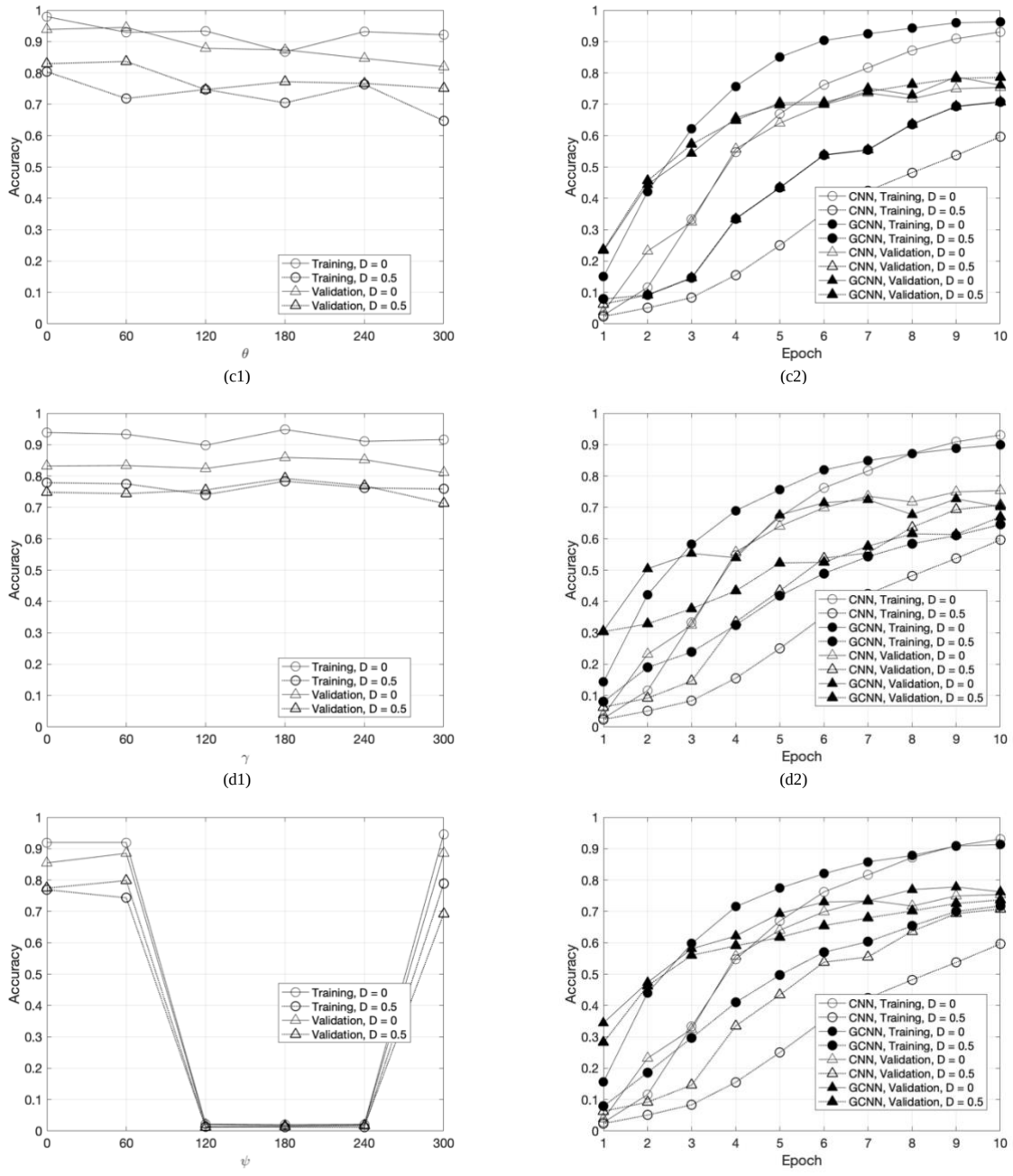
(a2)



(b1)



(b2)



(c1) (c2)

(d1) (d2)

(e1) (e2)

Figure 5. Training and validation accuracy, left: changing Gabor parameters in GCNN, right: changing epoch of CNN model 1 and GCNN (a) σ, (b) λ, (c) θ, (d) γ, (e) ψ.

Table 3: Highest training and validation accuracy of the GCNN with static first Gabor layer.

Dropout	σ	λ	θ	γ	ψ	Training		Validation	
						Accuracy	Loss	Accuracy	Loss
0	14	50	0	150	0	0.9790	0.1340	0.8024	2.1065
0.5	6	50	0	150	0	0.9208	0.2567	0.8768	0.7960
0	10	20	0	150	0	0.9885	0.0657	0.8015	2.3630
0.5	10	40	0	150	0	0.9437	0.1805	0.7908	0.9356
0	10	50	0	150	0	0.9793	0.0753	0.8037	1.6760
0.5	10	50	60	150	0	0.9452	0.1763	0.8364	0.7158
0	10	50	0	0	0	0.9389	0.6047	0.7787	4.9222
0.5	10	50	0	180	0	0.8591	0.4497	0.7925	0.8572
0	10	50	0	150	300	0.9466	0.3743	0.7886	2.3969
0.5	10	50	0	150	300	0.8858	0.3621	0.6919	0.8192

ii. Effect of epoch

Training accuracy decreased when dropout was introduced to the GCNN. It can be observed that the error increases with dropout. There appears to be no significant pattern in terms of σ. The highest training and validation accuracies were achieved when σ = 14 and σ = 6 for dropouts 0.0 and 0.5, respectively. Considering Figure 5(a2), it can be concluded that for both dropout values 0.0 and 0.5, the training and validation accuracies of the GCNN are higher than CNN accuracy for each epoch. So, the convergence of the GCNN is faster than the corresponding CNN model. In addition, according to Figure 5(a2), σ has no significant impact on the final results.

According to Figure 5(b2), It can be observed that the accuracy levels are very close to each other and irregular like the σ. For both dropout values of 0.0 and 0.5, λ appears to be good in the range [20, 100]. When dropout is introduced to the GCNN architecture, there is no significant change in performance. The important conclusion of this test is that λ = 0 should never be used for the Gabor filter parameter setting because the network has trouble learning when there is no value given to lambda. The highest training and validation accuracy levels were obtained when λ = 20 and λ = 40 for dropout values 0.0 and 0.5 respectively. Considering Figure 5(b2), it can conclude that for dropout value 0.0, the training and validation accuracy of the GCNN with λ = 20 become almost similar to CNN accuracy for higher epochs. For dropout value 0.5, the validation accuracy of both CNN and GCNN with λ = 40 becomes very close when the number of epochs increases. In contrast, the training accuracy of GCNN is higher than the CNN for a higher number of epochs.

When θ = 0 and θ = 60, the highest accuracy level can be obtained for the dropout at 0.0 and 0.5, respectively. It is shown that there is no significant impact on the training and validation accuracy when introducing the dropout to the GCNN. According to Figure 5(c2), training accuracy may decrease with the higher value of θ for both dropout values. In addition, there is no significant pattern to recognize from the accuracy level and they are very close to one another. Considering Figure 5(c2), it can conclude that, for dropout 0.0 the training and validation accuracy of the CNN and GCNN with θ = 0 becomes almost similar for higher epochs. For dropout value 0.5, the validation and training accuracy of GCNN with θ = 60 is higher than the corresponding CNN accuracy. As discussed, different features at each frequency can be extracted by rotating the Gabor filters; hence Gabor filter is considered an alternative to transfer learning (Pham, 2019).

When γ = 0 and γ = 180, the highest accuracy level can be obtained for the dropout at 0.0 and 0.5 respectively. Training accuracy was decreased when introducing the dropout effect. It can be observed that there is no significant pattern visible when increasing the γ value. Considering Figure 5(d2), it can conclude that, for dropout 0.0, the training and validation accuracy of the GCNN with γ = 0 becomes almost similar to CNN accuracy for the higher epoch. For dropout value 0.5, the training and validation accuracy of GCNN with γ = 180 is higher than the corresponding CNN accuracy.

According to Figure 5(e2), it is clear that there is a range for the Gabor variable ψ. ψ values available within the range [120, 240] should never be used. The GCNN network has difficulty in training when ψ variable values are within this interval. The accuracy levels do not significantly change when adding dropout to the network. When ψ = 300 highest accuracy level can be obtained for both dropouts 0.0 and 0.5. The highest training and validation accuracy of the GCNN with static first Gabor layer are provided in Table 3.

Gabor filters are based on the mathematical equation with the parameters σ, λ, θ, γ, and ψ. When the value of λ is 0, the GCNN network fails to converge and does not train as expected. As the Gabor filter is subject to its rotational behavior to extract features, θ should have the most importance. However, in the above experiment using a single parameter value for θ does not show a significant impact. Subject to the constructed Sinhala character dataset, it can be concluded that λ should not be equal to 0. The results show that when the value of parameter ψ is in [120, 240], the network is unable to provide any significant results. The parameter ψ should not be in [120−240] to obtain the optimal accuracy level. Table 3 presents the highest values achieved for each experiment and Table 4 summarizes the results obtained from the experiment.

Table 4. Summary

Parameter	σ	λ	θ	γ	ψ
Impact	Low	Low	High	Low	High
Best value range	[2-14]	[20-100]	0	180	[0-120]
Worst value range	-	0	-	-	[120-240]

iii. Testing on the GCNN model

The parameter values of the Gabor filter bank are obtained by considering the results obtained from the above experiment and the trial-and-error method to achieve higher accuracy. For the test, 30 epochs are considered. Filter size will remain at 5×5 at 32 filters in the initial layer. Grid search was applied to determine the Gabor filter bank. The test was carried out for four different dropout values $D \in \{0.0, 0.25, 0.5, 0.8\}$.

The training and validation accuracy of the GCNN model are listed in Table 5. Here 30 epochs have been used for four different dropout values $D \in \{0.0, 0.25,$

0.5, 0.8}. Finally, the comparison of two different CNN architectures and GCNN architecture is listed in Table 6 and plotted in Figure 6 for each epoch.

Table 5. Results of the GCNN model

Dropout	Training		Testing	
	Accuracy	Loss	Accuracy	Loss
0.00	0.9575	0.2896	0.7503	3.4728
0.25	0.9515	0.2652	**0.8067**	2.1316
0.50	0.9030	0.3127	0.7908	1.0172
0.80	0.6455	1.1720	0.6659	1.1311

According to Table 5, the highest training and validation accuracy was achieved at 0.25 dropout value. Compared with the CNN model 1 results, the training accuracy of the GCNN is reduced by approximately 3%, and the validation accuracy is reduced by approximately 5%. But considering Table 6, it can be concluded that the convergence speed of the GCNN is higher than the CNN models 1 and 2.

Table 6. Comparison of two different CNN networks and GCNN

Epoch	Accuracy					
	Training			Validation		
	CNN model 1 (D = 0.5)	CNN model 2 (D = 0.5)	GCNN model (D = 0.25)	CNN model 1 (D = 0.5)	CNN model 2 (D = 0.5)	GCNN model (D = 0.25)
1	0.0235	0.0232	0.1392	0.0628	0.0568	0.3319
2	0.0505	0.0525	0.3892	0.0917	0.1507	0.5045
3	0.0835	0.1177	0.5258	0.1464	0.2514	0.5971
4	0.1553	0.2009	0.6438	0.3349	0.3577	0.6263
5	0.2500	0.2821	0.7448	0.4339	0.4580	0.6823
6	0.3533	0.3427	0.8022	0.5381	0.5118	0.7099
7	0.4237	0.4237	0.8473	0.5549	0.5428	0.7314
8	0.4819	0.4753	0.8711	0.6367	0.5669	0.7508
9	0.5373	0.5198	0.8941	0.6931	0.6259	0.7486
10	0.5962	0.5729	0.9087	0.7077	0.6457	0.7908

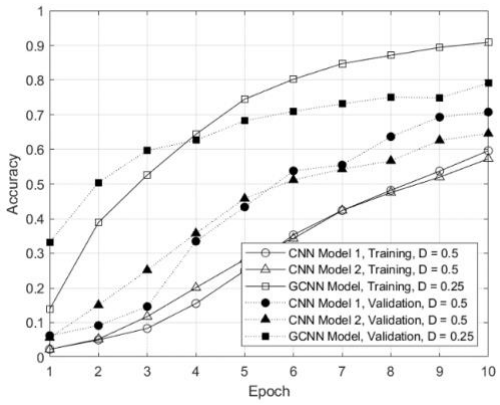


Figure 6. Comparison of two different CNN networks and GCNN.

When comparing CNN model 1 with a dropout of 0.5 and GCNN with a dropout of 0.25 their eventual accuracy levels are close to each other. However GCNN structures converge to the accuracy level faster than the corresponding CNN models. So, Gabor filters can be applied in the initial layer of the CNN model instead of convolutional learning in the first layer, even though GCNN reduces the accuracy level by approximately 5% under the same hyperparameters. In addition, by the trial-and-error method, it is found that the constructed Sinhala character image dataset provides a better accuracy level with a smaller number of filters in the Gabor filter bank. As a part of this research, the impact of the Gabor variables is investigated. The five parameters, σ, λ, θ, γ, and ψ in the Gabor filter have a remarkable impact on training performance as some values cause decreased performance, increased performance, or are unable to converge. It was found that variables θ and ψ have a higher impact on the training. It was found that the λ should not be equal to zero. When λ is greater than zero, it behaves similarly to the variables σ and γ in the training phase.

V. CONCLUSION

This research proposed two CNN architectures and one Gabor-initialized CNN (GCNN) architecture for the Sinhala character recognition. Since the unavailability of an appropriate Sinhala character image dataset for the research, a dataset of 6,000 character images was constructed for all Sinhala characters in the alphabet. The CNN model 1 consisted of five convolutional layers with a max-pooling layer between some convolutional layers. In CNN model 2, the input layer is directly fed into the max-pooling layer and then fed into four convolutional layers. Both CNN architectures introduced a dropout layer with 0.0, 0.25, 0.5, and 0.8 values before the fully connected layer to obtain how the dropout effect affects CNN's performance. We further experimented with the importance of the Gabor parameters. Accordingly, Gabor parameter λ should not be equal to 0, and the value of parameter ψ should not be in the range between [120−240] to obtain the optimal accuracy level. In the GCNN architecture, trained layers of CNN are replaced by fixed Gabor kernels to reduce expensive computation within the layers. We applied Gabor filter initialization to the first layer of the CNN architecture. It is observed that the GCNN trains faster than the corresponding CNN architecture in the initial epochs. It is concluded that the GCNN is capable of training faster in the early phases of the training process. This can prove that GCNN can enhance the convergence of the network and reduce the training time with the same or slightly less accuracy. Considering the strengths and limitations of proposed CNN architectures and GCNN, the more significant architecture was selected to recognize the Sinhala characters. This research has applied the Gabor-initialization method only for the initial layer of the CNN. It can apply to all the convolutional layers. It is important to experiment with how the performance of the GCNN varies when introducing the Gabor initialization method for other convolutional layers. All experiments conducted

above use the grid search for assigning values for the parameters of the Gabor filter. It is important to examine the behavior of the GCNN network with other Gabor initialization methods, such as random initialization. Experiments can be conducted to find the performance of the GCNN architecture when varying the filter size.

VI. REFERENCES

Albawi, S., Mohammed, T.A., Al-Zawi, S., 2017. Understanding of a convolutional neural network, in: 2017 International Conference on Engineering and Technology (ICET). Ieee, pp. 1–6.

Alekseev, A., Bobe, A., 2019. GaborNet: Gabor filters with learnable parameters in deep convolutional neural network, in: 2019 International Conference on Engineering and Telecommunication (EnT). IEEE, pp. 1–4.

Alom, M.Z., Sidike, P., Hasan, M., Taha, T.M., Asari, V.K., 2018. Handwritten bangla character recognition using the state-of-the-art deep convolutional neural networks. Computational intelligence and neuroscience 2018.

Anello, E., 2021. A Comprehensive Guide of Regularization Techniques in Deep Learning [WWW Document]. Towards Data Science. URL https://towardsdatascience.com/a-comprehensive-guide-of-regularization-techniques-in-deep-learning-c671bb1b2c67 (accessed 12.15.22).

Gabor, D., 1946. Theory of communication. Part 1: The analysis of information. Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering 93, 429–441. https://doi.org/10.1049/ji-3-2.1946.0074

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., 2018. Recent advances in convolutional neural networks. Pattern recognition 77, 354–377.

Hewavitharana, S., Fernando, H.C., Kodikara, N.D., 2002. Off-Line Sinhala Handwriting Recognition Using Hidden Markov Models., in: ICVGIP. Presented at the Indian Conference on Computer Vision, Graphics and Image Processing, Ahmedbad, India.

Jain, S. jain, 2018. Regularization Techniques | Regularization In Deep Learning. Analytics Vidhya. URL https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/ (accessed 12.15.22).

Nilaweera, N.P.T.I., Premeratne, H.L., Sonnadara, D.U.J., 2007. Comparison of projection and wavelet based techniques in recognition of Sinhala handwritten scripts, in: Proceedings of the 25th National IT Conference. Colombo, Sri Lanka, pp. 93–97.

O'Shea, K., Nash, R., 2015. An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458.

Özbulak, G., Ekenel, H.K., 2018. Initialization of convolutional neural networks by Gabor filters, in: 2018 26th Signal Processing and Communications Applications Conference (SIU). Presented at the 2018 26th Signal Processing and Communications Applications Conference (SIU), pp. 1–4. https://doi.org/10.1109/SIU.2018.8404757

Petkov, N., Wieling, M.B., 2008. Gabor filter for image processing and computer vision.

Pham, L., 2019. Gabor filter initialization and parameterization strategies in convolutional neural networks.

Rajapakse, R.K., Weerasinghe, A.R., Seneviratne, E.K., 1995. A neural network based character recognition system for Sinhala script. Presented at the the South East Asian Regional Computer Confederation, Conference and Cyberexhibition (SEARCC'96), Citeseer, Bangkok, Thailand.

Silva, C.M., Jayasundere, N.D., Kariyawasam, C., 2015. Contour tracing for isolated Sinhala handwritten character recognition, in: 2015 Fifteenth International Conference on Advances in ICT for Emerging Regions (ICTer). IEEE, pp. 25–31.

Wang, Y., Yu, P., 2019. Offline handwritten new Tai Lue characters recognition using CNN-SVM, in: 2019 IEEE 2nd International Conference on Electronic Information and Communication Technology (ICEICT). IEEE, pp. 636–639.

Wasalthilake, W., Kartheeswaran, T., 2020. Sinhala handwritten character recognition using convolution neural networks, in: Proceedings of the FARS2020. Presented at the First Annual Research Session (FARS 2020), Faculty of Applied Science, Vavuniya campus of the University of Jaffna, Vavuniya, Sri Lanka, pp. 41–45.

Wu, G., 2021. Study on Face Recognition Based on Improved CCN and Ensemble Learning, in: 2021 33rd Chinese Control and Decision Conference (CCDC). IEEE, pp. 3414–3417.

Wu, J., 2017. Introduction to convolutional neural networks. National Key Lab for Novel Software Technology. Nanjing University. China 5, 495.

Yuan, Y., Wang, L.-N., Zhong, G., Gao, W., Jiao, W., Dong, J., Shen, B., Xia, D., Xiang, W., 2022. Adaptive Gabor convolutional networks. Pattern Recognition 124, 108495.