Proceedings of 10th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2021 Month: October Volume: 10

Spiking Neural Network based Handwritten Devanagari Character Recognition

Bikram Acharya [a], Diwakar Raj Pant [b], Manoj Joshi [c]

[a, b, c] *Institute of Engineering, Pulchowk Campus*
Corresponding Email: [a] 075mscsk004.bikram@pcampus.edu.np, [b] drpant@ioe.edu.np

Abstract
Spiking neural networks (SNNs) involves communication and processing of fragmented and asynchronous binary signals. It differs from traditional approach neural networks as it operates on spikes, which are discrete events that occur at particular times rather than constant values. Application of spiking neural network are being widely explored at recent times and while deep Neural Networks and Classical Machine Learning techniques were previously used to address handwritten recognition problem, Spiking Neural Networks(SNN) have become more promising with recent development of neuromorphic device. This research work implements SNN model for hand-written Devanagari digits classification. The proposed method weights transfer from ANN network to similar SNN network along with threshold calculation for SNN layers. Learning in neurons is achieved by using spike timing dependent plasticity(STDP), an unsupervised learning process, to strengthen synapses that results to the generation of an output spike, while those that do not contribute are weakened. This work uses a very popular Devanagari Handwritten Character Dataset and over the course of number of experiments and using evaluation metrices like Accuracy, Loss, confusion matrix and f1 score, it was found that Hybrid SNN(VGG-16) performed better on overall metrics and had accuracy of 96.7% on test sets and 98.5% on train sets, highest among all tested architectures.

Keywords
Spiking Neural Network,Spike Timing Dependent Plasticity, Hybrid Learning, Thresholding, Synapse

1. Introduction

A brain functions as a framework of neurons which communicate which each other employing a complex web of synaptic associations. The smallest entity in the brain are nerve cells or neurons which can be called as processing elements. A neuron is cell which is electrically excitable and uses electrical and chemical signals to process and transmit information [1]. These elementary biological neurons forms a complex interconnect biological neural networks, which is considered the morphological and functional unit of the nervous system, which occur via, synapses. Human brain consists of approximately hundred billion neurons with synapse average of 10^{14} 10^{16} in an adult's brain [2]. Despite being such a huge and complex system, neurons in the brain manage to send signals rapidly and precisely to other cells through these connections.

Biological information processing in brain centers in neural network primarily on processing performance of high magnitude yet with very low power consumption like that of biological brain. In recent times, much of research works has centered on imitating the brain functionality, building complex neural systems which can be adjusted to be prepared and allot meaning to complex information designs. Artificial Neural Systems (ANN) with wide range of applications, contain straightforward implementation of natural neurons and are restricted in their computational capacity. Deep learning, emerged as successor and the go-to tools for many machine learning tasks, has been widely used and proven to outperform other machine learning algorithms in cognitive tasks[3]. However, these algorithms are greedy computationally and to yield valuable information by crunching inputs, and requires large computing clusters and capable GPUs; consuming high energy. This acts as deterrent for long-term active system, or edge devices and has encouraged researchers to focus in optimizing neural networks implementation. Subsequently, in scenarios where low power consumption is prioritized, on-site use of ANNs may not be a reasonable choice.

In recent years, SNN has guaranteed empowerment of low-power assignments and it makes use of event-driven neuromorphic hardware. The neurons in an SNN computes, transfers and carries data through distinct parallel occasions (or spikes), whereas the standard ANNs handles information in real-values way. The binary spike-based communication in combination with sparse temporal preparing establish SNNs as low power alternative to traditional ANNs. Despite higher control effectiveness and power efficiency, preparing SNNs remains a challenge till date [4].

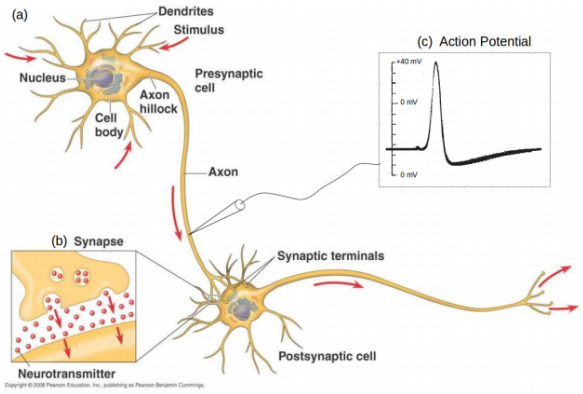

Figure 1: (a) A neuron constituting : dendrites, the cell body, and the axon, and 2 neurons connected by synapses (b) Synapse connecting A pre-synaptic cell and its post-synaptic cell ,and (c) the neural signal, the action potential , propagation shown by the red arrows.

Spiking Neural Networks (SNN) are a set of neurons that communicates through spikes and compute through the timing of spikes. Spiking Neural Network are what we call the third generation networks [5] after McCulloch-Pitts and feedback networks in first and second generation respectively. The spiking neurons have become popular since they mimic the spiking nature of biological neurons and can reproduce those neuron spiking patterns. Besides synaptic and neuronal state, SNN has included an idea of time in the workflow. Unlike traditional multi-layer ANNs, in SNNs neurons are active only then the membrane potential, a difference in electric potential between neurons, is exceeded predetermined value. There are cases where SNNs are more biologically plausible and more powerful than non-spiking ones. Previously, few approaches like multilayer perceptron and radial basis function classifier [6, 7] had been used for Devanagari characters classification and some Deep Convolution Neural Networks[8] that used dropout and dataset increment approach have shown impressive results over conventional shallow networks, but no algorithmic implementation and realization is aimed to achieve such comparable results using SNN networks. The implementation of the Handwriting Recognition System is an evolving need to digitize handwritten Nepali documents . Also there is less discovery and research of optical character identification for Devanagari characters.

2. Related Literature

In many computer vision issues, character detection in cases as handwriting recognition is an important aspect. In writing units, individual writing style brings unevenness, which is the most difficult aspect to define. Writing units denote number, form, scale, stroke order, and writing speed with the number of strokes differing. The important variables to be included in classification for Devanagari Characters are their shapes and proportions, orders, tilting angles and similarities between characters from each other.I. K. Sethi et al. [9]'s work published in 1976 seems to be first research work published. By using an organized approach, which in turn discovered the presence and locations of vertical and horizontal line segments, C-curve, D-curve, right slant, left slant, the proposed system recognized the handwritten Devanagari numerals. Many researchers [10, 11] used various techniques to establish an HDCR (Handwritten Devanagari character recognition) method. There are several methods to identify such characters, such as Support Vector Machine, Hidden Markov model, Fuzzy based classification used in this work [12]. However, Pre-processing and features extraction from the data must be performed in these conventional methods and then a classifier is trained using those derived features.

Similarly, Deep neural nets have been used for object identification, classification and segmentation in the field of image processing since the early 2000s, with immense results. The first deep learning method for character recognition was proposed in 1998 for the MNIST database [13]. Every hidden layer in deep learning consisting multiple neuron measure and maintain the necessary weights for the network but inverts handcrafting and designing features into an automated mechanism to calculate the optimum features for the problem with a specific question.

In most shallow learning models, the features are

extracted only once, but in the case of deep learning models, multiple convolution layers have been introduced multiple times to extract discriminating features. This is one of the reasons that there is generally traction with deep learning models. The pretrained convolutiona neural net is also suitable for the handwritten Devanagari character recognition as proposed in paper by [14].This strategy is called learning to move.

Most recent studies have concentrated on standard CSNN and the dataset of the MNIST. Wang et al. [15] suggested a way of looking for similarity using successively connected encoding in a forward SNN. A new technique for training multi-layer spiking convolution neural networks (CSNN) combining supervised and unsupervised learning was suggested by the authors. Two components for unsupervised feature extraction and supervised classification include the training phase using improvised version of Spike-Timing Dependent Plasticity (STDP). Kheradpisheh et al., constructed an STDP-based deep spiking CSNN consisting of three convolutional layers and three pooling layers of one Difference of Gaussians (DoG) layer (temporary encoding), using unsupervised STDP learning. The paper suggests the SNN eight-layer architecture and is evaluated in the dataset of MNIST. Kulkarni et al.[16] suggested a three-layer architecture trained on the MNIST dataset with a supervised learning technique using the spike induced Normalised Approximate Descendant. A deep spiking CSNN (SpiCNN) consisting of a hierarchy of stacked convolution layers, a spatial pooling layer and a totally linked layer is suggested by Gerstner et al. [17] and checked using the MNIST dataset. For image processing, relevant literature on CSNNs [16, 18, 15] all in common have multi-layer CSNNs, use of the Leaky-integrate-and-fire (LIF) neuron model, use of unsupervised/supervised STDP (or STDP adaptation) for training, and use of black-and-white handwritten MNIST dataset.

Krizhevsky et. Al.[19] describes how a Deep Belief Network(DBN) can be trained on CIFAR-10 and makes use of NatCSNNN like 2-layer architecture in the paper. Similarly, in the paper by R. K. Srivastava et al. [20], a 32-layer network is proposed, described as Long Short-Term Memory (LSTM)-inspired highway networks .Futher, an All-CNN consisting of 10 layer network without data augmentation for the CIFAR-10 dataset was introduced by Springenberg et al.[21].

No similar research work is available for DHCD dataset.

3. Methodology

The block diagram 2 below shows the overall methodology of the project work.

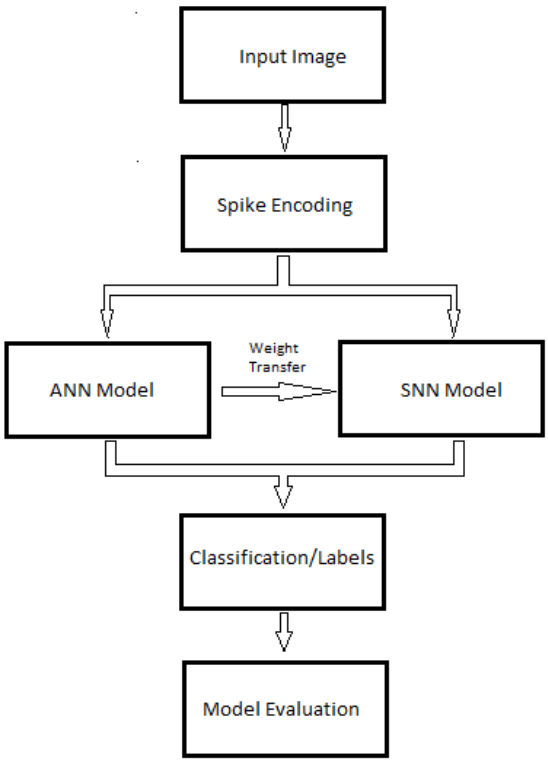

Figure 2: methodology

3.1 DHCD Datasets

For training and testing both ANN and SNN networks, publicly accessible dataset [8], Devnagari Handwritten Character Dataset (DHCD). The following table gives the composition of the dataset. The characters are white images on a dark backdrop.

Table 1: Dataset Composition

Parameter	Statistics
Total Images	92000
Total Devnagari Characters	46
Per Character Images	2000
Consonant Character Images	72,000
Numeric Character Images	20000
Actual Image Size	28*28
Padded Image Size	32*32

To increase the image size, padding of 0 was placed at 2 pixels on all four edges,
Figure 3 shows the snapshot of characters in DHCD dataset.

Figure 3: DHCD Dataset

3.2 Spike Encoding

The input and output to a SNN are in the form of spike trains while the data used is image data which is analog in nature and needs encoding into spike trains. All data values are mapped with a threshold function, and the encoded bits pipe can also be represented by a spike function. This results in a significant loss of data but can be minimized by selecting threshold to be normally distributed. Spike generation can be done by using a Poisson's Generator for same purpose.

3.3 Model Implementation

3.3.1 VGG Architecture

VGGNet is a Convolutional Neural Network architecture proposed by Karen Simonyan and Andrew Zisserman from the University of Oxford in 2014. For this project, the input channels and image size for original VGG Network is change from 224*224 to 32*32 and input color channels to 1 since our dataset image has only one channel.

- **Input.** VGG takes in a 32*32 pixel black and white image.
- **Convolutional Layers.** VGG's convolutional layers have very small receptive field of size 3x3 and convolution filters also filters of same size that function as a linear transformation of the input before a ReLU unit is applied. The convolution stride is set to 1 pixel in order to maintain spatial resolution after convolution.
- **Fully-Connected Layers.** VGG has three completely connected layers, the first two of which each have 4096 channels and the third of which has 46 channels, one for each class.
- **Hidden Layers.** All of VGG's hidden layers use ReLU.

Two VGG architectures, VGG5 and VGG16 are used for this project.

3.3.2 SNN with Hybrid Conversion

Learning in Spiking Neuron STDP is used as a method of learning for building spiking neural networks to imitate biological neural system which can perform complex computational operations.

Spike Timing Dependent Plasticity (STDP) Based on the current state, when a spiking neuron fluctuates the volume of its membrane potential while receiving data (spike, refractory), it emits an all-or-none response potential after reaching a threshold, relaying the information via the SNN. A synaptic portion that connects the neurons on which various learning functions could be implemented modulates this knowledge transfer.

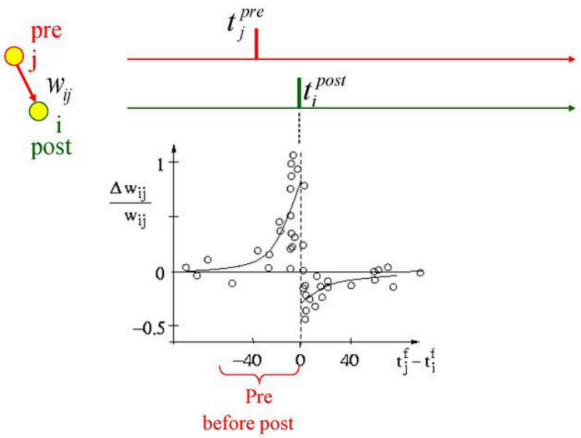

Figure 4: Synaptic changes vs Spike timing

Synaptic shifts are seen as a result of a pre and a post-synaptic neuron's spike timing differentiation as shown in Figure 4.

STDP is a weight updating rule. When the presynaptic spike (LTP) occurs before the postsynaptic spike, then the weight increases and reduces when presynaptic spike occurs after the postsynaptic spike. To do so, the connectivity between the neurons is enhanced or diminished, depending on the degree of coordination and synchronization of their spikes.

As STDP is an unsupervised learning process, the idea of which is to improve synapses which results in the

production of an output spike, while those that do not contribute are weakened.

For a pre and post synaptic neuron i.e i and j respectively, the modification of synaptic weight is characterized by the function as follows:

$$\blacksquare w_j = \sum_{k=1}^{N} \sum_{l=1}^{N} W(t_j^l - t_i^k) \tag{1}$$

In addition to the function, which determines the degree of synaptic weight change(increase/decrease depends on the timing of momentum between the pre and postsynaptic neuron as expressed as:

$$W(x) = \begin{cases} A_+ exp(-\frac{x}{\tau_+}) & if \quad x \geq 0 \\ A_- exp(\frac{x}{\tau_-}) & otherwise. \end{cases} \tag{2}$$

In the 1 and 2 equation, t_j^l represents the l^{th} activation time of the j neuron; similarly, t_i^k represents the k^{th} activation time of the i neuron; $A-$ and $A-$ are the constants defining the extent of the synaptic weight change (at $t = 0+$ and $t = 0\_$, respectively); and the constants for the exponential decrease in the shift in synaptic weight are represented by $\tau+$ and $\tau-$.

Hybrid Training and Weight Transfer This research work consists of implementation and modification of algorithm [4] for hybrid learning, and implements a threshold computation along with weight transfer in VGG, and other architecture for DHCD dataset. For all layers, excluding the input and output layer of the VGG architecture, the threshold balancing was carried out. The maximal input to the neuron for each hidden convolution/linear layer was calculated over all time steps and threshold was assigned for that particular layer sequentially with regards to both forward and pre-nonlinearity. Finally, the weights of every layer of ANN models (VGG) was transferred to similar SNN model.

Figure 5 presents the hybrid conversion and learning architecture.

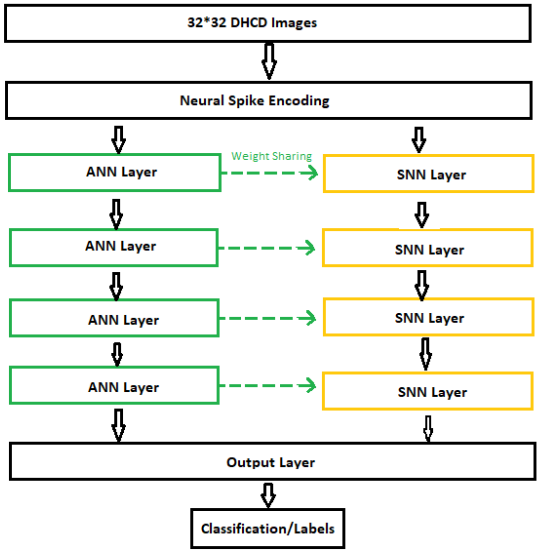

Figure 5: Architecture for SNN based on Hybrid Conversion

3.4 Model Evaluation

The performance of the the proposed models is dependent is done based on different parameter metrices obtained from confusion matrix as shown in the figure 6 below.

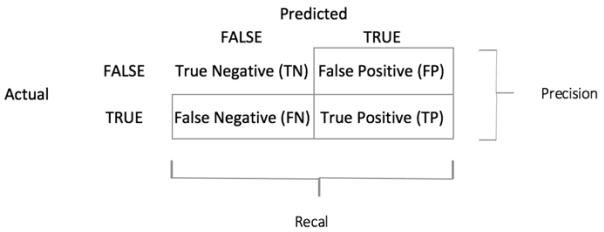

Figure 6: Confusion Matrix

Based on confusion matrix, different other performance parameters are calculated.

- **Recall**

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

- **Precision**

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4}$$

- **Accuracy**

$$\text{Accuracy} = \frac{\text{TP} + [\text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}} \tag{5}$$

- **F1 Score**

$$\text{F1 Score} = \frac{2 * \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \quad (6)$$

4. Result and Analysis

The data-set is divided into train and test sets at a ration of 78:22. Input image is converted to series of spikes using poisson generator as to emulate the spikes generated in brain when we visualize a digit. First ANN model was trained on the dataset. Weights from pre-trained ANN model was transferred to SNN by SNN-ANN conversion and SNN was trained with surrogate gradient computed with spike timing. The following parameters were used for optimum result after a number of experimentation:

Table 2: Parameter Used

SNN		ANN
Parameters	**Value**	**Value**
Batch Size	32	32
Learning Rate	0.001	0.001
Epochs	50	50
Optimizer	Adam	Adam
Droupout(Feature Layer)	0.3	0.3
Droupout(Classifier)	0.5	0.5
Kernel Size	3	3
Alpha	0.3	-
Beta	0.01	-
Leak	1.0	-

Table 3: SNN-ANN Parameters for VGG5 & VGG16

4.1 Learning Curve

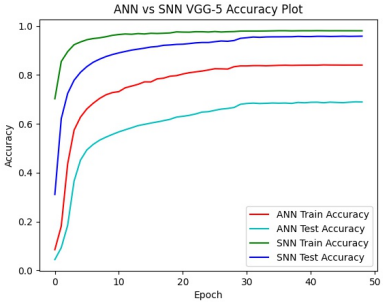

Figure 7: Learning Curve ANN vs SNN Hybrid VGG5

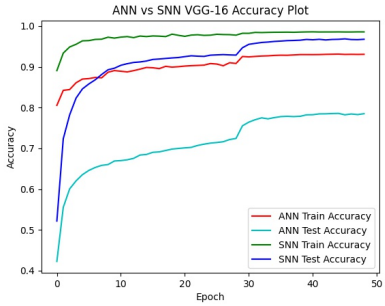

Figure 8: Learning Curve ANN vs SNN Hybrid VGG16

4.2 Loss Curve

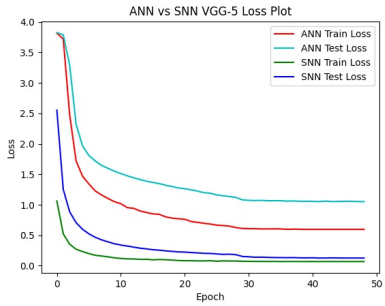

Figure 9: Loss Curve ANN vs SNN Hybrid VGG5

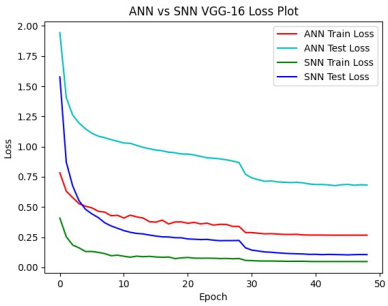

Figure 10: Loss Curve ANN vs SNN Hybrid VGG16

4.3 Precision, Recall and F1 Score

Metrics	ANN		SNN	
	VGG5	**VGG16**	**VGG5**	**VGG16**
Precision	**0.6591**	**0.7789**	**0.9589**	**0.9683**
Recall	**0.6620**	**0.7760**	**0.9582**	**0.9679**
F1 Score	**0.6561**	**0.7731**	**0.9582**	**0.9679**

Table 4: Metrics Comparison

Table 4 shows precision, recall and f1 score for different.

Architecture	ANN	SNN Hybrid
VGG5 (Epoch 50)	**68.98%**	**95.89%**
VGG16 (Epoch 50)	**78.5%**	**96.76%**

Table 5: Test Accuracy Comparison

Table 5 shows the test accuracy for different architectures at different epochs. VGG5 attained 68.98% accuracy at 50 epoch. However, SNN using hybrid training achieved that accuracy at 8 epoch and 95.89% accuracy at 50 epoch.

Similarly, For VGG16, highest accuracy was 78.54% and SNN Hybrid achieved that accuracy at 3rd epoch and further to final accuracy of 96.76% at epoch 50.

With hybrid training method, models are able to achieve similar performance at lower epoch. These results suggests there is plenty room for further research on threshold optimization for optimization of hybrid model.

5. Conclusion

This research works implements various SNN Hybrid models and evaluating all the matrics, SNN Hybrid conversion models appears promising. For same accuracy level, SNN managed to achieve it in fewer epoch and loss too has decreased significantly i.e ANN VGG5 attained 68.98% accuracy at 50 epoch and SNN using hybrid training achieved that accuracy at 8 epoch and 95.89% accuracy at 50 epoch, also for ANN VGG16, highest accuracy was 78.54% and SNN Hybrid achieved that accuracy at 3rd epoch and further to final accuracy of 96.76% at epoch 50. Hybrid model converges faster with higher accuracy and performs better on complex model i.e VGG16 than VGG5. Pure SNN requires high computation power and neuaromorphic devices, but Hybrid SNN is algorithm trained on gpus is faster.

References

[1] Neurons & Synapses - Memory & the Brain - The Human Memory.

[2] F. A. Azevedo, L. R. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. Ferretti, R. E. Leite, R. Lent, S. Herculano-Houzel, and et al. Equal numbers of neuronal and non-neuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology,*, page 513(5):532–541, 2009.

[3] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning,. *nature*, 521(7553):436, 2015.

[4] Nitin Rathi, Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation. In *International Conference on Learning Representations*, 2020.

[5] Simon W Moore, Paul J Fox, Steven JT Marsh, A Theodore Markettos, and Alan Mujumdar. Bluehive-a field-programable custom computing machine for extreme-scale real-time neural network simulation. In *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, pages 133–140. IEEE, 2012.

[6] V. J. Dongre and V. H. Mankar. A review of research on devnagari character recognition. *International Journal of Computer Applications*, (2):0975 – 8887, November 2010.

[7] A. K. Pant, S. P. Pandey, and S. P. Joshi. Off-line nepali handwritten character recognition using multilayer perceptron and radial basis function neural networks. In *Third Asian Himalayas International Conference on Internet (AH-ICI)*, pages 1–5, 2012.

[8] Shailesh Acharya, Ashok Kumar Pant, and Prashnna Kumar Gyawali. Deep learning based large scale handwritten devanagari character recognition. In *Software, Knowledge, Information Management and Applications (SKIMA), 2015 9th International Conference on*, pages 1–6. IEEE, 2015.

[9] I.K. Sethi and B. Chatterjee. Machine recognition of hand-printed devnagri numerals. *IETE J. Res*, 22:532–535, 1976.

[10] U. Pal and B. B. Chaudhuri. Indian script character recognition: a survey. *Pattern Recognition*, 9:1887–1899, 2004.

[11] R. Jayadevan and et al. Offline recognition of devanagari script: A survey. In *Systems, Man, and Cybernetics, Part C: Applications and Reviews*, pages 782–796. IEEE Transactions on 41.6, 2011.

[12] B. Shaw, S. Parui, and M. Shridhar. Offline handwritten devanagari word recognition: A holistic approach based on directional chain code feature and hmm. In *International Conference on Information Technology*, pages 203–208, 2008.

[13] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *In Proc. IEEE*, 86:2278–2324, 1998.

[14] Y. Tang, L. Peng, Q. Xu, Y. Wang, and A. Furuhata. Cnn based transfer learning for historical chinese character recognition. *IAPR Workshopon Document Analysis System*, pages 25–29, 2016.

[15] Z. Wang, Y. Ma, Z. Dong, N. Zheng, and P. Ren. piking locality-sensitive hash: Spiking computation with phase encoding method. In *n: 2018 International Joint Conference on Neural Networks (IJCNN)*, page 1–7, July 2018.

[16] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier. Stdp-basedspiking deep convolutional neural networks for object recognition. In *Neural Networks 99*, page 56–67, Mar 2018.

[17] W. Gerstner and W. M. Kistler. Spiking neuron models: Single neurons,populations, plasticity. Cambridge: Cambridge University Pres, 2002.

[18] A. Tavanaei, Z. Kirby, and A. S. Maida. Training spiking convnets by stdp and gradient descent. In *International Joint Conference on Neural Networks(IJCNN)*, pages 1–98, July 2018.

[19] A. Krizhevsky. Convolutional deep belief networks on cifar-10. 2010.

[20] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. *Advances in neural information processing systems*, 28, 2015.

[21] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. pages 1–14, 2014.