Deep Signature ID: CNN and SNN algorithms in Signature Identification

[1]Shehzad Ibrahim K. M., [2]Rahul Xavier, [3]Sanjay Steephen, [4]Sam Stephen Paulson, [5]Sabna N
[1,2,3,4]Student, [5]Assistant Professor, Rajagiri School of Engineering and Technology, Kochi, India,
[1]shaiz98@gmail.com, [2]rahul619xr@gmail.com, [3]sanjaysteephen1418@gmail.com,
[4]samstephenpaulson98@gmail.com, [5]sabnan@rajagiritech.edu.in

Abstract **Image recognition is one of the most trending topics in data science. With the onset of techniques in Machine Learning and Deep Learning, the classic method of image recognition was redefined. Today, Convolutional Neural Network is an algorithm in Deep Learning that is named one of the best image recognition algorithms and can be used for most image recognition applications. In today's world, signature forging is common and almost unsuspecting to the human eye. Hence, Convolutional Neural Network can be employed to accurately identify the signature of a person. The signature of each person must be trained into the Convolutional Neural Network before deploying the algorithm for use. A drawback in Convolutional Neural Network is that each person's data should be explicitly trained for it to work. This drawback can be overcome by Siamese Neural Network which can to be used to make a universal signature comparison algorithm and hence identify anyone's signature by comparison. This paper works on both the algorithms and obtains the comparison of performances of both Convolutional and Siamese Neural Networks.**

Keywords —Convolutional Neural Network, Siamese Neural Network, machine learning, signature recognition, image recognition

I. INTRODUCTION

Convolutional Neural Network (CNN) [1, 2] is believed to be one of the best algorithms for image recognition. New and effective techniques for automatic writer identification and verification have been studied in [3-6].The objective of this paper is to verify whether this claim stands in the field of human handwritten signatures and whether CNN can identify the author of signature with high accuracy. The next objective is to test a new algorithm introduced to the family of Artificial Neural Networks named Siamese Neural Networks (SNN) [7, 8] for the same application. The paper also aims to compare the performance of CNN and SNN in order to find the advantages and disadvantages of both, for the application under consideration.

Convolutional neural network is high-accuracy image recognition algorithm. This algorithm can be theoretically used to verify handwritten scripts and signatures. This is to be verified by building a CNN structure for signature recognition. Siamese Neural Network is another deep learning algorithm that can be used to identify similarity of images by method of comparison. There are many methods to implement CNN out of which one of the most feasible and efficient method would be to use the Keras Library in python to construct and train the algorithm. The feature extractor for CNN is built using convolutional layers and the number of convolutional layers determines the accuracy and training efficiency of the algorithm. Since both the algorithms are relatively new and not much experimented in the domain of handwriting/signature recognition, this paper compares the performance of SNN and CNN. Simulation studies were performed on Google's cloud platform - Google Colab as it provides free GPU support.

A. Convolutional Neural Networks

A Convolutional Neural Network (ConvNet) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. A CNN contains convolutional and pooling layers, fully connected layers, and finally, the output layer.

a) Convolutional Layer: This layer does 2-dimensional convolution on images for feature extraction. It takes the original image and runs a filter over it and obtains the output image. The filter is used for feature extraction. The image after convolutional operation contains the highlighted features from the original image. The convolutional layer uses multiple kernels/filters such as sharpening, blur, emboss, etc. to perform convolution on the input image to extract features. The number of convolutional layers is defined by the complexity of the input image. To identify simple shapes, a single convolutional layer will be enough.

 DOI : 10.35291/2454-9150.2020.0562

But to identify complex structures like buildings, faces, signatures etc., at least 3 layers may be necessary. The lower limit on the number of convolutional layers is such that the algorithm should be able to identify the features necessary for classification, and the upper limit is that the algorithm should not refer to unnecessary details in an image for classification. There are several filters used in the convolutional layer out of which some are already specified, like edge detection filter and sharpening filter while others are learned during the training in-order to find features that are virtually invisible to the human eye which makes the CNN algorithm powerful. It can make its own feature detection kernel in-order to extract features. In this way it automatically extracts features without human intervention. A typical convolutional layer operation is shown in Fig. 1. This figure clearly shows us how a kernel (K) of size 3*3 perform convolution on an image (I) of size 7*7 to obtain a feature map (I*K) of size 5*5.

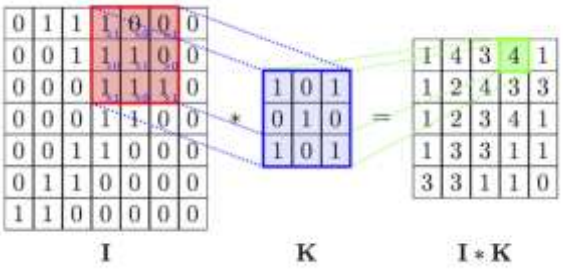

Fig. 1 A typical Convolutional layer operation

b) Pooling Layer: Pooling layer is used to decrease the complexity and size of further calculations and to highlight the extracted features from the convolutional layers. This layer works on each feature map to decrease the resolution and size. Pooling layer is essential so that the algorithm is efficient. In the simulation studies max pooling, which is a method of taking maximum value from four pixels in order to make the new pixel, has been employed. This makes the size of the feature map a fourth of the original size while sustaining all the extracted features. Fig. 2 demonstrates a typical max pooling operation. The figure demonstrates how a feature map of resolution 4*4 undergoes 4:1 max-pooling to obtain a pooled image of size 2*2. In 4:1 max pooling, a cluster of 4 pixels is replaced by a single-pixel which has maximum value among the four pixels.

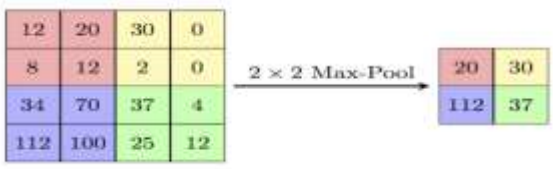

Fig. 2 Pooling layer

c) Fully connected Layer: From this point on, the image is converted to one dimensional as this layer is one dimensional. Fully connected layer is the layer in which actual training using manipulations in weights and biases happen using the feature extracted using the previous layers. After training, this layer has data on how to classify the image based on input data.

d) Output Layer: The output from the fully connected layer is a bunch of numbers that do not mean a lot until they are normalized using an appropriate function. The best function used for normalization in image classification application is SoftMax function. The reason why SoftMax is useful is because it converts the output of the last layer into a probability distribution.

B. Siamese Neural Networks

Siamese networks are a special type of neural network architecture. Instead of a model learning to classify its inputs, the neural network learns to differentiate between two inputs. In image recognition domain, the Siamese neural network learns to find the amount of similarity between 2 input images. The advantage of SNN is that it lets us create a universal image classifier for signature that can work with any signature without explicitly training the network with that signature's dataset. This is because, SNN effectively learns to compare two signatures and tell if they are similar instead of learning to identify a particular signature. In the case of CNN, the entire neural network needs to be explicitly trained again when a new person's signature is added or when a signature is removed. This is the main challenge posed by CNN which can be overcome using SNN. Fig. 3 shows a general Siamese neural network, the layers of which are as follows.

a) Convolutional Layer: The input layer of Siamese Neural Network is a convolutional layer for feature extraction from the input images. This layer is used for getting the feature vector from both the input images. If the images are similar, the feature vector from both the images will be similar; otherwise the feature vector will not be similar.

b) Embedding Layer: The Embedding layer is essentially a multidimensional space that is used to plot the location of an image. Each dimension in the embedding layer represents a feature map, and the position of the image is plotted on the embedding layer based on the feature vector obtained from the image upon passing it through the convolutional layer. Hence each image will have a location in the multidimensional space, and similar images tend to cluster together as their values from the feature maps will be closer.

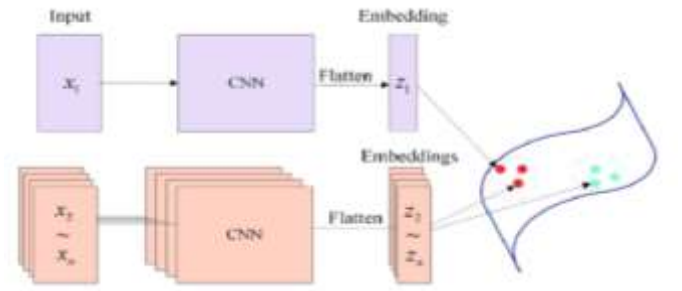

Fig. 3 Siamese Neural Network

 DOI : 10.35291/2454-9150.2020.0562

c) Output Layer: The output layer works in two parts while comparing two images. Firstly, the Euclidean distance between the two images in the multidimensional space is found out. This is done for all the images that have to be compared for image recognition. Then the distance is normalized using an activation function to a value between 0 and 1 for comparison. When the images are fully similar the output will be 1 and when they are fully unique the output will be 0.

II. LIBRARIES FOR IMPLEMENTATION

A. *TensorFlow and Keras*

These are the two libraries using which the SNN and CNN layers are built. TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state of the art in ML and developers easily build and deploy ML powered applications. Using TensorFlow for Machine Learning is complicated as the parameters and layers have to be added manually. So Keras is a library for deep-learning which is used on top of TensorFlow to make the implementation of TensorFlow more streamlined and easier.

B. *Numpy*

NumPy is the most important library when it comes to working with images. NumPy is short for NumbersPython and it helps to work on images easily as it lets us save images as matrices. It helps us to make fine-tuned modifications to an image.

C. *MatPlotLib*

It is a library in python that can be used to perform plot operations. It has been used to plot final results and for debugging.

D. *Platform – Google Colab*

Google Colab is a free platform to train and test machine learning models. It supports Python 2 as well as Python 3 and it uses a Linux system in its backend. It lets us save the dataset to Google Drive and train directly from there. Google Colab is preferred for training because training deep learning models is a GPU intensive process and needs high capacity GPUs. Google Colab uses a Tesla K80 GPU. Training using low-end devices will take very long time and might also result in failure of GPU.

III. METHODOLOGY

The objective of this paper is to obtain a comparison of performances of Convolutional and Siamese Neural Networks for signature identification. This section explains how signature recognition using CNN and SNN were planned and implemented. This section also describes how the dataset was collected or generated. The method of training and debugging are also elaborated.

A. *Signature Recognition using CNN*

The steps of implementation are as follows.

a) Collection of dataset: Signature samples of 8 people were taken and each person was asked to put signature in a sheet of paper in which there were 88 boxes to record signature as in Fig. 4. The sheets were scanned, cropped signature by signature and then uploaded to Google Drive. In Google Drive, each person's signature is grouped by folder. Thus, there were 8 folders with 88 signatures of each person. Then, they are put in a parent folder which is named Training data. To the train using this dataset, Google Drive was connected to Google Colab and the training folder for training data was referred. Each picture in a folder is automatically classified as belonging to the person whose name is the name of the folder. This way, the supervised training can be done much efficiently without having the trouble of labeling images. The result is that there are 8 classes with 88 images to train for each class. But, 88 images are too less data for training as signature of a person can have slight variations. Hence, the ImageDataGenerator Library that comes along with Keras were used, which lets us create thousands of images from the existing database by rotating, stretching, zooming and skewing already existing images in the database. Through this method, 2000 images were created from 88 images of each person.

Fig. 4 Data Collection

The type of neural network was specified as Sequential Neural Network, in which each layer is connected only to the next layer and is added one after the other. The first layer is Conv2D layer which is short for 2D convolutional layer. It takes in images of resolution 128*128 and the filter size was taken as 3*3. The next layer to add is pooling layer which is used to reduce the size of the extracted features. A pooling layer that uses the max pooling function with 2*2 resolution was added. This means that the pooling layer takes 4 pixels and merges them into one which has the highest pixel value. Then, more convolutional and pooling layers for feature extraction were added. The number of convolutional and pooling layer depends on the complexity of the input images for classification. Classifying simple shapes might require only one convolutional layer whereas classifying faces and signature will need more convolutional

 DOI : 10.35291/2454-9150.2020.0562

layers. The next layer is a flattening layer that does the simple job of converting the 2D matrix of the image to a 1D matrix so that the image can be input to the fully connected layer which only accepts 1D matrix. Then the most important layer for training is added which is fully connected layer, where the learning happens. The weights and biases in the fully connected layers are altered to classify the images. A fully connected layer with 256 neurons was added. The final layer is the output layer and the number of neurons in the output layer must be equal to the number of classes, i.e. the number of folders in the training dataset. The activation function used is SoftMax because it gives the probability of belonging to a class. Once all the layers are specified one by one, it is compiled and connected in order to form a proper neural network. At this step it points out if there any error in the connections and layers.

b) Training: Once the architecture of the Neural Network is built, it is ready to be trained using the dataset from Google Drive. So, Google Drive is linked to the Google Colab and parameters for training are defined. The training is connected to WandB website which records the parameters of the training for later analysis. One of the most important parameters in training a neural network is the number of epochs. Using less epochs can lead to underfitting of data where the network does not learn the parameters to capacity and it results in less accuracy. Using more epochs can lead to overfitting of data which means the trained model latches on to unnecessary details in the training data and becomes overly reliant on the training data. Even though this increases the validation accuracy, it reduces the accuracy when it is run with new data. So best method to find the optimum number of epochs is to use the training analysis to find the point where the accuracy curve flattens and will not increase further. At this point, the training is said to be successfully completed and the algorithm is ready to be tested on the testing dataset.

c) Debugging and Improvements: Now that initial stages are done, improvements can be made in the architecture by adjusting some parameters. The most important parameter to be adjusted for improvement in accuracy is to add/remove the convolution and pooling layers. It has to be manually increased and decreased until the best results are obtained and thus, the best possible architecture. The CNN is finally debugged and optimized and now the architecture of CNN for Signature recognition can be deployed. The epoch should also be set to the optimum number by looking at the training analysis.

B. Signature Recognition using SNN

A signature recognition using SNN measures the similarity between two images by comparison. The value of the similarity function is near to 1 if the signatures are similar, and are near to 0 if the images are not similar. Fig. 5 shows a typical Siamese neural network. In a Siamese Neural Network, two inputs are given simultaneously to its two input branches for comparison. CNN layer can extract features for comparison. For the purpose of comparison of simple images, the CNN part can be avoided and images can be directly passed to the embedding layer. The embedding layer finds the position of both the images in a multi-dimensional space. The Euclidean distance between the points is found out and normalized using the sigmoid function. The normalized value represents the amount of similarity between the input images.

Fig. 5 SNN Framework

a) Preparation of dataset for Triplet Loss Training: The method of dataset preparation for training SNN is different from that of CNN. For CNN, images can be directly trained for classification. But since SNN learns to compare two images and tell the similarity, SNN architecture has to be trained with pairs of similar and dissimilar images with target values as 1 for similar images and target value as 0 for dissimilar images. As shown in Fig. 6, pairs of images are chosen at random from the dataset folder, and they are labeled a target of 1 if they are signatures of the same person, else they are labeled a target of 0. In this way, an array of pairs of images is built along with a label array, which totals to about 6000 pairs. Thus, Triplet Loss training where pairs of similar and dissimilar images are trained so that the algorithm learns to tell if the images are similar or not, has been employed for training SNN. The completed array for triplet loss training is as shown in Fig. 6.

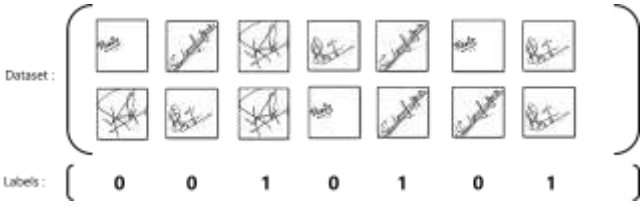

Fig. 6 Triplet Loss Training Dataset

b) Building the framework for SNN: The SNN framework adheres to what its name suggests, it looks similar on top and bottom. It has two input channels for an image each, and an output channel which gives the similarity score between 0 and 1. Both the images are input to a convolutional layer, which has multiple kernels that extract specific features. Both the CNN in this architecture will be trained together, hence they will have same weights, biases and connections, and they will be fully identical in every aspect. The CNN passes the extracted feature map to the next layer. The next layer is embedding layer which is

 DOI : 10.35291/2454-9150.2020.0562

responsible for making a multidimensional space which has dimensions equal to the number of feature maps. Each image is plotted in the multidimensional space using the data obtained from its feature maps and the positions will be called h1 and h2. The next step finds the Euclidean distance between h1 and h2 from the multidimensional space. This Euclidean distance represents the similarity between both the images. But random number which represents distance does not give immediate information. So, this distance is passed to activation function, which gives us the normalized final answer which represents the similarity between the images.

c) Training: This step involves training the compiled SNN architecture using the triplet loss dataset saved in array format. The training is connected to WandB website which records the training parameters for later analysis. Fixing the number of epochs using the analyzed data is the next step, which involves observing the graph of training accuracy, and when accuracy reaches a peak and saturates, that epoch is called the optimum number of epochs. The optimum number of epochs is found out and SNN is trained again using that many epochs.

d) Debugging and improvements: The factor that affects the accuracy of SNN is its input resolution which has to be fixed using trial and error method. More resolution means more computational time but better accuracy. Hence an optimum resolution needs to be fixed which can clearly render a signature and an input resolution of 128*128 was chosen for the simulation studies.

IV. Results and Discussions

The convolutional layer of CNN is the most important part of the algorithm for image recognition. The number of convolutional layers must be optimized for peak performance and accuracy. The number of convolutional layers required for an image recognition system is estimated using general mathematics or intuition and then the number is refined and confirmed by trial and error method. When the number of convolutional layers is too less, the algorithm fails to identify or extract the key features required to recognize that image which leads to less accuracy of the algorithm. On the other hand, if the number of convolutional layers is too high, it results in the algorithm depending on too many unnecessary details to recognize an image, and it also results in unwanted complexity in training and executing the algorithm. Hence each application of CNN has an optimum number of convolutional layers for best accuracy. The complexity of the images that needed to be trained and recognized determines the number of convolutional layers and it was estimated that the number of convolutional layers required falls between 2 and 5 for the application under consideration. Hence the performance of algorithm with 2, 3, 4, and 5 Conv2D layers were compared. The next aspect of optimizing the algorithm is to train the algorithm with the optimum number of epochs, which is the number of times a dataset is looped to train the system. While the algorithm is being trained on the dataset, it can be seen that the accuracy keeps increasing and after a particular number of epochs the accuracy remains almost constant. It is best to stop training the algorithm when this happens and the maximum number of epochs was chosen as 25 for the present simulation studies. After analysing the training accuracy curve vs. number of epochs, the optimum number of epochs required can be found out. Then the algorithm can be retrained using the optimum epoch number for best results.

A. Results of Signature Recognition using CNN

For signature recognition using CNN, the algorithm was trained with 2, 3, 4, and 5 Conv2D layers. The results of training using 2, 3, 4 and 5 convolutional layers is as shown in Fig. 7 which shows accuracy versus number of epochs. It gives an idea about how many epochs are required to properly train the algorithm without overfitting or underfitting. Comparing the four runs of the CNN algorithm through 20 epochs, it can be seen that, CNN using 3 Conv2D layers offers the maximum accuracy of 83%. It can also be noticed that the training of the algorithm stabilizes at about 5 epochs, and hence, the best accuracy without overfitting can be obtained at 5 epochs.

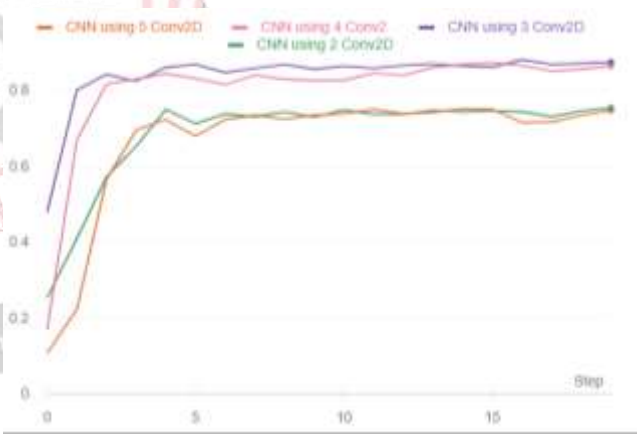

Fig. 7 Result of training CNN for Signature Recognition

Fig. 8 shows the optimized CNN architecture with three sets of convolutional and pooling layers for testing purposes. The architecture shows an input layer which accepts images of resolution 128*128. This resolution is best for identifying the signatures in the dataset used. The resolution of kernel is adjusted so that the convolutional filters can pick up features the best way possible. The optimum resolution of kernel we obtained was 3*3. Pooling was done to optimize the extracted features without losing much information of the image. Since the signatures are black on white background, max-pooling was used which retains the signature information, while compressing the resolution of the image by 4:1. Dense layer size should be directly proportional to the amount of data that should be learned in order to recognize signatures. Number of units in

 DOI : 10.35291/2454-9150.2020.0562

the dense layer was set to 256. The activation function at the output layer was set to SoftMax as the probablity of the input image belonging to each of the classes was needed. While SoftMax is not the best activation function to be used while training, it is very important to get the probablity value at the output. After all these layers were compiled, the system was finally tested with the testing set containing images with heavier distortions generated using ImageDataGenerator and the accuracy of CNN was found to be 78% for the testing set.

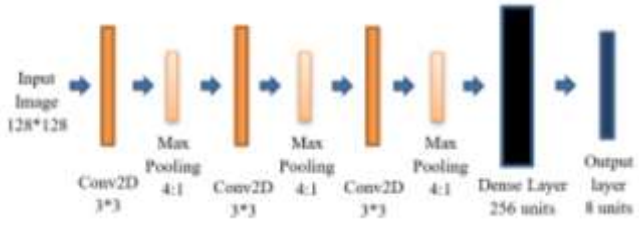

Fig. 8 Optimized CNN architecture for signature recognition

B. Results of Signature Recognition using SNN

In the implementation of SNN for the present simulation studies, initial convolutional layer present in the general structure of Fig. 5 was omitted. The images were compared not by the features extracted, but by pixel-by-pixel comparison which gives the simplest form of SNN and gave satisfactory results when implemented for signature recognition. From the performance plot, it can be seen that the SNN training was done successfully and the accuracy curve flattens after 4 epochs; the maximum accuracy attained was approximately 98% as in Fig. 9. Fig. 10 shows the optimized SNN architecture for testing purposes. The input resolution of the image was taken to be 128*128, which as mentioned before is the best resolution to identify images in the dataset. Units in the dense layer or embedding layer were set to 128 for efficient learning and training speed. A complicated dense layer results in more training and testing time without much improvement to the results. Activation function used in SNN is ReLu which stands for Rectified Linear Unit. It is the best activation function in terms of learning rate. It is used in SNN as opposed to CNN, because the output here is binary. ReLu can only give binary output. SNN was also tested with the testing set generated and the accuracy of SNN was found to be 84% for the testing set under consideration.

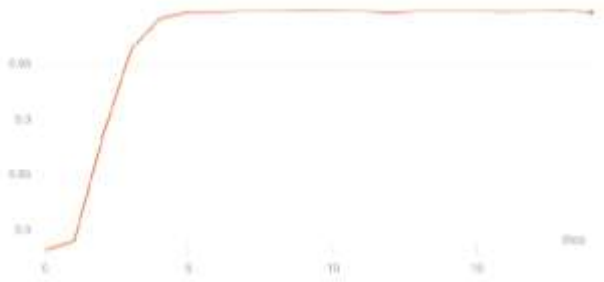

Fig. 9 Result of training SNN for Signature Recognition

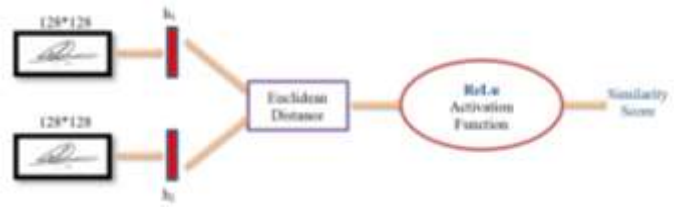

Fig. 10 Optimized SNN architecture for signature recognition

V. CONCLUSIONS

Convolutional Neural Network (CNN) is known as the best deep learning algorithm for image recognition. Siamese Neural Network (SNN) is a developing algorithm used for the purpose of comparison of two data from a similar data type.

CNN and SNN were used to perform image recognition using a signature dataset. The result of training the algorithms provided us with valuable insights about the working, implementation and applications of Convolutional Neural Network and Siamese Neural Network. As seen from the training graphs represented in Fig. 7 and Fig. 9, a maximum validation accuracy of 83% was obtained using CNN, and as for SNN, a maximum validation accuracy of 98% was obtained. The testing accuracy obtained was 78% for CNN and 84% for SNN. Even though this might suggest the SNN model developed is much superior to CNN, the reality is that the high accuracy of SNN is conditional. CNN is an algorithm that uses features detected from training images to find a similar feature in the testing image. This means that CNN can detect a similar signature in different orientations, sizes and in different regions in the image. Hence CNN can find the trained image wherever it is in the test image. Considering this, 78% accuracy is good and it can be concluded that CNN developed is a reasonably accurate algorithm that can be used to identify signature made in a random position in the frame.

The SNN algorithm developed in this work does pixel by pixel comparison of the images since the input convolutional layer is omitted and hence the test image needs to be in the same orientation and location as in the reference image. This does not mean that the images size should also be same since the algorithm automatically resizes the images into predefined resolution. Given this condition, an accuracy of 84% is good. In this context, it can be concluded that SNN delivers very good accuracy when the input conditions are met, i.e. when the location of the signature has a very definite boundary as in a bank cheque. Also, in the case of CNN, the entire neural network needs to be explicitly trained again when a new person's signature is added or when a signature is removed. This challenge posed by CNN can be overcome using SNN.

Considering the application of CNN and SNN in real life scenarios, CNN is the best algorithm to be used when the dataset and classes to be recognized is not dynamic, and hence remains same for a reasonable period. The main

 DOI : 10.35291/2454-9150.2020.0562

advantage of CNN is that it can identify images that are slightly distorted, or not exactly similar to the training dataset. SNN algorithm is used when the dataset and classes for recognition is dynamic, or rapidly changing. Since SNN does not require explicit training for each class added or removed, SNN can deployed in such environments.

ACKNOWLEDGMENT

Authors express their sincere thanks to Rajagiri School of Engineering and Technology for providing the facilities required for doing the work.

REFERENCES

[1] Luiz G. Hafemann, Robert Sabourin and Luiz S. Oliveira, "Writer-independent feature learning for Offline Signature Verification using Deep Convolutional Neural Networks," in *International Joint Conference on Neural Networks (IJCNN)*, Vancouver, BC, 2016.

[2] Weixin Yang, Lianwen Jin and Manfei Liu, "DeepWriterID: An End-to-End Online Text-independent Writer Identification System," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 45-53, Mar-Apr. 2016.

[3] Marius Bulacu and L. Schomaker, "Text-independent writer identification and verification using textural and allographic features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 701-717, April 2007.

[4] Donato Impedovo and Giuseppe Pirlo, "Automatic Signature Verification: The State of the Art," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 5, pp. 609-635, Oct. 2008.

[5] Rajiv Jain and David Doermann, "Offline Writer Identification Using K-Adjacent Segments," in 2011 International Conference on Document Analysis and Recognition, September 2011, pp. 769-773.

[6] Axel Brink, Marius Bulacu and Lambert Schomaker, "How Much Handwritten Text Is Needed For Text-Independent Writer Verification and Identification," 2008 19th International Conference on Pattern Recognition, Tampa, FL, 2008, pp. 1-4.

[7] Su Tang, Shan Zhou, Wenxiong Kang, Qiuxia Wu and Feiqi Deng, "Finger vein verification using a Siamese CNN", *IET Biometrics*, vol.. 8, no. 5, pp. 306 – 315, 2019.

[8] Xuning Liu, Yong Zhou, Jiaqi Zhao, Rui Yao, Bing Liu, and Yi Zheng, "Siamese Convolutional Neural Networks for Remote Sensing Scene Classification", *IEEE Geoscience And Remote Sensing Letters*, vol. 16, no. 8, Aug. 2019.

 DOI : 10.35291/2454-9150.2020.0562