International Journal of Scientific Research in Science, Engineering and Technology

Print ISSN - 2395-1990
Online ISSN : 2394-4099

OPEN ACCESS

Available Online at : www.ijsrset.com
doi : https://doi.org/10.32628/IJSRSET23103216

A Survey : Deep Learning Approaches for Signature Verification

Deepali Narwade, Vaishali Kolhe

Department of Computer Engineering, D. Y. Patil College of Engineering, Akurdi, Pune, Maharashtra, India

ARTICLEINFO

Article History :

Accepted: 01 Aug 2023
Published: 28 Aug 2023

Publication Issue :
Volume 10, Issue 4
July-August-2023

Page Number :
291-312

ABSTRACT

Since a person's signature serves as the primary authentication and authorization method in legal transactions, there is a greater need than ever for effective auto-mated signature verification solutions. The fact that signatures are already recognized as a popular way of identity verification gives signature verification systems a significant edge over other types of technologies. The methods used to solve this problem and a signature verification system can be categorized into two categories: online and offline. An electronic tablet and pen that are connected to a computer are used in the online technique to extract information about a signature and collect dynamic data for verification purposes, such as pressure, velocity, and writing speed. Offline signature verification, on the other hand, employs signature images that have been recorded by a scanner or camera and involve less electronic management. Extracted features taken from the scanned signature image are used in an offline signature verification system. The main contribution of this study is how we can use deep learning approaches/networks for the task of offline signature verification systems.

Keywords: Deep Learning, Signature Verification, RNNs, CNNs.

I. INTRODUCTION

Handwritten signatures are widely used in life. With the development of machine learning and artificial intelligence, the research on handwritten signature verification is also deepening [17]. The signature serves as the authority for all legal transactions. Therefore, the need for signature verification grows. Individuals' handwritten signatures are distinctive and impossible to replicate. The technique is simple to understand and reliable. The fact that signatures are already widely used as a means of identity verification gives signature verification systems a significant edge over other types of technology. Handwritten signature verification is still an active research field nowadays. Depending on the acquisition considered, it can be categorized as i) off-line, the signature is acquired in a traditional way by signing with an ink pen over the paper and then digitizing the image; and ii) online, the signature is acquired using electronic devices, having therefore not only the image of the signature, but also the signing information of the entire capturing process (time

sequences). Online handwritten signature verification has widely evolved in the last 40 years.

From the original Wacom devices specifically designed to acquire handwriting and signature in office-like scenarios to the current mobile acquisition scenarios in which signatures can be captured using our own personal smartphone anywhere. However, and despite the improvements achieved in the acquisition technology, the core of most of the state-of-the-art signature verification systems is still based on traditional approaches such as Dynamic Time Warping (DTW), Hidden Markov Models (HMM), and Support Vector Machines (SVM). This aspect seems to be a bit unusual if we compare with other biometric traits such as face and fingerprint in which deep learning has defeated by far traditional approaches, and even in tasks more related to signature verification such as handwriting recognition, writer verification, and handwritten passwords [1].

There are two sorts of handwritten signature verification: online and offline. The online method extracts data about a signature using an electronic technique and a computer and employs dynamic data for verification such as pressure, velocity, and writing speed. Offline signature verification employs photographs of the signer's signature taken with a scanner or camera and relies less on electronic control. The applications of biometric identification and verification [13–16] are present in documents and actions of everyday life such as passports, driver's licenses, migration, applications of security, personal device login, voter registration, medical records, and smart cards [7]. In the process of signature identification, the system should be provided with a user's signature to compare it with the various signatures registered in the datasets, and the similarity results will be calculated. The most similar result will indicate the identified user, whereas there are two basic approaches for signature verification. These are writer-dependent and writer-independent approaches. In the writer-independent approach, one paradigm is trained for the whole user population and the query signature is matched with the reference signatures in a similarity/dissimilarity space. For the reason that it does not require the systems to be retrained when adding a new writer, this approach is preferred by most researchers [14].

Signature verification has been an active research area because of the long-term and widespread use of signatures for personal authentication, however, it remains a challenging task owing to large intra-class variations and skilled forgeries. Signature verification can be categorized as online or offline, depending on the signature acquisition method. Traditionally, online signature verification systems perform better than offline systems because more dimensions of information are available [10]. Deep Learning (DL) has become a thriving topic in the last years, allowing computers to learn from experience and understand the world in terms of the hierarchy of simpler units. DL has enabled significant advances in complex domains such as natural language processing and computer vision, among many others. The main reasons to understand the high deployment of DL lie on the increasing amount of available data and also the deeper size of the models thanks to the increased computer resources. However, there are still some tasks in which DL has not achieved state-of-the-art results due to the scarcity of available data and therefore, the inability to train and use those traditional deep learning architectures [5].

The organizations receive a lot of scanned or photographed documents from their customers. Therefore, an important technology is to automatically detect the signatures on the documents and then match them with the customer's signature in the system for verification [6]. Handwritten signature verification is a crucial yet challenging problem. While previous studies have made great progress in this problem, they learn signature features passively from

given existing data [8]. In-air signature verification is vital for biometric user identification in contact-less mode. The state-of-the-art methods use heuristics for signature acquisition, and provide insufficient data to train neural networks for the verification [9]. The use of signatures for personal identification and verification is quite common. Signatures are validated for many documents such as Bank cheques and legal transactions. The necessity for effective automated solutions for signature verification has grown as signatures are now a prerequisite for both authorization and authentication in legal activities. Two images—the original signature and the test signature—are used as input in this process. Prior to feature extraction, these photos are pre-processed. To determine whether the signature is fake or not, the characteristics that were extracted are compared, and the difference in error values between them is examined. The key advantages of signature verification systems (SVS) over alternative verification technologies include time and energy savings, a reduction in the risk of fraud during authentication, and a reduction in the likelihood of human error during the signature process [24].

Offline signature verification is a challenging task, particularly in distinguishing between genuine signatures and skilled forgery. How to distinguish features and how to calculate the similarity score are key issues in signature verification. A comparison offline handwriting signature verification algorithms for solving skilled forgery problems by using Artificial Neural Network (ANN) with our proposed methodology as the Histogram Oriented Swerve Angle (HOSA). The main methodology were developed by the technique of skeletonization also known as the thinning process, which is an important step in the signature pre-processing. After the pre-processing process, we use the histogram-oriented swerve angle for determining the signature image feature extraction. The verification process is accessed with an artificial neural network (ANN) classifier [15]. Signature verification usually includes two stages of training and testing. In the training stage, different numbers of real signatures are used for pre-processing and feature extraction and then put into the classifier to obtain the model. In the test stage, the signatures are put into the classifier for comparison and output verification result. In the feature extraction stage, offline signature image features are called static features, which are mainly divided into local features and global features. Local features are mainly divided into texture features and gradient features, and global features are mainly geometric features. Online signature data features are called dynamic features, which are mainly divided into parameter-based features and function-based features. Parameters-based mainly refer to the signature duration and the number of pen tip upwards. Function-based features mainly refer to signature trajectories and pressure data. Dynamic features based on functional features generally have better results [17].

Deep Learning

Deep learning uses artificial neural networks to process intricate computations on enormous amounts of data. It is a type of artificial intelligence that is based on how the structure and operation of the human brain.
Deep learning methods are used to train machines by teaching them from examples. Healthcare, e-commerce, entertainment, and advertising are just a few industries that often use deep learning.

Defining Neural Networks

A neural network is structured like the human brain and consists of artificial neurons, also known as nodes. These nodes are stacked next to each other in three layers:

1. The input layer
2. The hidden layer(s)
3. The output layer

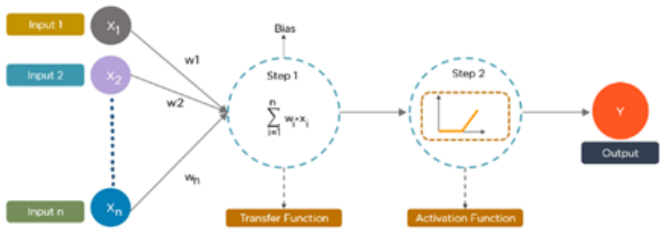

Fig.1: Neural Network

Artificial Neural networks (ANNs) are widely used in pattern recognition due mostly to their strength and usability. A straightforward method is to first take numerous samples from various signers and extract a feature set representing the signature (length, height, duration, etc.). The ANN must then understand the connection between a signature and its class (either "genuine" or "forgery") in order to proceed to the next stage. Once the network is aware of this link, test signatures that can be identified as coming from a certain signer can be submitted to it. ANNs are thus well adapted to modeling the overall properties of handwritten signatures. Handwritten signature verification systems have been shown to be highly sensitive to signature complexity [12]. Offline handwritten signature verification dictates a neural network to learn from training signature features and tell whether a tested signature is genuine or not. One common idea is that the author first input a reference signature and a tested signature of their original and grey-inverse version. After a couple of convolution and attention, each two of them are concatenated together. With the strategy that all the outcomes should be same, the model is trained to focus more on the signature stroke features than on colour [13].

This paper presents a comprehensive review of the latest studies and results in the last years in the field of online and offline handwritten signature verification. Here 42 research papers were used to make a comparison between different classifications techniques used in each system. In addition, the general limitations and advantages of deep-learning techniques that are used to classify or extract signature features.

Paper is organized as follows. Section II describes background details of Deep Learning Techniques with their advantages, disadvantages and applications. After this, literature review is given in Section III. Section IV presents methodology used for verification of signatures. Finally, Section V presents conclusion.

II. RELATED WORK

Deep learning techniques rely on artificial neural networks (ANNs), which replicate how the brain processes information, in addition to self-learning representations. Algorithms use unknown components in the input distribution during the training phase to extract features, classify objects, and discover useful data patterns. On many levels, this occurs when algorithms are used to create models in a way that is akin to teaching machines how to learn on their own. Deep learning models employ a number of different algorithms. No network is perceived as perfect, yet some algorithms are better suited to complete specific jobs. To make the best decisions, it is advantageous to gain a complete understanding of all important algorithms.

Types of Algorithms Used in Deep Learning

Here is the list of the top 10 most popular deep learning algorithms:

1. Convolutional Neural Networks (CNNs)
2. Long Short Term Memory Networks (LSTMs)
3. Recurrent Neural Networks (RNNs)
4. Generative Adversarial Networks (GANs)
5. Radial Basis Function Networks (RBFNs)
6. Multilayer Perceptron's (MLPs)
7. Self-Organizing Maps (SOMs)
8. Deep Belief Networks (DBNs)
9. Restricted Boltzmann Machines(RBMs)
10. Auto-Encoders

1. Convolutional Neural Networks (CNNs)

Artificial neural networks do incredibly well in machine learning. In many datasets, including those with images, audio, and text, neural networks are used. Different types of neural networks are employed for various tasks. For example, to predict the order of words, recurrent neural networks—more specifically, an LSTM—are used. Similarly, to classify images, convolution neural networks are employed. We're going to create the fundamental building element for CNN in this blog.

In a regular Neural Network there are three types of layers:

1. **Input Layers:** It's the layer in which we give input to our model. The number of neurons in this layer is equal to the total number of features in our data (number of pixels in the case of an image).
2. **Hidden Layer:** The input from the Input layer is then feed into the hidden layer. There can be many hidden layers depending on our model and data size. Each hidden layer can have different numbers of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of the output of the previous layer with learnable weights of that layer and then by the addition of learnable biases followed by activation function which makes the network nonlinear.
3. **Output Layer:** The output from the hidden layer is then fed into a logistic function like sigmoid or softmax which converts the output of each class into the probability score of each class.

In the following stage, known as feedforward, the model is supplied with input, and each layer's output is obtained. We then calculate the error using an error function; some popular error functions are cross-entropy, square loss error, etc. The network's efficiency is gauged by the error function. After that, we calculate the derivatives and backpropagate into the model. Backpropagation is the process that is utilized to reduce loss in general.

Convolutional Neural Network (CNN) is the extended version of artificial neural networks (ANN) which is predominantly used to extract the feature from the grid-like matrix dataset. For example visual datasets like images or videos where data patterns play an extensive role.

CNN architecture

A convolutional Neural Network consists of multiple layers like the input layer, Convolutional layer, Pooling layer, and fully connected layers.

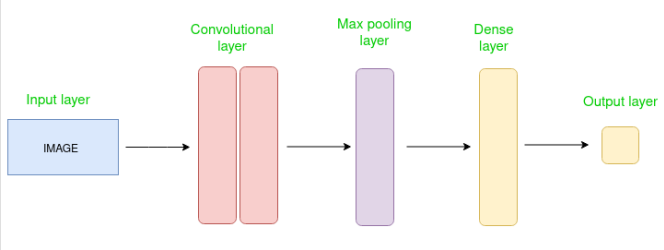

Fig. 2 CNN

The Convolutional layer applies filters to the input image to extract features, the Pooling layer down samples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

Advantages of Convolutional Neural Networks (CNNs):

1. Good at detecting patterns and features in images, videos, and audio signals.
2. Robust to translation, rotation, and scaling invariance.
3. End-to-end training, no need for manual feature extraction.
4. Can handle large amounts of data and achieve high accuracy.

Disadvantages of Convolutional Neural Networks (CNNs):

1. Computationally expensive to train and require a lot of memory.
2. Can be prone to overfitting if not enough data or proper regularization is used.
3. Requires large amounts of labeled data.
4. Interpretability is limited, it's hard to understand what the network has learned.

Applications of Convolutional Neural Networks (CNNs):

1. Face Detection
2. Facial emotion recognition
3. Object detection
4. Self-driving or autonomous cars
5. Auto translation
6. Next word prediction in sentence
7. Handwritten character recognition
8. X-ray image analysis
9. Cancer detection
10. Visual question answering
11. Image captioning
12. Document classification
13. 3D medical image segmentation

2. Long Short Term Memory Networks (LSTMs)

Long Short Term Memory is a kind of recurrent neural network. The output from the previous phase is sent into the current step of an RNN as input. Hochreiter & Schmidhuber created LSTM. It addressed the issue of long-term RNN dependency, in which the RNN can predict words from current data but cannot predict words held in long-term memory. RNN's performance becomes less effective as the gap length increases. By default, LSTM can save the data for a very long time. It is utilised for time-series data processing, forecasting, and classification. A specific kind of recurrent neural network (RNN) called long short-term memory (LSTM) is made to deal with sequential data, such time series, audio, and text. Language translation, speech recognition, and time series forecasting are among the tasks that LSTM networks are particularly suited for because they can learn long-term dependencies in sequential data. It can be challenging for a standard RNN to learn long-term dependencies because there is only one hidden state that is transferred over time. By including a memory cell—a unit that can store data for a considerable amount of time—LSTMs solve this issue.

Three gates—the input gate, the forget gate, and the output gate—control the memory cell. These gates determine what data should be input into the memory cell, removed from it, and output from it. What data is added to the memory cell is managed by the input gate. What information is erased from the memory cell is controlled by the forget gate. Additionally, the output gate regulates the data that is output from the memory cell. This enables LSTM networks to learn long-term dependencies by choosing which information to keep or discard as it moves through the network. The fundamental distinction between the LSTM and RNN architectures is that the LSTM's hidden layer is a gated unit or gated cell. It is made up of four layers that interact with one another to create both the cell state and the output of that cell. The following hidden layer receives these two items after that. LSTMs are made up of three logistic sigmoid gates and one tanh layer, in contrast to RNNs which only feature a single tanh layer. In order to restrict the amount of information that is passed through the cell, gates have been implemented. They choose which information will be needed by the following cell and which should be ignored. The output is usually in the range of 0-1 where '0' means 'reject all' and '1' means 'include all'.

Hidden layers of LSTM :

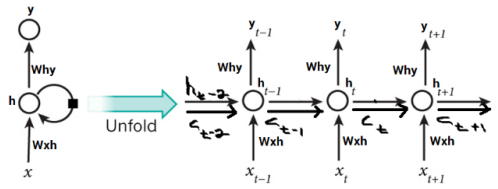

Fig. 3 LSTM

Applications of LSTM:

LSTM models need to be trained with a training dataset prior to its employment in real-world applications. Some of the most demanding applications are discussed below:

1. Language modelling or text generation, that involves the computation of words when a sequence of words is fed as input. Language models can be operated at the character level, n-gram level, sentence level or even paragraph level.
2. Image processing that involves performing analysis of a picture and concluding its result into a sentence. For this, it's required to have a dataset comprising of a good amount of pictures with their corresponding descriptive captions. A model that has already been trained is used to predict features of images present in the dataset. This is photo data. The dataset is then processed in such a way that only the words that are most suggestive are present in it. This is text data. Using these two types of data, we try to fit the model. The work of the model is to generate a descriptive sentence for the picture one word at a time by taking input words that were predicted previously by the model and also the image.
3. Speech and Handwriting Recognition
4. Music generation which is quite similar to that of text generation where LSTMs predict musical notes instead of text by analysing a combination of given notes fed as input.
5. Language Translation involves mapping a sequence in one language to a sequence in another language. Similar to image processing, a dataset, containing phrases and their translations, is first cleaned and only a part of it is used to train the model. An encoder-decoder LSTM model is used which first converts input sequence to its vector representation (encoding) and then outputs it to its translated version.

Disadvantages of LSTM:

As it is said, everything in this world comes with its own advantages and disadvantages, LSTMs too, have a few drawbacks which are discussed as below:

1. LSTMs became popular because they could solve the problem of vanishing gradients. But it turns out, they fail to remove it completely. The problem lies in the fact that the data still has to move from cell to cell for its evaluation. Moreover, the cell has become quite complex now with the additional features (such as forget gates) being brought into the picture.
2. They require a lot of resources and time to get trained and become ready for real-world applications. In technical terms, they need high memory-bandwidth because of linear layers present in each cell which the system usually fails to provide for. Thus, hardware-wise, LSTMs become quite inefficient.
3. With the rise of data mining, developers are looking for a model that can remember past information for a longer time than LSTMs. The source of inspiration for such kind of model is the human habit of dividing a given piece of information into small parts for easy remembrance.
4. LSTMs get affected by different random weight initialization and hence behave quite similar to that of a feed-forward neural net. They prefer small weight initialization instead.
5. LSTMs are prone to overfitting and it is difficult to apply the dropout algorithm to curb this issue. Dropout is a regularization method where input and recurrent connections to LSTM units are probabilistically excluded from activation and weight updates while training a network.

3. Recurrent Neural Networks (RNNs)

Recurrent Neural Network (RNN) is a type of Neural Network where the output from the previous step are fed as input to the current step.

In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence.

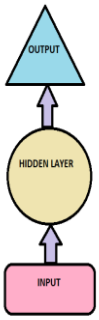

Fig. 4 RNN

RNN have a **"memory"** which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

How RNN works

The working of an RNN can be understood with the help of the below example:

Example: Suppose there is a deeper network with one input layer, three hidden layers, and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let's say, for hidden layer 1 the weights and biases are (w1, b1), (w2, b2) for the second hidden layer, and (w3, b3) for the third hidden layer. This means that each of these layers is independent of the other, i.e. they do not memorize the previous outputs.

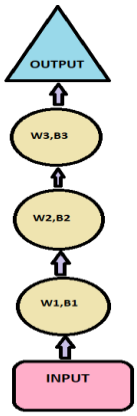

Fig. 5 Example of RNN

Now the RNN will do the following:

1. RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous output by giving each output as input to the next hidden layer.
2. Hence these three layers can be joined together such that the weights and bias of all the hidden layers are the same, in a single recurrent layer.

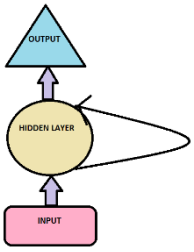

Fig. 6 Loop in RNN

The formula for calculating current state:

$h_t = f(h_{t-1}, x_t)$

Where:

h_t -> current state

h_{t-1} -> previous state

x_t -> input state

Formula for applying Activation function (tanh):

$$h_t = \tanh (W_{hh}h_{t-1} + W_{xh}x_t)$$

Where:

w_{hh} -> weight at recurrent neuron

w_{xh} -> weight at input neuron

The formula for calculating output:

$$y_t = W_{hy} h_t$$

Y_t -> output

W_{hy} -> weight at output layer

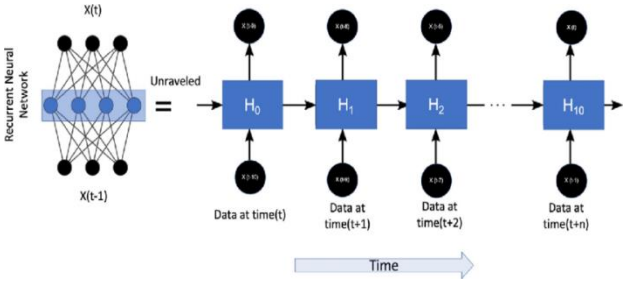

Fig. 7 RNN with Time Stamp

Training through RNN

1. A single-time step of the input is provided to the network.
2. Then calculate its current state using a set of current input and the previous state.
3. The current ht becomes ht-1 for the next time step.
4. One can go as many time steps according to the problem and join the information from all the previous states.
5. Once all the time steps are completed the final current state is used to calculate the output.
6. The output is then compared to the actual output i.e. the target output and the error is generated.
7. The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.

Advantages of Recurrent Neural Network

1. An RNN remembers each and every piece of information through time. It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short Term Memory.
2. Recurrent neural networks are even used with convolutional layers to extend the effective pixel neighbourhood.

Disadvantages of Recurrent Neural Network

1. Gradient vanishing and exploding problems.
2. Training an RNN is a very difficult task.
3. It cannot process very long sequences if using tanh or Relu as an activation function.

Applications of Recurrent Neural Network

1. Language Modelling and Generating Text
2. Speech Recognition
3. Machine Translation
4. Image Recognition, Face detection
5. Time series Forecasting

New trends based on the use of RNNs are becoming more and more important nowadays for modelling sequential data with arbitrary length. The range of applications can be very varied, from speech recognition to biomedical problems. RNNs are defined as a connectionist model containing a self-connected hidden layer. One benefit of the recurrent connection is that a memory of previous inputs remains in the networks internal state, allowing it to make use of past context. One of the fields in which RNNs has caused more impact in the last years is in handwriting recognition due to the relationship that exists between current inputs and past context. However, the range of contextual information that standard RNNs can access is very limited due to the well-known vanishing gradient problem [11].

4. Generative Adversarial Networks (GANs)

A Generative Adversarial Network (GAN) is a deep learning architecture that consists of two neural networks competing against each other in a zero-sum game framework. The goal of GANs is to generate new, synthetic data that resembles some known data distribution.

What is a Generative Adversarial Network?

Generative Adversarial Networks (GANs) are a powerful class of neural networks that are used for unsupervised learning. It was developed and introduced by Ian J. Goodfellow in 2014. GANs are basically made up of a system of two competing neural network models which compete with each other and are able to analyze, capture and copy the variations within a dataset.

How do GANs work?

Generative Adversarial Networks (GANs) can be broken down into three parts:

- **Generative:** To learn a generative model, which describes how data is generated in terms of a probabilistic model.
- **Adversarial:** The training of a model is done in an adversarial setting.
- **Networks:** Use deep neural networks as artificial intelligence (AI) algorithms for training purposes.

In GANs, there is a **Generator** and a **Discriminator**. The Generator generates fake samples of data(be it an image, audio, etc.) and tries to fool the Discriminator. The Discriminator, on the other hand, tries to distinguish between the real and fake samples. The Generator and the Discriminator are both Neural Networks and they both run in competition with each other in the training phase. The steps are repeated several times and in this, the Generator and Discriminator get better and better in their respective jobs after each repetition. The work can be visualized by the diagram given below:

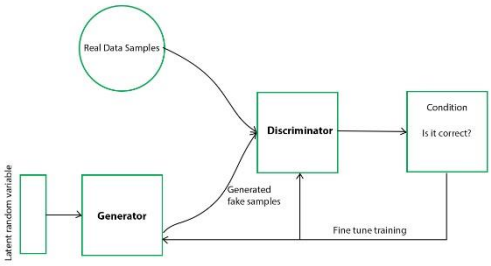

Fig.8 Generative Adversarial Network Architecture and its Components

Here, the generative model captures the distribution of data and is trained in such a manner that it tries to maximize the probability of the Discriminator making a mistake. The Discriminator, on the other hand, is based on a model that estimates the probability that the sample that it got is received from the training data and not from the Generator. The GANs are formulated as a minimax game, where the Discriminator is trying to minimize its reward **V(D, G)** and the Generator is trying to minimize the Discriminator's reward or in other words, maximize its loss. It can be mathematically described by the formula below:

$$\min_G \max_D V(D,G)$$

$$V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z))]$$

Loss function for a GAN Model

where,

G = Generator
D = Discriminator
Pdata(x) = distribution of real data
P(z) = distribution of generator
x = sample from Pdata(x)
z = sample from P(z)
D(x) = Discriminator network
G(z) = Generator network

Advantages of Generative Adversarial Networks (GANs):

1. Synthetic data generation: GANs can generate new, synthetic data that resembles some known data distribution, which can be useful for data

augmentation, anomaly detection, or creative applications.
2. High-quality results: GANs can produce high-quality, photorealistic results in image synthesis, video synthesis, music synthesis, and other tasks.
3. Unsupervised learning: GANs can be trained without labeled data, making them suitable for unsupervised learning tasks, where labeled data is scarce or difficult to obtain.
4. Versatility: GANs can be applied to a wide range of tasks, including image synthesis, text-to-image synthesis, image-to-image translation, anomaly detection, data augmentation, and others.

Disadvantages of Generative Adversarial Networks (GANs):

1. Training Instability: GANs can be difficult to train, with the risk of instability, mode collapse, or failure to converge.
2. Computational Cost: GANs can require a lot of computational resources and can be slow to train, especially for high-resolution images or large datasets.
3. Overfitting: GANs can overfit the training data, producing synthetic data that is too similar to the training data and lacking diversity.
4. Bias and Fairness: GANs can reflect the biases and unfairness present in the training data, leading to discriminatory or biased synthetic data.
5. Interpretability and Accountability: GANs can be opaque and difficult to interpret or explain, making it challenging to ensure accountability, transparency, or fairness in their applications.

5. Radial Basis Function Networks (RBFNs)

Radial Basis Function (RBF) Networks are a particular type of Artificial Neural Network used for function approximation problems. RBF Networks differ from other neural networks in their three-layer architecture, universal approximation, and faster learning speed. In this article, we'll describe Radial Basis Functions Neural Network, its working, architecture, and use as a non-linear classifier.

Radial Basis Functions: What Are They?

An input layer, a hidden layer, and an output layer make up the three layers of feed-forward neural networks that belong to the particular class of radial basis functions. This differs significantly from the majority of neural network architectures, which have several layers and generate nonlinearity by repeatedly using nonlinear activation functions. The hidden layer is where computation takes place after receiving input data from the input layer. The Radial Basis Functions Neural Network's hidden layer is its most potent and distinctive feature compared to other neural networks. Prediction tasks like classification or regression are reserved for the output layer.

How Do RBF Networks Function?

Though their implementations are very different, RBF Neural Networks and K-Nearest Neighbour (k-NN) models are conceptually similar. Radial Basis Functions' core tenet is that items with similar values of predictor variables will generally have similar predicted target values. One or more RBF neurons may be placed in the area that the predictor variables describe. The number of predictor variables present is reflected in the space's numerous dimensions. We determine the Euclidean distance between each neuron's centre and the assessed point. To determine each neuron's weight (impact), a Radial Basis Function (RBF), also known as a kernel function, is applied to the distance. The name of the Radial Basis Function comes from the radius distance, which is the argument to the function. Weight = RBF[distance)The greater the distance of a neuron from the point being evaluated, the less influence (weight) it has.

Advantages of RBFN

1. Easy Design
2. Good Generalization
3. Faster Training
4. Only one hidden layer
5. A straightforward interpretation of the meaning or function of each node in the hidden layer.

What is the difference between RBF and MLP?

Multilayer perceptron (MLP) and Radial Basis Function (RBF) are popular neural network architectures called feed-forward networks. The main differences between RBF and MLP are:
MLP consists of one or several hidden layers, while RBF consists of just one hidden layer.
RBF network has a faster learning speed compared to MLP. In MLP, training is usually done through backpropagation for every layer. But in RBF, training can be done either through backpropagation or RBF network hybrid learning.

6. Multi-Layer Perceptron (MLP)

Multi-layer perception is also known as MLP. It is fully connected dense layers, which transform any input dimension to the desired dimension. A multi-layer perception is a neural network that has multiple layers. To create a neural network we combine neurons together so that the outputs of some neurons are inputs of other neurons.

Neural Networks

A multi-layer perceptron has one input layer and for each input, there is one neuron (or node), it has one output layer with a single node for each output and it can have any number of hidden layers and each hidden layer can have any number of nodes. A schematic diagram of a Multi-Layer Perceptron (MLP) is depicted below.

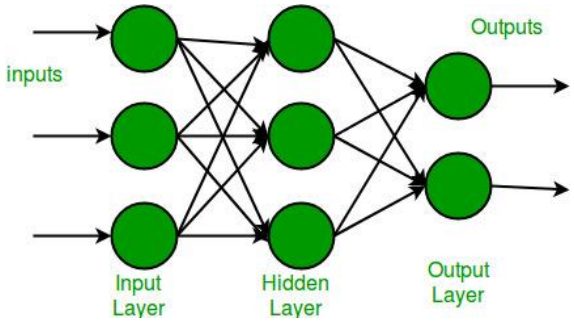

Fig. 9 MLP

In the multi-layer perceptron diagram above, we can see that there are three inputs and thus three input nodes and the hidden layer has three nodes. The output layer gives two outputs, therefore there are two output nodes. The nodes in the input layer take input and forward it for further process, in the diagram above the nodes in the input layer forwards their output to each of the three nodes in the hidden layer, and in the same way, the hidden layer processes the information and passes it to the output layer.

Every node in the multi-layer perception uses a sigmoid activation function. The sigmoid activation function takes real values as input and converts them to numbers between 0 and 1 using the sigmoid formula.

$$(x) = 1/(1 + exp(-x))$$

Some important points to note:

1. The Sequential model allows us to create models layer-by-layer as we need in a multi-layer perceptron and is limited to single-input, single-output stacks of layers.
2. Flatten flattens the input provided without affecting the batch size. For example, If inputs are shaped (batch_size,) without a feature axis, then flattening adds an extra channel dimension and the output shape is (batch_size, 1).
3. Activation is for using the sigmoid activation function.

4. The first two Dense layers are used to make a fully connected model and are the hidden layers.
5. The last Dense layer is the output layer which contains 10 neurons that decide which category the image belongs to.

7. Self-Organizing Maps – Kohonen Maps

Self-Organizing Map (or Kohonen Map or SOM) is a type of Artificial Neural Network which is also inspired by biological models of neural systems from the 1970s. It follows an unsupervised learning approach and trained its network through a competitive learning algorithm. SOM is used for clustering and mapping (or dimensionality reduction) techniques to map multidimensional data onto lower-dimensional which allows people to reduce complex problems for easy interpretation. SOM has two layers, one is the Input layer and the other one is the Output layer.

The architecture of the Self Organizing Map with two clusters and n input features of any sample is given below:

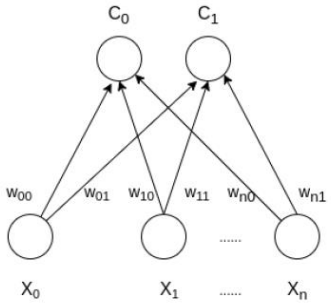

Fig. 10 SOM

How do SOM works?

Let's say an input data of size (m, n) where m is the number of training examples and n is the number of features in each example. First, it initializes the weights of size (n, C) where C is the number of clusters. Then iterating over the input data, for each training example, it updates the winning vector (weight vector with the shortest distance (e.g Euclidean distance) from training example).

Weight updation rule is given by :

$w_{ij} = w_{ij}(old)+alpha(t)*(x_i^k-w_{ij}(old))$

where,

alpha is a learning rate at time t,

j denotes the winning vector,

i denotes the i^{th} feature of training example and k denotes the k^{th} training example from the input data.

After training the SOM network, trained weights are used for clustering new examples. A new example falls in the cluster of winning vectors.

Algorithm

Training:

Step 1: Initialize the weights w_{ij} random value may be assumed. Initialize the learning rate α.

Step 2: Calculate squared Euclidean distance.

$D(j) = \Sigma (wij - xi)^2$ where i=1 to n and j=1 to m

Step 3: Find index J, when D(j) is minimum that will be considered as winning index.

Step 4: For each j within a specific neighborhood of j and for all i, calculate the new weight.

$wij(new)=wij(old) + \alpha[xi - wij(old)]$

Step 5: Update the learning rule by using :

$\alpha(t+1) = 0.5 * t$

Step 6: Test the Stopping Condition.

8. Deep Belief Networks (DBNs)

A deep belief network (DBN) is a type of deep neural network that consists of layers of latent variables (also known as "hidden units") with connections between the levels but none between the units within each layer. It is a generative graphical model.

Unsupervised training on a set of instances enables a DBN to develop the ability to probabilistically recreate

its inputs. After that, the layers serve as feature detectors. A DBN can be further taught under supervision to perform categorization after this learning phase.

DBNs can be thought of as a combination of straightforward, unsupervised networks like restricted Boltzmann machines (RBMs) or autoencoders, where the hidden layer of one sub-network acts as the visible layer for the following one.

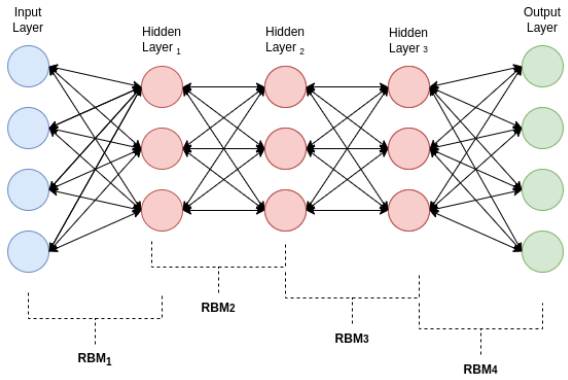

Fig. 11 DBNs

9. Restricted Boltzmann Machine (RBM)

A restricted Boltzmann machine (RBM) with fully connected visible and hidden units. Note there are no hidden-hidden or visible-visible connections.

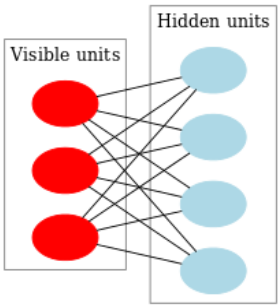

Fig. 12 RBMs

An RBM is an undirected, generative energy-based model with connections between but not within layers, a hidden input layer, and a "visible" input layer. Contrastive divergence is applied to each sub-network in turn, starting from the "lowest" pair of layers (the lowest visible layer being a training set), in this composition, which results in a quick, layer-by-layer unsupervised training approach.

10. Auto-Encoders

An artificial neural network called an autoencoder is utilised for unsupervised learning to develop effective codings for unlabeled input. An autoencoder learns two functions: a decoding function that reconstructs the input data from the encoded representation, and an encoding function that modifies the input data. For dimensionality reduction, typically, the autoencoder learns an effective representation (encoding) for a set of data.

There are variations that try to make the learnt representations take on beneficial features. Examples include variational autoencoders, which have applications as generative models, and regularised autoencoders (Sparse, Denoising, and Contractive), which are efficient in learning representations for subsequent classification tasks, and generative models that use variational autoencoders. Autoencoders are used to solve a variety of issues, such as facial identification, feature detection, anomaly detection, and word meaning acquisition. Autoencoders are generative models as well, and they have the ability to produce new data at random that is comparable to the training data.

III. LITERATURE REVIEW

(i) Hamade`ne et al. proposed a method based on both the contourlet transform and the co-occurrence matrix. First, for computing contour segment directions of the handwritten signature, the contourlet transform was applied. Then, for computing the direction number, the co-occurrence matrix was applied. Tests were applied on the CEDAR dataset by using a support vector machines (SVM) classifier. The outcomes

showed an AER of 0.07 for the writer -dependent approach and 0.18 for the writer-independent approach [16].

ii) Nemmour and Chibani investigated their applicability for handwritten signature verification. Ridgelet transform and grid features were used to extract important characteristics. Performance evaluation was applied to the CEDAR dataset relative to SVM classifiers. The results showed that the EER of the proposed system was 4.18 [18].

(iii) Kamihira et al. proposed a signature verification technique called "combined segmentation-verification" based on both offline and online features. Three different off-line feature vectors were extracted from the images of the Japanese signature (full name) and the images of the Japanese signature (first name and last name). Then, for each off-line feature vector, the Mahalanobis distance was computed for verifying the signature. The approach of online features uses a dynamic programming matching method for signature time-series data. The last decision of verification was carried out using an SVM classifier based on the Mahalanobis metric and dynamic programming matching [19].

(iv) Griechisch et al. proposed an online signature verification technique that utilized simple statistical tests and time constraints. They tested the x, y coordinates, pressure, and velocity features in a separate way and combined them. System performance was estimated based on the Dutch dataset [20].

(v) Fayyaz et al. presented a system dependent on learning the features using an autoencoder that tries to learn signature features. These features were used to show users' signatures. Then, one class classifier has been utilized for classifying users' signatures. The proposed verification process was evaluated on the SVC2004 signature database. The experimental results showed a reduction in errors and an enhancement in accuracy [21].

(vi) Radhika and Gopika focused on combining online and off-line handwritten signature features to verify them. A webcam was used to collect online data, and digital signature images were used to collect off-line data. First, online and offline data went through preprocessing stages. Second, both features were extracted in which features based on pen tip tracking were utilized in the case of online and gradient and projection based features were utilized in the case of offline approach. Finally, signatures were verified using both techniques separately, and their outputs were merged and inputted to the SVM classifier [22].

(vii) Lech and Czyzewski presented a signature verification system that uses both static features and time-domain functions of signals acquired by a tablet. The proposed approach fundamentally depends on Dynamic Time Warping (DTW) merged with some features from signature images acquired by a Wacom tablet that provides pressure information. Experimental results of this approach indicated a 0.82 average decision [23].

TABLE I. Signature Verification Methods [2]

Sr. N0	Ref. No, Author and Year	Work Done By Author	Findings by Author	Conclusion
1	Kesana Mohana Lakshmi, Tummala Ranga Babu "An Efficient Algorithm For	Implemented an efficient framework for off	Extensive simulation analysis disclosed that the proposed	Performing texture features extraction by

	Hand Written Signature Recognition Using Transform Based Approach With Image Statistics" 2020	line signature recognition system by utilizing NSCT framework.	methodology achieved superior performance over the conventional retrieval system by achieving higher mAP and mAR.	computing spatial dependence matrix and utilized NSCT algorithm for multi-scale decomposition and directionality.
2	A. B. Jagtap and R. S. Hegadi "Offline Handwritten Signature Recognition Based on Upper and Lower Envelope" 2017	Both the envelopes are fused by performing union operation and their covariance is computed.	The feature set consists of features such as large and small Eigen values computed from upper envelope and lower envelope and its union values.	Using Eigen Values" These features set are coupled with support vector machine classifier that lead to 98.5% of accuracy.
3	D. Morocho, A. Morales, J.Fierrez, and R. VeraRodriguez "Towards human assisted signature recognition: Improving biometric systems through attribute based recognition" 2020	Work explores human-assisted signature recognition including a baseline performance of human recognition of signatures and the analysis of manual attributebased signature recognition.	Present a crowdsourcing experiment to establish the human baseline performance for signature recognition tasks and a novel attribute based semiautomatic signature verification system inspired in FDE analysis.	FDE analysis give better understanding of output.
4	R. Sa-Ardship and K. Woraratpanya "Offline Handwritten Signature Recognition Using PolarScale Normalization" 2016	They has reviewed offline signature verification schemes in their paper. They have considered the Artificial Neural Network Technique.	The adaptive variance reduction algorithm is proposed to retain.	These problems can be considered for improving the system.

5	Vinayak Jadhav, Nikhil kadam, Paresh Keluskar, Ayyaj Khan "Artificial Neural Network Based Signature Verification" 2016	Paper presents a method of offline signature verification using artificial neural network approach.	For implementation of above this paper uses Feed Forward Neural Network (FFNN) using Error Back propagation algorithm for recognition and verification of signatures of individuals.	For verification of signatures some novel features needs to be extracted. The extracted features are used to train a neural network using error back propagation training algorithm.

The approaches used by different researchers differ in the type of features extracted, the training method, and the classification and verification model used.

Hidden Markov Models

Approach Hidden Markov Model (HMM) is one of the most widely used models for sequence analysis in signature verification. A handwritten signature is a sequence of vectors of values related to each point of signature in its trajectory. Therefore, a well-chosen set of feature vectors for HMM could lead to the design of an efficient signature verification system. These Models are stochastic models which have the capacity to absorb the variability between patterns and their similarities. In HMM stochastic matching (model and the signature) is involved. This matching is done by steps of probability distribution of features involved in the signatures or the probability of how the original signature is calculated. If the results show a higher probability than the test signatures probability, then the signatures is by the original person, otherwise the signatures are rejected. A HMM is used to model each writer signature. The method achieves an AER of 18.4% for a set of 440 genuine signatures from 32 writers with 132 skilled forgeries [4].

Neural Networks Approach

The main reasons for the widespread usage of neural networks (NNs) in pattern recognition are their power and ease of use. A simple approach is to firstly extract a feature set representing the signature (details like length, height, duration, etc.), with several samples from different signers. The second step is for the NN to learn the relationship between a signature and its class (either "genuine" or "forgery"). Once this relationship has been learned, the network can be presented with test signatures that can be classified as belonging to a particular signer. NNs therefore are highly suited to modelling global aspects of handwritten signatures. The proposed system uses structure features from the signatures contour, modified direction feature and additional features like surface area, length skew and centroid feature in which a signature is divided into two halves and for each half a position of the centre of gravity is calculated in reference to the horizontal axis. For classification and verification two approaches are compared the Resilient Back propagation (RBP) neural network and Radial Basic Function(RBF) using a database of 2106 signatures containing 936 genuine and 1170 forgeries. These two classifiers register 91.21% and 88% true verification respectively [4].

Template matching approach

Fang proposed two methods for the detection of skilled forgeries using template matching. One method is based on the optimal matching of the one-dimensional projection profiles of the signature patterns and the other is based on the elastic matching of the strokes in the two-dimensional signature patterns. Given a test signature to be verified, the positional variations are compared with the statistics of the training set, and a decision based on a distance measure is made. Both binary and grey-level signature images are tested. The average verification error rate of 18.1% was achieved when the local peaks of the vertical projection profiles of grey-level signature images were used for matching and with the full estimated covariance matrix incorporated [4].

Statistical approach

Using statistical knowledge, the relation, deviation, etc. between two or more data items can easily be found out. To find out the relation between some set of data items we generally follow the concept of Correlation Coefficients. In general statistical usage refers to the departure of two variables from independence. To verify an entered signature with the help of an average signature, which is obtained from the set of, previously collected signatures, this approach follows the concept of correlation to find out the amount of divergence in between them. In this approach, various features are extracted which include global features like image gradient, statistical features derived from the distribution of pixels of a signature and geometric and topographical descriptors like local correspondence to trace of the signature. The classification involves obtaining variations between the signatures of the same writer and obtaining a distribution in distance space. For any questioned signature the method obtains a distribution that is compared with the available known and a probability of similarity is obtained using a statistical Kolmorogorv-Smirnov test. Using only 4 genuine samples for learning, the method achieves 84% accuracy which can be improved to 89% when the genuine signature sample size is increased. This method does not use the set of forgery signatures in the training/learning [4].

Support Vector Machine

Support Vector Machines (SVMs) are machine learning algorithms that use a high dimensional feature space and estimate differences between classes of given data to generalize unseen data. The system uses global, directional and grid features of the signature and SVM for classification and verification. The database of 1320 signatures is used from 70 writers. 40 writers are used for training with each signing 8 signatures thus a total of 320 signatures for training. For initial testing, the approach uses 8 original signatures and 8 forgeries and achieves FRR 2% and FAR 11% [4].

IV. METHODOLOGY

The design of a system is divided into two stages:

A) Training stage
B) Testing stage

A. Training Stage consist of Four Major Steps

1. Retrieval of a signature image from a database
2. Image pre-processing
3. Feature extraction
4. Neural network training [3]

B. Testing Stage consists of Five Major Steps

1. Retrieval of a signature to be tested from a database
2. Image pre-processing
3. Feature extraction
4. Application of extracted features to a trained neural Network

5. Checking output generated from a neural network [3]

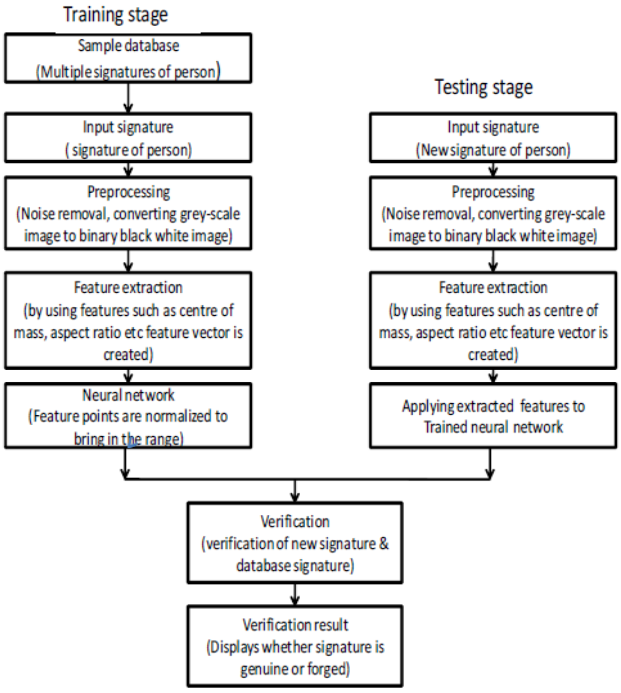

Fig. 6: Flow Chart of the system [3]

STAGES OF SIGNATURE VERIFICATION SYSTEM

In general, the offline and online signature verification processes consist of the following steps, as shown in Figure 7:

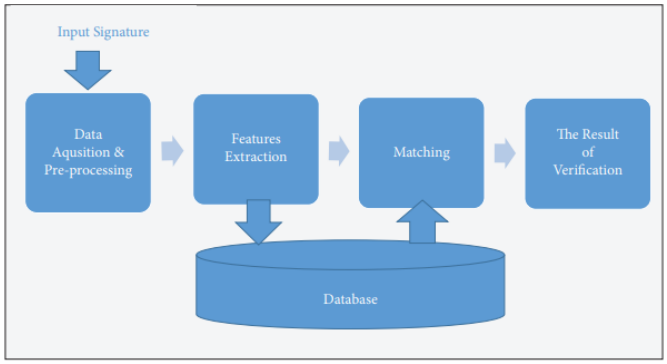

Fig 7: Signature verification system [7].

(1) Data acquisition: It is the first step in signature verification and is considered very important. In offline signature verification systems, data can be collected by utilizing off-line acquisition devices, for example, a camera or optical scanner, to scan the signature image to convert it into a digital image. In the online category, the data can be obtained by utilizing many digitizing devices, for example, tablets, electronic pens, and personal digital assistants (PDAs), as shown in Figure 8. However, the researcher can evaluate the performance of the system by using datasets that are publicly available on the Internet.

(2) Data preprocessing: preprocessing of datasets is the operation of improving the signature data after reading it. In both online and off-line verification systems, it is considered a very significant stage. In image preprocessing, various operations are applied to signature images; for example, color image to gray image conversion, noise removal, thresholding, morphological operations, cropping, binarization, and signature size normalization [25].

(3) Feature extraction: Some features of the signature are extracted in this step. These extracted features are the inputs to the training and recognition stages. The features can be categorized into global, mask, and grid features. Global features give wavelet coefficients and Fourier coefficients. Mask features give information about the signature lines' directions. Grid features give information about the overall appearance of a signature. The selection of feature sets in signature verification systems is a complicated task due to the fact that the user features must be appropriate for the application [26].

The main techniques of feature extraction for signature are given as follows: (a) Local and global feature techniques: Global features can be computed from the whole signature, while local features can be computed from a specific signature region. (b) Functional techniques: In these techniques, the online signature features can be considered as temporary sequences which include information about signature time changes. (c) Combined Techniques: These techniques are also called hybrid methods, and they depend on

merging various techniques from the previous techniques [25, 26].

(4) Classification: Classification is a method for determining the validity of a query signature. After the feature extraction stage, features are matched with those already stored in the database. Features are classified as genuine if they are matched, otherwise forged [27]. Support vector machines, template matching, hidden Markov models, and neural networks or deep learning are the most widely used classification methods [28, 29].

V. CONCLUSION

After thorough research and analysis, we have come to the following conclusions regarding deep learning techniques for signature verification. These methods have shown great promise in accurately identifying genuine signatures while minimizing false positives, particularly when using convolutional neural networks (CNNs) and long short-term memory (LSTM) networks.

Furthermore, the combination of these networks with advanced feature extraction methods has led to significant improvements in model performance and robustness against various forms of forgery. As these techniques continue to evolve, it is evident that they will play a crucial role in ensuring the security and integrity of sensitive documents and transactions moving forward.

VI. REFERENCES

[1]. Ruben Tolosana , Ruben Vera-Rodriguez , Julian Fierrez, and Javier Ortega-Garcia," DeepSign: Deep On-Line Signature Verification", IEEE Transactions On Biometrics, Behavior, And Identity Science, Vol. 3, No. 2, April 2021

[2]. Nakshita Pramod Kinhikar, Dr.K.N. Kasat, "Offline Signature Verification using Python", IJCRT 2022

[3]. Pallavi V. Hatkar, Zareen J Tamboli, "Image Processing for Signature Verification", International Journal of Innovative Research in Computer Science & Technology (IJIRCST) May- 2015

[4]. B.Akhila, G.Nikhila , A Lakshmi , G.Jahnavi, Mrs. J.Himabindhu, "Signature Verification Using Image Processing And Neural Networks", IJCRT 2021

[5]. Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, Javier Ortega-Garcia, "Exploring Recurrent Neural Networks for On-Line Handwritten Signature Biometrics", IEEE Feb-2018

[6]. Seymanur Akt, Hazım Kemal Ekenel, "Pre-processing of Signature Images for Detection and Verification", IEEE 2020

[7]. Zainab Hashim, Hanaa M. Ahmed, Ahmed Hussein Alkhayyat, "A Comparative Study among Handwritten Signature Verification Methods Using Machine Learning Techniques", Hindawi Scientific Programming Volume 2022

[8]. Huan Li , Student Member, IEEE, Ping Wei , Member, IEEE, and Ping Hu, "AVN: An Adversarial Variation Network Model for Handwritten Signature Verification", IEEE Transactions On Multimedia, VOL. 24, 2022

[9]. Jameel Malik, Ahmed Elhayek, Suparna Guha, Sheraz Ahmed, Amna Gillani, Didier Stricker, "DeepAirSig: End-to-End Deep Learning Based In-Air Signature Verification", IEEE 2020

[10]. Songxuan Lai, Lianwen Jin, Weixin Yang, "Online Signature Verification using Recurrent Neural Network and Length-normalized Path Signature Descriptor", 14th IAPR International Conference on Document Analysis and Recognition 2017

[11]. Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez and Javier Ortega-Garcia, "Biometric Signature Verification Using Recurrent Neural Networks", 14th IAPR International Conference on Document Analysis and Recognition 2017

[12]. Ruben Vera-Rodriguez, Ruben Tolosana, Miguel Caruana, Gustavo Manzano, Carlos Gonzalez-Garcia, Julian Fierrez, and Javier Ortega-Garcia, "DeepSignCX: Signature Complexity Detection using Recurrent Neural Networks", (ICDAR) 2019

[13]. Ziyun Zeng, "Multi-scale Attention-based Individual Character Network for Handwritten Signature Verification," 2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA), May 2022.

[14]. H. M. Ahmed and R. T. Rasheed, "A raspberry PI real-time identification system on face recognition," in Proceedings of the 2020 1st. Information Technology To Enhance e-learning and Other Application (IT-ELA, pp. 89–93, Baghdad, Iraq, July 2020.

[15]. Suriya Soisang, Suvit Poomrittigul, "Artificial Neural Network with Histogram Oriented Swerve Angle for Signature Verification," 2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC), 2022.

[16]. A. Hamadene, Y. Chibani, and H. Nemmour, "Off-line handwritten signature verification using contourlet transform and Co-occurrence matrix," in Proceedings of the 2012 International Conference on Frontiers in Handwriting Recognition, pp. 343–347, Bari, Italy, September 2012.

[17]. M. Sharif, M. A. Khan, M. Faisal, M. Yasmin, and S. L. Fernandes, "A framework for offline signature verification system: best features selection approach," Pattern Recognition Letters, vol. 139, pp. 50–59, 2020

[18]. H. Nemmour and Y. Chibani, "Off-line signature verification using artificial immune recognition system," in Proceedings of the 2013 International Conference on Electronics, Computer and Computation (ICECCO), pp. 164–167, Ankara, Turkey, November 2013.

[19]. Y. Kamihira, W. Ohyama, T. Wakabayashi, and F. Kimura, "Improvement of Japanese signature verification by combined segmentation verification approach," in Proceedings of the 2013 2nd IAPR Asian Conference on Pattern Recognition, pp. 501–505, Naha, Japan, November 2013.

[20]. E. Griechisch, M. I. Malik, and M. Liwicki, "Online signature verification based on Kolmogorov-Smirnov distribution distance," in Proceedings of the 2014 14th International Conference on Frontiers in Handwriting Recognition, pp. 738–742, Hersonissos, Greece, September 2014.

[21]. M. Fayyaz, M. H. Saffar, M. Sabokrou, M. Hoseini, and M. Fathy, "Online signature verification based on feature representation," in Proceedings of the 2015 #e International Symposium on Artificial Intelligence and Signal Processing (AISP), pp. 211–216, Mashhad, Iran, March 2015.

[22]. K. Radhika and S B. Gopika, "Online and offline signature verification: a combined approach," Procedia Computer Science, vol. 46, pp. 1593–1600, 2015.

[23]. M. Lech and A. Czyzewski, "A handwritten signature verification method employing a tablet," in Proceedings of the 2016 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), pp. 45–50, Poznan, Poland, September 2016.

[24]. V. Hindumathi, Jennifer Chalichemala, Endla Ameya, Vatadi Kavya Sri, Bhople Vaibhavi, "Offline Handwritten Signature Verification using Image Processing Techniques," 2023 IEEE 8th International Conference for Convergence in Technology (I2CT)

[25]. A. Beresneva, A. Epishkina, and D. Shingalova, "Handwritten signature attributes for its verification," in Proceedings of the 2018 IEEE

Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), pp. 1477–1480, Petersburg, Russia, January 2018.

[26]. H. M. Ahmad and S. R. Hameed, “Eye diseases classification using hierarchical MultiLabel artificial neural network,” in Proceedings of the 2020 1st. Information Technology To Enhance e-learning and Other Application (IT-ELA, pp. 93–98, Baghdad, Iraq, July 2020. Scientific Programming 15

[27]. M. R. Deore and S. M. Handore, “A survey on offline signature recognition and verification schemes,” in Proceedings of the 2015 International Conference on Industrial Instrumentation and Control (ICIC), pp. 165–169, Pune, India, May 2015.

[28]. G. Padmajadevi and K. S. Aprameya, “A review of handwritten signature verification systems and methodologies,” in Proceedings of the 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pp. 3896–3901, Chennai, India, March 2016.

[29]. T. Wilkin and O. S. Yin, “State of the art: signature verification system,” in Proceedings of the 2011 7th International Conference on Information Assurance and Security (IAS), pp. 110–115, Melacca, Malaysia, December 2011.

Cite this article as :

Deepali Narwade, Vaishali Kolhe, "A Survey : Deep Learning Approaches for Signature Verification", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 10 Issue 4, pp. 291-312, July-August 2023. Available at doi : https://doi.org/10.32628/IJSRSET23103216
Journal URL : https://ijsrset.com/IJSRSET23103216