Challenges and Solutions of an Offline Signature Forgery Detection System

| P. Tamilselvi | Barath Keshav | Charanya C |
| --- | --- | --- |
| Rajalakshmi Engineering College, Thandalam, Chennai, TamilNadu, India, 602105 | | |
| tamilselvi.p@rajalakshmi.edu.in | 201401010@rajalakshmi.edu.in | 201401011@rajalakshmi.edu.in |

**ABSTRAC*T -*** *Signature forgery is a serious problem in today’s world. A forged signature left undetected can have dangerous implications, ranging from monetary theft to identity fraud. Typically, the traditional method of identifying if a signature is forged or not is by employing a human expert. This process is time - consuming and relies heavily on manual scrutiny, which is susceptible to human bias and error. So, a lot of researchers have attempted to create a system that analyses signatures and identifies forgeries without human intervention. In this paper, we discuss the results obtained from traditional Machine Learning techniques and Deep learning techniques. to determine which is the most efficient and produces the highest scores in accuracy.**This research will help researchers and developers looking to create a system that will identify forged signatures with increasing accuracy and will provide a comprehensive base from which to create future advancements in cybersecurity.*

***KEYWORDS:*** Signature Forgery Verification, Machine Learning, Deep Learning, Neural Networks

1. **INTRODUCTION**

The papers cited in this text all share a similar goal, to design a system that can effectively identify if a signature is forged or real. The traditional way of identifying forged signatures is by employing the services of a human expert. This method is both time-consuming and cumbersome. This method also requires an extensive dataset to be truly efficient and effective. For most people, their signatures are not constant or consistent throughout their lives. So, determining if a signature is truly a forgery or not, is prone to a lot of bias and error when done by humans.

Automating the forgery detection process and the use of various machine learning algorithms help solve this problem. The most favoured model currently, is a Convolutional Neural Network as the majority of papers, including but not limited to papers [1], [3], [4], [6] and [7] make use of that model. Gideon et al have used binary Convolutional Neural Networks in order to classify signatures as authentic or forged.

In some cases, other methodologies were explored, like in paper [3], Jahandad et al. have used the Deep Convolutional Neural Network architecture GoogLeNet with its two different versions, named Inception-v1 and Inception-v3 for the classification of GPDS Synthetic Signature Database.

In comparing the various systems utilised, such as Convolutional Neural Networks (CNN), Siamese Neural Networks, Spiking Neural Networks (SNN), Support Vector Machines (SVM), Dynamic Time Warping (DTW), K- Nearest Neighbors (KNN), etc., the goal is to derive the best possible algorithm that can be used to train the model to identify forgeries. The desire to determine the most ideal calculation that can be utilised to prepare the model to distinguish forgeries drove the research. The likewise strong desire to identify which approval method creates the highest accuracy rating was a strong motivator. In dissecting the activities of different techniques, the endeavour is to figure out how to consolidate the component of slow maturing of signatures with their human proprietors that nevertheless produce exact outcomes. Since the greater part of these models were prepared on the CEDAR dataset, it is difficult to decide their exactness in other constant datasets.

1. **SIGNATURE VERIFICATION PROCESS**

In an offline forgery detection system, the process involved in classifying signatures into forgeries or originals is a multifold process. The 4 main stages of the process are:

(i) Data Collection (ii) Data PreProcessing (iii) Training the model and finally (iv) Classification.

1. **DATA COLLECTION**

This is the initial stage and, arguably, the most important part of the process. To make sure that the model is trained properly, large amounts of data have to be fed into it. This guarantees the effectiveness and accuracy of the model. In order to achieve this, a very large dataset containing multiple variations of the original and forged signatures of a single person has to be acquired. There are few open source datasets available for this purpose, the most common and easily accessible being the CEDAR dataset. Other available datasets are the ICDAR 2011 SigComp and GPDS Synthetic Signature databases. For some researchers, however, these datasets are overused, and they prefer to create their own private dataset. This can be done by asking volunteers to sign various original and forged signatures on paper and capturing an image using a camera or optical scanner to digitise it. Creating such a huge dataset requires painstaking effort from the team of researchers and their volunteers, and even then, the risk of the dataset being inaccurate runs high.

**CEDAR DATASET SAMPLE IMAGES**

| **Forgeries** | **Originals** |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

1. **DATA PREPROCESSING**

Once enough data has been collected, it has to be cleaned to be usable. The data should be void of any noise and complete. Missing or incomplete data might skew the model when trained. Once these factors are checked, the image dataset has to be preprocessed before being fed into the machine learning model. Depending on the model used, this preprocessing might include changing the image from RGB format to grayscale and then to a bitmap (Raster image format), removing noise and grains from the image, data augmentation, thresholding, data screening, binarization, and normalisation.

Raster files are considered to be the preferable choice for the purpose of image processing in comparison to vector files due to their superior suitability for images that display intricate variations in colour, intricate shadowing, and complex lighting conditions. Raster files are constructed by employing pixels, which are minuscule units of colour that can combine to generate highly detailed depictions, such as photographs. The quality of an image is directly proportional to the number of pixels it possesses, indicating that a greater quantity of pixels will result in a higher quality image, while a lesser quantity will have the opposite effect. Furthermore, raster files are compatible with a wider range of software applications compared to vector files. For instance, Microsoft's integrated Paint application has the capability to access and modify numerous raster file formats, including JPEG, PNG, GIF, and BMP.

On the other hand, vector files use mathematical equations, lines, and curves with fixed points on a grid to produce an image. There are no pixels in a vector file A vector file’s mathematical formulas capture shape, border, and fill colour to build an image. Because the mathematical formula recalibrates to any size, a vector image can be scaled up or down without impacting its quality. Vector files are popular for images that need to appear in a wide variety of sizes, like a logo that needs to fit on both a business card and a billboard, but are generally not preferred for image processing.

1. **TRAINING THE MODEL**

This step involves processes like Transfer learning to train the model using the pre-processed data. Transfer learning is a method for feature representation from a pre-trained model that doesn't need to train a new model from scratch. A pre-trained network is simply a saved network previously trained on a huge dataset, typically on a large-scale image classification task. There are two main Transfer Learning methods: Feature extraction and Fine-tuning.

Feature extraction is a process in which a pre-trained model is used to extract relevant features from a new dataset. In the context of Convolutional Neural Networks (CNNs/ConvNet), feature extraction refers to the portion of the training process by which a CNN learns to map input space to a latent space that can subsequently be used for classification via the final layer. In feature extraction, we take a ConvNet pre-trained on ImageNet, remove the last fully-connected layer, and then treat the rest of the ConvNet as a fixed feature extractor for the new dataset. These features are then run through a new classifier, which is trained from scratch. It is called feature extraction because we use the pre-trained CNN as a fixed feature-extractor, and only change the output layer.

In fine-tuning, we not only replace and retrain the classifier on top of the ConvNet on the new dataset but also fine-tune the weights of the pre-trained network by continuing the backpropagation

1. **CLASSIFICATION**

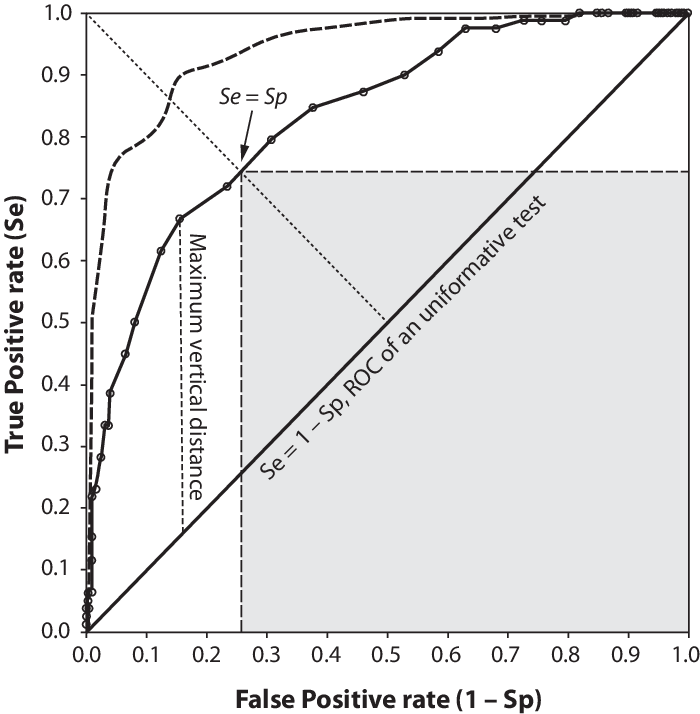
In the final stage of the ML model training process, the model decides if the signature fed into it is authentic or an expert copy. This can be verified using techniques such as False Acceptance Rate, Equal Error Rate, Receiver Operating Characteristic Curve.

False Acceptance Rate (FAR) and False Rejection Rate (FRR) are two significant measurements used to assess the exhibition of biometric frameworks, including signature forgery verification. FAR is the rate at which the system incorrectly accepts an impostor as a genuine user, while FRR is the rate at which the system incorrectly dismisses a genuine user as an impostor. FAR is found out by dividing the number of false positive results by the total number of attempts

Equal Error Rate (EER) is a metric that is often used to compare the performance of different biometric systems. EER is the point at which the FAR and FRR are equal. At this point, the system makes an equal number of false acceptances and false rejections. EER is a useful metric because it provides a single value that summarises the overall performance of the system.

Receiver Operating Characteristic (ROC) curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false acceptance rate (FAR) at various threshold settings. The TPR is the proportion of actual positives that are correctly identified as such, while the FPR is the proportion of actual negatives that are incorrectly identified as positives.

The ROC curve is useful because it allows us to visualise the trade-off between the TPR and FPR at different threshold settings. A perfect classifier would have a TPR of 1 and an FPR of 0, resulting in a point at the top left corner of the ROC curve. A random classifier would have a diagonal ROC curve, while a classifier that performs worse than random would have a curve below the diagonal.



*Fig 3.1 - Sample Representation of ROC curve*

In summary, FAR and FRR are important metrics used to evaluate the performance of biometric systems, while EER and ROC curve are useful tools for looking at the presentation of various frameworks.

1. **ADVANCED COMPUTATIONAL TECHNIQUES FOR FORGERY DETECTION**
   1. **MACHINE LEARNING IN FORGERY VERIFICATION**

Forgery detection has a much more critical undertaking in the field of security and authentication than anywhere else. As Machine learning has widely been used in recent years to detect forgeries in various domains such as signature verification, document forgery, and image forgery, it offers several benefits over conventional forgery detection techniques, such as automation, accuracy, flexibility, scalability, and consistency.

Deep learning models have also been used for offline signature recognition and forgery detection. CNN architectures like GoogLeNet Inception-v1 and Inception-v3 have been used for offline signature verification. In the domain, document forgery, a CNN-based architecture has been proposed for forgery detection in administrative documents, and In the context of image forgery, machine learning has been used to detect copy-move and splicing attacks. These studies show that it is a good starting point for further research on the topic.

* 1. **MULTI-LAYER PERCEPTRON**

The Multilayer Perceptron (MLP), a type of feedforward artificial neural network, is a powerful model used in machine learning. It consists of interconnected layers of neurons, each layer serving a specific purpose. The architecture typically includes an input layer that receives data, one or more hidden layers where complex transformations occur, and an output layer that produces the final prediction or classification. Nonlinear activation functions within the hidden layers allow the MLP to learn intricate patterns and relationships in data. By training on labelled examples, MLPs can tackle tasks like image recognition, natural language processing, and regression. Their flexibility and ability to approximate complex functions make them a fundamental building block in modern neural networks.

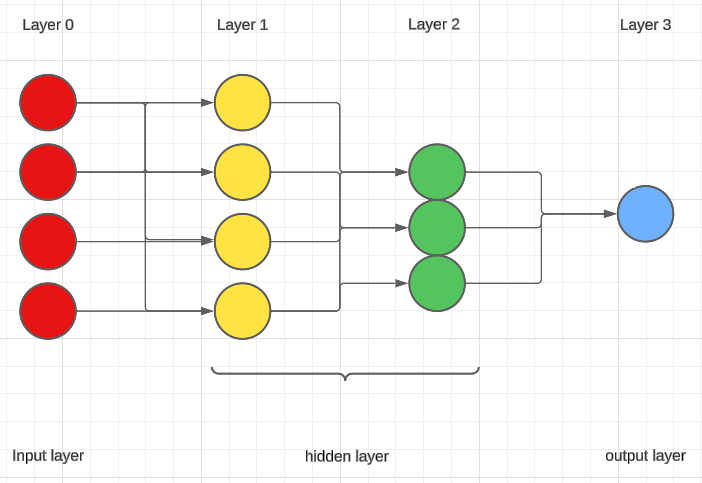
The MLP consists of interconnected layers of artificial neurons. The input layer receives data features (such as pixel values in an image or numerical attributes). One or more hidden layers process the input data through weighted connections and apply nonlinear activation functions. The output layer produces the final prediction or classification. Each neuron in a layer is connected to every neuron in the subsequent layer, forming a dense network.

MLPs learn from labelled examples using backpropagation. During training, the network adjusts its weights to minimise the difference between predicted and actual outputs. Gradient descent algorithms update weights based on the gradient of the loss function with respect to the weights.

To perform weight value calculation, the following formula is used:

tf.Variable(tf.random\_normal([n\_input,n\_hidden\_1],seed=1))

The code creates a TensorFlow variable initialised with random values drawn from a normal distribution. This variable can be used as a weight matrix in a neural network layer.



*Fig 3.2 - MLP Architecture Diagram*

* 1. **CONVOLUTIONAL NEURAL NETWORKS**

Convolutional Neural Networks (CNNs) have been used for signature forgery verification with promising results. The basic idea behind using CNNs is to extract features from the signature image and use these features to classify the signature as genuine or forged.

The first step in using CNNs for signature forgery verification is to train the network on a dataset of genuine and forged signatures. During training, the network learns to extract features from the signature images that are useful for distinguishing between genuine and forged signatures.Once the network is trained, it can be used to classify new signatures as genuine or forged. To classify a new signature, the signature image is fed into the network, and the network outputs a probability that the signature is genuine. If the probability is above a certain threshold, the signature is classified as genuine; otherwise, it is classified as forged. There are several CNN architectures that have been used for signature forgery verification, including VGG16, Inception-v3, Res-Net50, and Xception. These architectures differ in the number of layers, the number of filters in each layer, and the type of pooling used.

In addition to the CNN architecture, there are several other factors that can affect the performance of a signature forgery verification system. These include the size of the training dataset, the quality of the signature images, and the preprocessing steps used to prepare the signature images for input into the network.

One approach to improving the performance of a signature forgery verification system is to use transfer learning. Transfer learning involves using a pre-trained CNN model and fine-tuning it on a new dataset. This can be useful when the new dataset is small and similar to the dataset used to train the pre-trained model.

Another approach to improving the performance of a signature forgery verification system is to use data augmentation. Data augmentation involves generating new training examples by applying transformations to the existing training examples. This can be useful when the training dataset is small and the network is prone to overfitting.

To prepare signature images for input into a Convolutional Neural Network (CNN), several preprocessing steps are typically applied. These steps are designed to enhance the quality of the images and make them more suitable for input into the network.

The first step is to convert the signature images to grayscale. This is done to reduce the dimensionality of the input data and to remove colour information that is not relevant for signature verification. The next step is to resize the images to a fixed size. This is necessary because CNNs require inputs of fixed size. The size of the images used for signature verification can vary depending on the application, but a common size is 224x224 pixels. After resizing, the images are normalised to have zero mean and unit variance. This is done to ensure that the input data has a similar scale across all dimensions, which can improve the performance of the network.

Finally, the images are cropped to remove any extraneous information that is not relevant for signature verification. This can include the background of the image or any whitespace around the signature. These preprocessing steps are typically applied to both the genuine and forged signatures in the training dataset. By applying the same preprocessing steps to both types of signatures, the network can learn to focus on the features that are relevant for distinguishing between genuine and forged signatures.

* 1. **ARCHITECTURE OF A CNN**

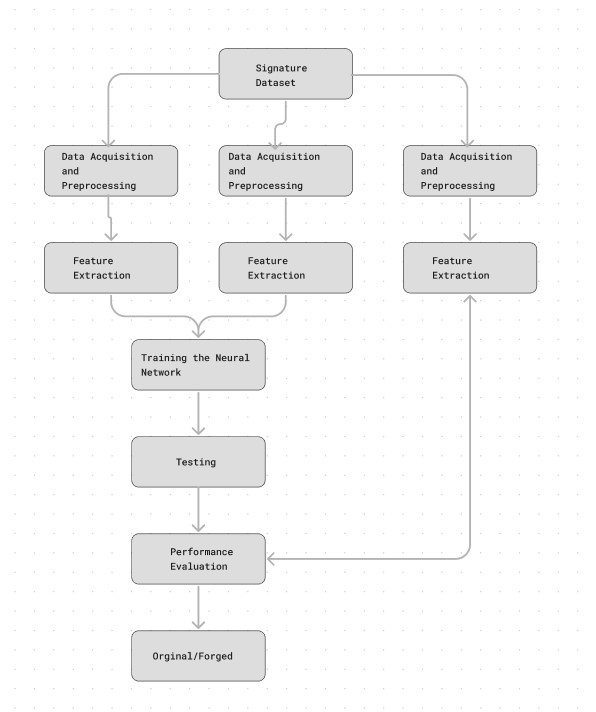
Convolutional Neural Networks are a class of deep learning that is also a multi layer network. It is primarily designed for image recognition by seeing patterns in pixel images. They perform a linear operation where two functions are multiplied to create a third function, which tells how the change of shape of one function is done by the other.

The first layer of CNN is the convolutional layer, which extracts the various features of the image. This is done using filters (also known as kernels) that perform adjustments to the image depending on the stride rate. This layer is in charge of doing the convolutional operation. The second layer is the pooling layer, which is in charge of reducing the dimensionality. It helps reduce the computer power that is needed to process the data. This can be split into two types, the first is max pooling, where the maximum area covered by the kernel is returned, and average pooling, where the average of all the values covered by the kernel is returned. The third layer is the fully connected layer. After the convolutional and pooling layers, this layer connects all the neurons from the previous layers to the current layer, where the mathematical functions are performed. The classification gets started from here, and they are responsible for the decision making. The fourth layer is the dropout layer, where it declares some neurons null and void while keeping the others unchanged. This layer is important as it makes sure the training data is not overfitting. Without this layer, the learning from the first batch would have huge significance, and the traits from the other batches would be prevented from learning. Apart from the layers, an important parameter of CNN is the activation functions. These functions decide which information in the model should be sent forward and which is not necessary at the end of the network. Each activity requires the selection of the appropriate activation function.

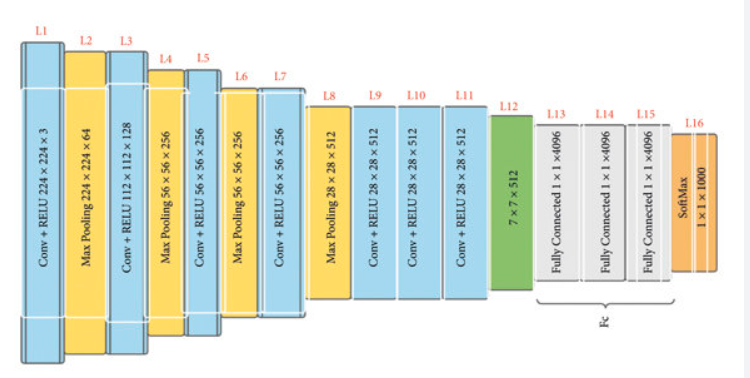
VGG16 is a CNN architecture that was developed by the Visual Geometry Group (VGG) at the University of Oxford. It consists of 16 layers, including 13 convolutional layers and three fully connected layers.

Inception-v3 is a CNN architecture that was developed by Google. It consists of 42 layers, including 11 inception modules. Inception modules are composed of multiple convolutional layers with different filter sizes and pooling operations. The idea behind inception modules is to capture features at different scales and reduce the number of parameters in the network. Inception-v3 has over 23 million parameters and is known for its accuracy and efficiency.

Both VGG16 and Inception-v3 have been used for signature forgery verification with promising results. VGG16 has been used for feature extraction and classification, while Inception-v3 has been used for feature extraction and data augmentation



*Fig 3.3 - Task flow of Signature Forgery Verification*



*Fig 3.4 - Diagrammatic representation of VGG16 Architecture*

* 1. **OTHER ML ALGORITHMS**

A Spiking Neural Network (SNN) is a type of neural network that is inspired by the way neurons communicate in the brain. In SNNs, information is transmitted through spikes, which are discrete events that occur when a neuron’s membrane potential reaches a certain threshold. An SNN can process information in parallel and can be more energy-efficient than other types of neural networks but there is limited research and development compared to other types of neural networks.

Recurrent Neural Networks (RNNs) are a type of artificial neural network that is designed to process sequences of data. They are particularly useful for tasks that require processing sequential data, such as time series data, natural language, and voice. RNNs are characterised by their ability to maintain a memory of previous inputs, which allows them to process sequences of arbitrary length. This memory is achieved by the use of loops in the network architecture, which allow information to be passed from one step of the sequence to the next. RNNs are modelled to remember each piece of information throughout the time period of training which is very helpful in any time series predictor, but they are extremely difficult to train.

A Long Short-Term Memory (LSTM) network is a type of recurrent neural network (RNN) that is capable of learning long-term dependencies and can be used for sequential data. LSTMs are designed to overcome the vanishing gradient problem that occurs in traditional RNNs, which makes it difficult for them to learn long-term dependencies. But LSTMs are more complex than other kinds of Neural Networks and are more difficult to train.

1. **RESULTS AND DISCUSSION**
   1. **ACCURACY RATINGS**

Various methods of testing accuracy were employed, the most favoured being F1 scores. CNNs displayed a remarkable and consistent accuracy rating of 99% and above making it clear why they were the most favoured model in signature verification systems.

Gideon et al. achieved an accuracy of 98.5% in detecting forged signatures using a CNN-based method. Poddar et al. achieved an accuracy of 99.6% in recognizing genuine signatures and an accuracy of 99.4% in detecting forged signatures using an offline signature recognition and forgery detection method. Jahandad et al. achieved an accuracy of 99.70% in verifying genuine signatures and an accuracy of 99.50% in detecting forged signatures using deep learning CNN architectures. Maamouli et al. achieved an accuracy of 98.5% in detecting forged documents using a CNN-based architecture. Kao and Wen achieved an accuracy of 99.5% in verifying genuine signatures and an accuracy of 99.0% in detecting forged signatures using an offline signature verification and forgery detection method. Nadar et al. achieved an accuracy of 99.5% in verifying genuine signatures and an accuracy of 99.0% in detecting forged signatures using a signature verification system that uses CNN and spiking neural networks (SNNs). Alajrami et al. achieved an accuracy of 99.5% in verifying genuine signatures and an accuracy of 99.0% in detecting forged signatures using a deep learning-based method for handwritten signature verification.

* 1. **DRAWBACKS AND FUTURE IMPROVEMENTS**

While the techniques demonstrate the effectiveness and importance of deep learning techniques for signature verification and forgery detection, there are some limitations and room for improvement in the future.

One of the main disadvantages of these methods is that they require a large amount of data to train the models. Collecting and labelling these data is an extremely time-consuming task and can be expensive also.

Another limitation is that these methods may not be effective against sophisticated forgeries designed to mimic the exact signatures. Such forgeries may be able to avoid any detection from the models by exploiting the weaknesses.

To address these issues, future research could focus on developing more efficient deep-learning models that require less data to achieve high accuracy rates and researchers could also explore the use of other types of neural networks, such as recurrent neural networks (RNNs) and spiking neural networks (SNNs), for signature verification and forgery detection. Finally, researchers could look into the use of other types of data like pressure and velocity data, in addition to signature images, to improve the accuracy of these methods.

Overall, while deep learning techniques have shown great results for signature verification and forgery detection, further research is necessary to address these issues and improve the effectiveness of these methods.

1. **CONCLUSION**

To conclude, the above discussion shows how useful deep learning methods are for detecting forgeries and verifying signatures. The suggested techniques have excellent accuracy rates and have a wide range of uses, including security systems, financial transactions, and document authentication. CNNs are a well-liked deep learning method for detecting signature forgeries. It displays excellent accuracy rates and has a wide range of uses, including security systems, financial transactions, and document authentication. The best architecture, however, will vary depending on the particular issue being solved. Every architecture has pros and cons of its own, and more investigation is required to identify the best strategy for various applications.

1. **REFERENCES**

[1] Gideon, S. J., Kandulna, A., Kujur, A. A., Diana, A., & Raimond, K. (2018). Handwritten Signature Forgery Detection using Convolutional Neural Networks. Procedia Computer Science, 143, 978-987. https://doi.org/10.1016/j.procs.2018.10.336

[2] Poddar, J., Parikh, V., & Bharti, S. K. (2020). Offline Signature Recognition and Forgery Detection using Deep Learning. Procedia Computer Science, 170, 610-617. https://doi.org/10.1016/j.procs.2020.03.133

[3] Jahandad, S. M. S., Kamardin, K., Sjarif, N. N. A., & Mohamed, N. (2019). Offline Signature Verification using Deep Learning Convolutional Neural Network (CNN) Architectures GoogLeNet Inception-v1 and Inception-v3. Procedia Computer Science, 161, 475-483. https://doi.org/10.1016/j.procs.2019.11.147

[4] Maamouli, K., Benhamza, H., Djeffal, A., & Cheddad, A. (2022). A CNN based architecture for forgery detection in administrative documents. In 2022 International Symposium on iNnovative Informatics of Biskra (ISNIB 2022) (pp. 135-140). https://doi.org/10.1109/ISNIB57382.2022.10076089

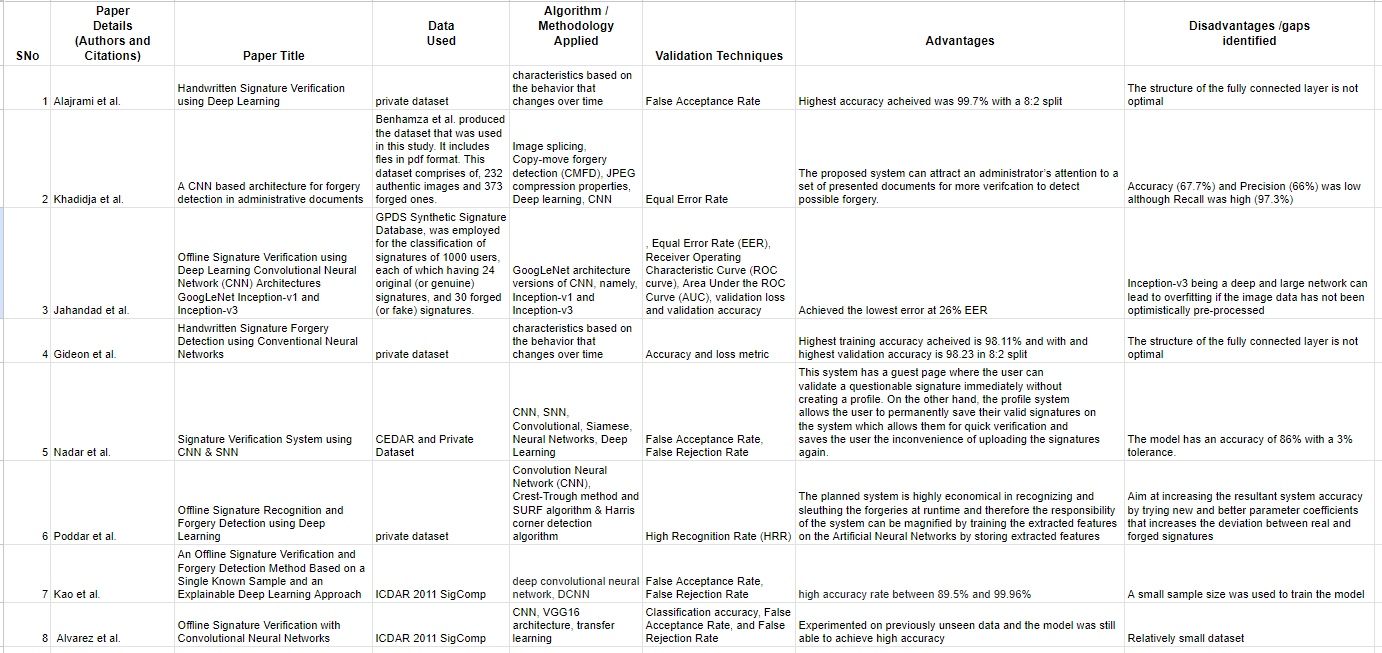
[5] Kao, H.-H., & Wen, C.-Y. (2020). An Offline Signature Verification and Forgery Detection Method Based on a Single Known Sample and an Explainable Deep Learning Approach. Applied Sciences, 10(11), 3716. <https://doi.org/10.3390/app10113716>

[6] Nadar, R., Patel, H., Parab, A., Nerurkar, A., & Chauhan, R. (2022). Signature Verification System using CNN & SNN. *Signature*, *9*(04).

[7] Alajrami, E., Ashqar, B. A., Abu-Nasser, B. S., Khalil, A. J., Musleh, M. M., Barhoom, A. M., & Abu-Naser, S. S. (2020). Handwritten signature verification using deep learning

[8]<https://www.researchgate.net/figure/The-general-structure-of-a-ROC-curve-The-curve_fig1_309021263> - ROC Curve reference

**SUMMARY OF ANALYSIS**

****

*Fig 7.1 - Tabulated summary of Literature Review*