Proceedings of the Fourth International Conference on Sentiment Analysis and Deep Learning (ICSADL-2025)
IEEE Xplore Part Number: CFP25UU5-ART; ISBN: 979-8-3315-2392-3

Online Signature Verification using Deep Learning

Kancharagunta Kishan Babu
Department of CSE(AIML&IOT)
VNR VJIET
Hyderabad-500090, Telangana, India
kishan.kancharagunta@gmail.com

Srikanth Lukka
Department of CSE
SRM University
Amaravati-522240, Andhra Pradesh India
srikanth.l@srmap.edu.in

Palliyana Shabarish
Department of CSE(AIML&IoT)
VNR VJIET
Hyderabad-500090, Telangana, India
palliyanashabarish004@gmail.com

Aviresh Laxman Sai
Department of CSE(AIML&IoT)
VNR VJIET
Hyderabad-500090, Telangana, India
laxmansai1752004@gmail.com

Bandi Sai Varshini Goud
Department of CSE(AIML&IoT)
VNR VJIET
Hyderabad-500090, Telangana, India
bandivarshini1204@gmail.com

Ganji Yeshwanth
Department of CSE(AIML&IoT)
VNR VJIET
Hyderabad-500090, Telangana, India
ganjiyeshwanth5@gmail.com

2025 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL) | 979-8-3315-2392-3/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICSADL65848.2025.10933213

***Abstract*—Online Signature Verification (OSV) systems are crucial for secure authentication in digital environments, where handwritten signatures are electronically captured and verified. These systems use machine learning and deep learning methods to analyze unique dynamic features of a signature, such as pen pressure, speed, and stroke order, in addition to its static shape. Recent approaches leverage techniques such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Generative Adversarial Networks (GANs) and hybrid models, to analyze the unique dynamic and static features of signatures. However, there exists a problem of designing real-time systems that are responsive, and have reliable performance using the mentioned deep learning techniques because these models have to be trained each time a new user is added to the database. The objective of our experiment is to build an OSV system that integrates a CNN-based Siamese Network and a webpage created using ReactJS to allow the user to upload their signature or store it in the database. The model extracts spatial features from the signatures and makes decisions based on the similarity of the uploaded signature with the original signature of the user. The system is designed to handle real and forged signatures through continuous model training. Using a combination of signature preprocessing techniques, feature extraction, and classification models, this system aims to ensure a robust, reliable, and secure method for identity verification in online transactions and sensitive digital applications.**

***Index Terms*—Online Signature Verification, Convolutional Neural Network, Siamese Network, ReactJS, feature extraction.**

I. Introduction

In today's world, most transactions and interactions happen electrically, highlighting the need for secure and reliable authentication methods. One popular authentication method is online signature verification, which uses the unique features of a person's signature to confirm their identity. Unlike passwords or PINs, which can easily be compromised, signatures provide a biometric solution that is much harder to copy. OSV holds immense potential across various applications, including financial transactions, access control and identity verification [1] [2] [3]. Some signature verification systems are used to validate documents [4]. Additionally, OSV systems enhance convenience and user experience over traditional methods. By using the natural signing behaviour of individuals, these systems remove the need to remember complicated passwords or carry physical tokens. This approach streamlines the authentication process while ensuring robust security. As remote work and online services become increasingly common, OSV enables secure authentication from any location.

Some OSV systems depend on capturing and analyzing the dynamic characteristics of a signature as it is being written [5] [6]. In contrast to static signatures that are gathered from printed photos or documents, online signatures are recorded on-site while the signing process is taking place with the help of digital input devices such as digitized pens, touch screens or other such devices. Even though it has much significance and capability, online signature verification faces several challenges that limit its widespread adoption and effectiveness. Variations in signature styles, environmental factors, and the characteristics of input devices can introduce noise and inconsistency in the data. It complicates the development of reliable and accurate verification systems. Online signatures are subjected to imitation by people to access sensitive information. In some cases, users may not have access to devices capable of capturing dynamic signature characteristics, limiting the robustness of OSV on some platforms.

To address these challenges, researchers are exploring novel methodologies and emerging technologies in OSV. The field of deep learning has produced many algorithms that are more resistant to noise and distortions compared to traditional methodologies. A signature is characterized by curves, strokes, speed, pressure, spatial organization of letters and many such features. These features are not properly utilized by traditional verification systems. The deep learning algorithms can capture complex features from images, identify subtle differences in patterns and automatically adjust to changes over time by continuously learning from new data. This makes the algorithms more appropriate for the task of signature verification compared to traditional algorithms. Opting to deep learning

technologies such as CNN [7] [8] have shown a substantial increase in the accuracy of the OSV systems. The advent of transformers [9] [10] has further increased the performance of such verification systems. Authentication systems that auto-adjust based on some risk factors, tend to provide a flexible solution to identity verification in real time. The current research in machine learning and deep learning [11] [12], has shown great potential in improving accuracy and resilience of authentication systems, contributing to the development of a more seamless and secure method of authentication.

In this paper, Section II describes the related works in the field of signature verification. Section III describes the system proposed in the paper. Section IV of the paper describes the dataset used for training and evaluating the model. Section V of the paper provides the metrics used to evaluate the system, along with the results. The last section concludes the paper by discussing the future works.

II. Related Works

Over the years, many deep learning technologies have been introduced to improve the performance of signature verification systems. The system proposed by Hafemann et al. [13] used deep CNNs that extracted features from a signature to classify it as real or forged. The model was trained on both real and fake signatures from users. The system proposed by Chandra Sekhar Vorugunti et al. [14] tried to improve the discriminatory power of the traditional CNN by using Depth-Wise-Separable CNN, which used a mix of depth-wise convolutions and point-wise convolutions over. This CNN was integrated with LSTM architecture for signature classification. However, for building any model specifically for signature classification, there is always a necessity for forged signatures of users to train the model. CNNs perform less efficiently if there is no sufficient dataset to train them. They may provide biased output if the dataset is not balanced.

In many cases, there is always a scarcity in the number of forged signatures which can affect the training accuracy of the model. To address the issue of the limited size of the dataset, Generative Adversarial Networks (GANs) were introduced. The GAN proposed by Vorugunti et al. [15] consists of a generator that generates new fake signatures and a discriminator that tries to classify those images. This helps train the discriminator to improve its classifying power as the quality of the generated signatures increases each time. Over time, different modifications for GANs were proposed such as the CycleGAN which is primarily used for image-to-image translation. In the context of signature verification, CycleGAN is used to clean the signatures that contain noise such as marks or stamps [4]. Using GANs can sometimes lead to overfitting and they might not capture subtle yet critical features of genuine forgeries created by humans.

Although CNNs are used to process image data, they are not so useful in processing sequential data. Signatures are text and come under the category of sequential data and to process sequential data, Recurrent Neural Networks (RNNs) are proposed. The feasibility of RNNs in signature verification was studied by Ruben Tolosana et al. [16]. LSTMs (Long Short Term Memory) are a type of RNN and came into the picture because they can retain the long-term dependencies that are required for accurate predictions while RNNs cannot. More recent research is based on integrating transformers with CNNs and RNNs to increase the prediction capabilities of the models [17]. The transformers are also used to process sequential data but unlike RNNs, transformers can carry forward the context of the previous data which can help in creating an optimum signature verification system.

III. Proposed System Architecture

A. Frontend and Backend

The webpage has been built using HTML, CSS, and JavaScript. The figure 1 shows the user interface of the system. The user interface consists of an option for authenticating the user by asking for a username and password. The backend has been built using Flask, a Python framework for lightweight applications. The database used for this project is MongoDB, due to its ability to store images using GridFS. MongoDB is a NoSQL database that is popular due to its flexible schema and horizontal scalability [18]. It stores data in the form of documents which are similar to JSON objects. The signatures are stored in documents inside the collections which are equivalent to records in the table in an SQL database [19], with the name of the collection being the same as the user's name. Each document in the collection contains the unique identifier of the signature of the respective user along with the username and document ID which is used to uniquely represent the document.

When a signature is submitted to the database, GridFS splits the image into multiple chunks of smaller size and stores them in separate documents in **fs.chunks** collection. The **fs.files** collection contains metadata of the signature such as file size, file type, and references to the chunks in fs.chunks. Signature_file_id in the user collection contains the unique identifier that GridFS uses to reference the stored signature image in the fs.files and fs.chunks collections. The webpage allows users to fill in the username and also upload the signature that needs to be verified. The user signature is retrieved from the database to compare with the uploaded signature and classify the uploaded image as either real or forgery.

B. CNN-based and Siamese Architecture

Convolutional Neural Networks (CNNs) are highly effective for image-related tasks, particularly for extracting spatial feature images, which are critical for signature analysis since signatures are complex visual patterns. The CNN-based Siamese Network comprises two Convolutional Networks that take two signatures as input and compute a similarity metric. Siamese Networks are specifically designed to solve tasks involving finding the similarity or difference between two inputs. A pair of signatures form the input to the two CNNs of the model that share the same network weights and architecture, and sequential convolution and pooling operations are done to

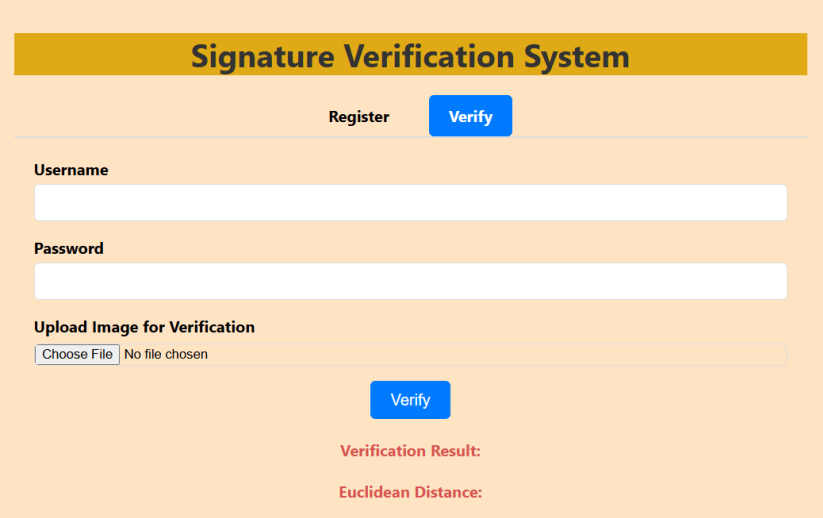

Fig. 1: **User Interface of the system: It provides the option for authentication and uploading signature. The register tab allows user to add signature to database while verify tab allows to verify the uploaded signature.**

extract high-level feature representations. These feature representations from both CNNs are joined using a contrastive loss function that calculates the distance between them using the Euclidean formula and learns the similarity metric. For similar pairs, the loss function persuades the model to minimize the distance between the data points, while for dissimilar pairs, it makes the model to maximize the distance between the points. The input signatures are grouped into pairs consisting of either genuine or genuine-forgery combinations. The architecture of a Siamese Network is illustrated in figure 2. The architecture of the model used for the experiment is inspired from [20]. It takes two input images of shape 224 × 224 × 3 and processes them through a series of convolutional and fully connected layers. It begins with a set of convolutional layers: the first convolution has 96 filters with a 7 × 7 kernel, followed by max pooling which reduces the spatial dimensions to 112 × 112, focusing on the most important features. This is followed by a second convolution with 256 filters with 5 x 5 kernels and another pooling layer reducing the dimensions to 56 × 56. After this, there are two more convolutional layers with 384 and 256 filters with 3 x 3 kernels, followed by max pooling, dropping the spatial dimensions to 27 × 27, with intermediate dropout layers for regularization. The output is then flattened into a vector of 186,624 elements. This vector is passed through two fully connected layers: the first has 1024 units with a dropout of 50%, and the second produces a final 128-dimensional vector. This network processes two input images independently to produce two feature embeddings, which can then be compared to determine their similarity.

IV. Dataset

The dataset used in the experimentation was obtained from kaggle[1]. All the data in the dataset is extracted from the ICDAR 2011 Signature Dataset. The dataset has a total of 2149 images with 69 fake classes and 69 genuine classes in the training set 21 fake classes, and 21 genuine classes in testing. There are a total of 23205 pairs of signatures in the training set and a total of 5747 pairs of signatures in the testing set. The pairs of signatures include both genuine-genuine as well as genuine-forged pair sets. The figure 3 displays the authentic signature of a sample user of the dataset while figure 4 shows the fake signature of the same user.

V. Experimentation and Results

Evaluation metrics are used to estimate performance and effectiveness of a model, especially in tasks like signature verification. The choice of metrics depends on the specific goal of the system. Overall it provides a summary of the performance of the model. Before discussing the different types of metrics, the fundamental terms that are used to derive them are described below:

1. **True Positive (TP)**: It is the case where a genuine signature is correctly identified as genuine.
2. **False Positive (FP)**: It is the case where a fake signature is incorrectly identified as genuine.
3. **True Negative (TN)**: It is the case where a fake signature is correctly identified as fake .
4. **False Negative (FN)**: It is the case where a genuine signature is wrongly identified as fake.

[1]https://www.kaggle.com/datasets/robinreni/signature-verification-dataset

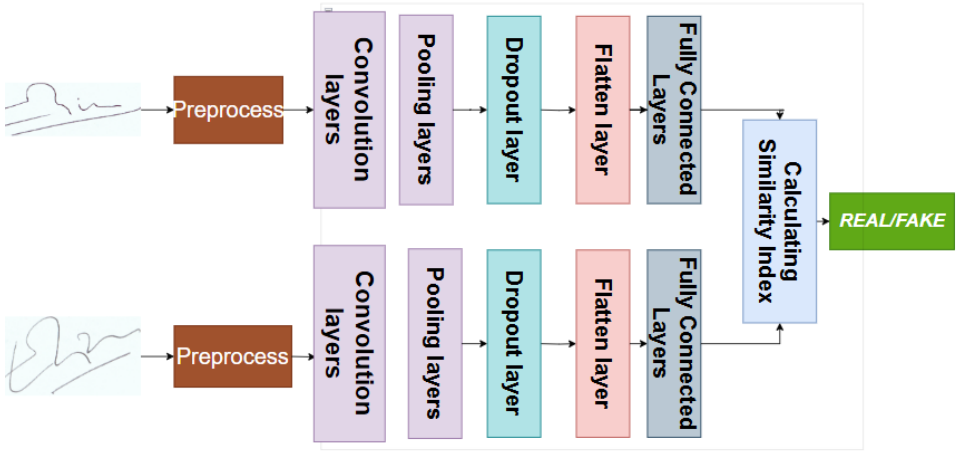

Fig. 2: **The general architecture of the model: the model implemented uses two signatures as inputs. These signatures are preprocessed before providing them to the model. The model outputs feature vectors that are compared to obtain similarity index.**

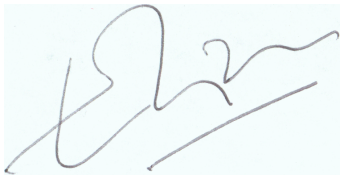

Fig. 3: **Real signature of sample user of the dataset**

Fig. 4: **Forged signature of the same sample user**

A. Precision

Precision in signature verification is the proportion of genuine signatures correctly identified out of all signatures classified as genuine(both correctly and incorrectly). A high precision indicates that model is good at minimizing the chances of accepting forged signatures as genuine.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

B. Recall

Recall, also known as **True Positive Rate (TPR)**, is the ratio of the total number of genuine signatures correctly identified to the total number of correct classifications. A high recall means the model is less likely to reject genuine signatures. It is important to make sure that genuine signatures are not wrongly classified.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

C. Accuracy

Accuracy is the proportion of all correct classifications. Accuracy provides an overall picture of how often the model makes correct predictions, considering both genuine and forged signatures. It is given as

$$CorrectClassifications = TP + TN \tag{3}$$

$$TotalClassifications = TP + TN + FN + FP \tag{4}$$

$$Accuracy = \frac{CorrectClassifications}{TotalClassifications} \tag{5}$$

Here *CorrectClassifications* denotes the number of signatures that the model classified correctly which is accepting genuine as genuine and forged as forged, and *TotalClassifications* denotes the size of the dataset that the model is evaluated on.

D. F1 Score

The F1 score evaluates a model's ability to classify each class accurately by balancing precision and recall. It is the harmonic mean of precision and recall.

$$F1Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{6}$$

E. Specificity

This metric represents the proportion of actual forged signatures that are incorrectly classified as genuine by the model. It is also called **False Positive Rate (FPR)**.

$$Specificity = \frac{FP}{FP + TN} \tag{7}$$

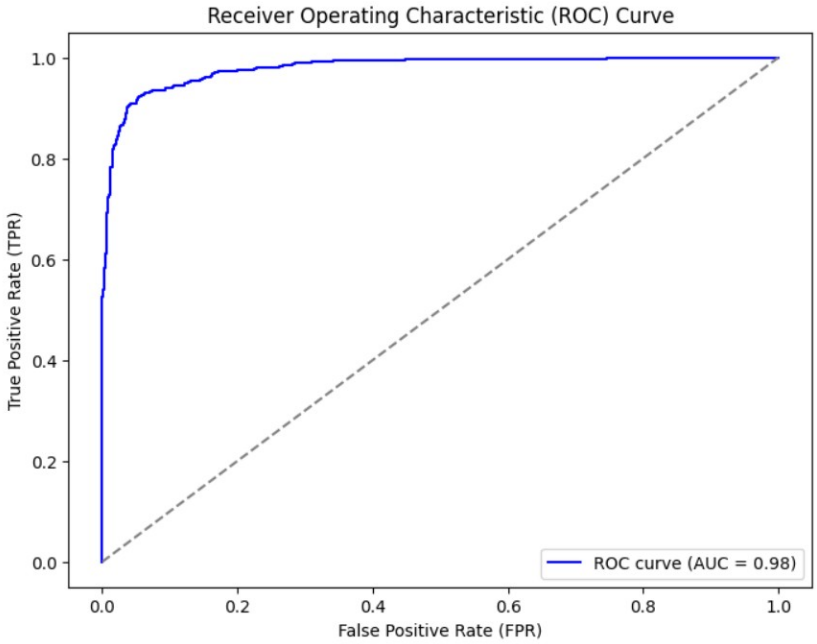

Fig. 5: **The ROC curve is plotted with FPR on x-axis and TPR on y-axis. It also displays AUC value which denotes general performance of the model. The AUC value for the model is 0.98**

F. Result and Analysis

The signatures of the dataset were resized to the required dimensions of (224,224,3) before providing them as input to the model. To evaluate the system, the model was first trained on a subpart of the training dataset that included around 10,000 signature pairs and then the model was made to predict on the testing dataset which includes 5747 pairs of signatures with the number of genuine-genuine pairs of signatures being 2772 and number of genuine-forged pairs of signatures being 2975. We have implemented our framework using the Keras library. The final output of the model is 2 vectors for the input pair. The Euclidean distance between vectors is compared with a threshold.

The ROC curve is a graph that shows the performance of a binary classifier across different thresholds. The ROC curve shown in the figure 5 is created by plotting TPR against FPR at various threshold levels. Each point on the curve specifies a threshold value. The ROC curve helps visualize the trade-off between sensitivity and specificity. The area under the ROC curve (AUC) provides a single metric to evaluate performance of the model. The AUC ranges from 0 to 1 and high AUC tells that the model is better at correctly classifying samples across all threshold levels. The AUC value for the model as shown in figure 5 is 0.98. The optimal threshold is that maximizes the TPR and minimizes the FPR. From the experiment, an optimal threshold of about 0.24 is obtained.

After evaluating the test dataset using the obtained threshold value, below Table I I representing the confusion matrix of the model is obtained. The following Table II II provides the results of the experimentation, providing the precision, recall, f1 score, and accuracy of the model.

TABLE I: **The confusion matrix shows the number of True Positives, True Negatives, False Positives and False Negatives obtained using the model.**

	Predicted:Real	**Predicted:Forged**
Actual:Real	2572	200
Actual:Forged	198	2777

TABLE II: **The table shows the different metrics that assess performance of the model.**

Metric	**Value**
Precision	92.85%
Recall	92.78%
F1 Score	92.81%
Accuracy	93.07%

For the signature classification task, this high precision of 92.85% indicates that the model correctly identifies genuine signatures while minimizing false positives. A recall value of 92.78% signifies that the model is effectively identifying genuine signatures, with fewer false negatives. The f1 Score is 0.9281 or 92.81%. A high f1 score in this case implies that the model has strong balance between correctly identifying genuine signatures and avoiding misclassifications. The model's overall accuracy of 93.07% shows a strong capability to correctly classify signatures in the majority of cases, making it reliable for practical applications.

VI. Conclusion

In this paper, we described a CNN-based Siamese network for an online signature verification system. The paper demonstrated the experimentation and the evaluation of the model on the available Kaggle dataset. The metrics used to evaluate the model are precision, recall, f1-score and overall accuracy of the model. The accuracy achieved is around 93%. The model's performance owes to its Siamese-based framework, which can perform exceptionally well, even though trained on less data. There is still scope for improvement. The model was trained only on a subpart of training dataset. So increasing the size of subpart might increase the performance of the model. The addition of more convolutional and pooling layers as well as increasing the number of epochs over the dataset might significantly increase the performance. Our future work is aimed at developing more efficient, reliable and generalizable model.

VII.

References

[1] M. C. Fairhust and S. Ng, "Management of access through biometric control: A case study based on automatic signature verification," *Universal Access in the Information Society*, vol. 1, pp. 31–39, 2001.

[2] E. J. Potter, *Customer authentication: The evolution of signature verification in financial institutions*. PhD thesis, Citeseer, 2002.

[3] J. Putz-Leszczyńska and M. Kudelski, "Hidden signature for dtw signature verification in authorizing payment transactions," *Journal of telecommunications and information technology*, no. 4, pp. 59–67, 2010.

[4] D. Engin, A. Kantarci, S. Arslan, and H. K. Ekenel, "Offline signature verification on real-world documents," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[5] J. Huang, Y. Xue, and L. Liu, "Dynamic signature verification technique for the online and offline representation of electronic signatures in biometric systems," *Processes*, vol. 11, no. 1, p. 190, 2023.

[6] X. Song, X. Xia, and F. Luan, "Online signature verification based on stable features extracted dynamically," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 10, pp. 2663–2676, 2016.

[7] C. S. Vorugunti, V. Pulabaigari, R. K. S. S. Gorthi, and P. Mukherjee, "Osvfusenet: online signature verification by feature fusion and depthwise separable convolution based deep learning," *Neurocomputing*, vol. 409, pp. 157–172, 2020.

[8] C. S. Vorugunti, P. Mukherjee, V. Pulabaigari, *et al.*, "Osvnet: Convolutional siamese network for writer independent online signature verification," in *2019 international conference on document analysis and recognition (ICDAR)*, pp. 1470–1475, IEEE, 2019.

[9] H. Li, P. Wei, Z. Ma, C. Li, and N. Zheng, "Transosv: Offline signature verification with transformers," *Pattern Recognition*, vol. 145, p. 109882, 2024.

[10] V. Bharadi, Z. Hamdule, S. Kambli, R. Salvi, R. Chavan, and V. Nimbalkar, "Online signature recognition using transformer networks: An overview," in *2023 6th International Conference on Advances in Science and Technology (ICAST)*, pp. 116–121, IEEE, 2023.

[11] M. M. Hameed, R. Ahmad, M. L. M. Kiah, and G. Murtaza, "Machine learning-based offline signature verification systems: A systematic review," *Signal Processing: Image Communication*, vol. 93, p. 116139, 2021.

[12] K. Bibi, S. Naz, and A. Rehman, "Biometric signature authentication using machine learning techniques: Current trends, challenges and opportunities," *Multimedia Tools and Applications*, vol. 79, no. 1, pp. 289–340, 2020.

[13] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Learning features for offline handwritten signature verification using deep convolutional neural networks," *Pattern Recognition*, vol. 70, pp. 163–176, 2017.

[14] C. Sekhar Vorugunti, V. Pulabaigari, P. Mukherjee, and A. Sharma, "Deepfuseosv: online signature verification using hybrid feature fusion and depthwise separable convolution neural network architecture," *IET Biometrics*, vol. 9, no. 6, pp. 259–268, 2020.

[15] C. S. Vorugunti, P. Mukherjee, and V. Pulabaigari, "Online signature profiling using generative adversarial networks," in *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pp. 894–896, IEEE, 2020.

[16] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, and J. Ortega-Garcia, "Biometric signature verification using recurrent neural networks," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 652–657, IEEE, 2017.

[17] T. Do Thanh, C. T. Nguyen, N. H. Phung, N. H. Minh, and V.-H. Nguyen, "Vit-signet: Combining deep cnn and vision transformer for enhanced signature verification," in *International Conference on Advances in Information and Communication Technology*, pp. 215–224, Springer, 2023.

[18] A. Chauhan, "A review on various aspects of mongodb databases," *International Journal of Engineering Research & Technology (IJERT)*, vol. 8, no. 05, pp. 90–92, 2019.

[19] C. Győrödi, R. Győrödi, G. Pecherle, and A. Olah, "A comparative study: Mongodb vs. mysql," in *2015 13th international conference on engineering of modern electric systems (EMES)*, pp. 1–6, IEEE, 2015.

[20] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Lladós, and U. Pal, "Signet: Convolutional siamese network for writer independent offline signature verification," *arXiv preprint arXiv:1707.02131*, 2017.