SETSCI Conference Proceedings
17, 30-33, 2024

Cognitive Models and Artificial Intelligence Conference

May 25-26, 2024, İstanbul-Türkiye

AICCONF 2024

https://doi.org/10.36287/setsci.17.1.0030

Enhancing Offline Handwritten Signature Identification with Pre-Trained CNN Architectures

Nader Ebrahimpour[1*]

[1]*Papilon Savunma, Ankara, Turkey*
*(naderebrahimpour@papilon.com.tr)

Abstract **–**Handwritten Signature Recognition (HSR) is a vital task in document authentication and verification systems. This paper proposes a novel approach for offline HSR leveraging pre-trained Convolutional Neural Network (CNN) models. CNNs have demonstrated remarkable performance in various computer vision tasks, including image recognition, making them suitable for HSR tasks. Our proposed method uses pre-trained CNN models trained on large-scale image datasets, such as ImageNet, to extract high-level features from handwritten signature images. By fine-tuning these pre-trained models on a dataset of offline handwritten signatures, we aim to transfer the learned knowledge to the task of HSR. We explore different pre-trained CNN architectures, such as MobileNet, ShuffleNet, ResNet, and EfficientNet, and investigate their performance in HSR tasks. Furthermore, we propose a signature verification system that combines the features extracted from pre-trained CNN models with Euclidean Distance (ED) metric to authenticate handwritten signatures. Experimental results on benchmark datasets demonstrate the effectiveness of our proposed approach in achieving state-of-the-art performance in offline HSR tasks.

Keywords –Handwritten Signature Recognition, Euclidean Distance, Pre-Trained CNN Architectures

I. Introduction

In the contemporary landscape, accurately ascertaining and authenticating individuals' identities is pivotal in safeguarding access to various resources. Various methodologies for identity verification exist, among which the utilization of usernames and passwords is the most prevalent and straightforward approach. Nevertheless, this method is susceptible to challenges such as password forgetfulness and unauthorized exploitation. A particularly intriguing alternative for identity verification is the employment of biometric technologies. Biometric methods circumvent the issues above, offering a significantly reduced likelihood of misuse, often bordering on impossibility [1]. Signature-based identification, a subset of biometric techniques, leverages the unique aspects of an individual's signature. Despite criticisms regarding its comparative reliability to other biometric modalities, signature identification remains widely adopted, underscoring the continuous advancement and increasing dependability of HSR systems [2, 3]. Nonetheless, the application of HSR comes with its set of challenges. Primary among these is the diversity in signature styles influenced by cultural differences, the natural evolution of a person's signature over time, and the inconsistencies that arise when capturing images of the same individual's signature at different times [4, 5].

The inherent variability in individual signatures necessitates a more advanced approach than conventional computational techniques. The advancement of machine learning, especially through Deep Learning (DL) technologies, has significantly altered how these challenges are tackled. CNNs are especially remarkable for their exceptional ability to process and recognize complex image patterns, positioning them ideally suited for HSR tasks [6].

Pre-trained CNN models [7] present a viable solution for identifying individuals through handwritten signatures. These models, meticulously fine-tuned on extensive datasets for a broad array of image recognition tasks, offer a workaround to the significant data requirements and computational demands typically required to develop DL models from the ground up. They hold the potential to markedly improve both the accuracy and the efficiency of HSR systems. In this paper, we explore integrating such pre-trained CNN models with the ED [8] metric for identifying offline handwritten signatures, evaluating their capacity to enhance and streamline current signature verification methods, thereby rendering them more dependable and efficient.

In this paper, the structure of our content is arranged as follows: Section 2 delves into the background of the field by examining prior research. Section 3 outlines our approach, including the specific pre-trained CNNs employed and our feature vector extraction strategy. Our study's outcomes and an analysis of these results are shared in Section 4. The paper is concluded in Section 5, where we encapsulate our findings.

II. Backgrounds

The recognition of offline handwritten signatures plays a pivotal role in the security frameworks of various sectors, including finance, governance, and commerce. Numerous studies have explored cutting-edge methods to improve the precision and efficiency of signature verification systems. Jagtap & Hegadi [9] developed a technique leveraging upper and lower envelope analysis along with Eigenvalues, reaching a notable accuracy of 98.5%. In their research, Foroozandeh et al. [10] delved into the use of deep CNNs to verify and recognize offline signatures, underscoring the high effectiveness of models such as VGG16 and SigNet in securing elevated accuracy levels. Additionally, [11] underscored the critical role of biometric authentication, especially in signature verification, to deter the falsification of signatures on crucial documents, employing deep CNNs to distinguish between genuine and counterfeit signatures with a 95.5% accuracy rate. Authors in [12] introduced an approach for signature recognition employing a probabilistic neural network alongside wavelet transform average framing entropy, achieving a recognition accuracy of 92% with a wavelet packet entropy neural network system. [13] advanced an offline signature verification method incorporating deep CNNs and a distinctive approach to local feature extraction, recording high accuracy rates ranging from 94.37% to 99.96%. In their study [14], Ramesh and colleagues highlight the critical role of CNNs in recognizing handwritten characters. They showcase the effectiveness of DL methods by reporting high accuracy rates in identifying Kannada characters. Similarly, in their research [15], Sharma and associates demonstrate the advantages of using a fine-tuned Inception V3 model. They found it to be more effective than other pre-trained models, including VGG 16, VGG 19, ResNet 50, ResNet 101, MobileNet, and EfficientNet, in distinguishing between authentic and counterfeit signatures with high precision. Together, these studies underscore the critical importance of offline HSR in bolstering security and authentication measures, demonstrating the efficacy of diverse techniques and models in delivering precise and dependable outcomes.

III. Proposed method

Our proposed method integrates the advanced capabilities of pre-trained CNN models for feature extraction with the simplicity and effectiveness of ED for similarity assessment and offline HSR.

The shortest straight-line distance between two places in Euclidean space is the ED or L2 distance. The square root of the sum of the squared differences between the matching elements of the two vectors is used to calculate it [8]. Mathematically, the ED between two points A and B in an n-dimensional space with coordinates $A = (a1, a2, \ldots, an)$ and $B = (b1, b2, \ldots, bn)$ is given by:

$$\mathrm{d}(\mathrm{A},\mathrm{B}) = \sqrt{\sum_{i=0}^{n}(ai - bi)^2}$$

The landscape of DL has been transformed by the advent of pre-trained CNN models, which serve as potent instruments for many image recognition tasks. Prominent architectures such as MobileNet [16], ResNet [17], EfficientNet [18], and ShuffleNet [19] stand out for their unique design approaches and advantages. MobileNet shines with its streamlined architecture, employing depthwise separable convolutions to reduce the model's size and computational demands, making it ideal for mobile and embedded vision applications [16]. ResNet, known for its Residual Network design, simplifies training significantly deeper networks by using skip connections that facilitate residual learning [17]. EfficientNet distinguishes itself by systematically enhancing its accuracy and efficiency through strategically scaling its network's width, depth, and resolution, guided by a predefined set of scaling coefficients [18]. On the other hand, ShuffleNet is designed with mobile environments in mind, employing channel shuffle and pointwise group convolution techniques to maintain the model's performance while reducing computational demands [19]. These pre-trained models play a crucial role in developing advanced deep-learning applications, offering a spectrum of options that cater to varying requirements regarding computational efficiency, model dimensions, and precision.

```
# Function to Acquire Signature Images
FUNCTION acquireSignatureImages()
   RETURN all signature images from the specified source

# Function to Detect Signatures in Images
FUNCTION detectSignaturesUsingYOLO(images)
   RETURN list of detected signatures using YOLO algorithm on each image

# Function to Clean Detected Signatures
FUNCTION cleanDetectedSignaturesUsingCycleGAN(detectedSignatures)
   RETURN list of cleaned signatures using CycleGAN for each detected signature

# Function to Pre-process Cleaned Signatures
FUNCTION preprocessSignatures(cleanedSignatures)
   RETURN list of resized and normalized signatures for further processing

# Function for Extracting Features from Signatures
FUNCTION extractSignatureFeatures(signatures, cnnModel)
   RETURN list of feature vectors by applying the specified CNN model to each signature

# Function to Compare Signature Features Against Database
FUNCTION compareSignatureFeaturesWithDatabase(features, database)
   RETURN list of comparison results, each indicating a match or not match status based on Euclidean distance

# Main Execution Flow
FUNCTION main()
   # Load necessary models and databases
   cnnModel = loadPretrainedCNNModel()
   signatureDatabase = loadSignatureFeatureDatabase()
   # Process flow
   images = acquireSignatureImages()
   detectedSignatures = detectSignaturesUsingYOLO(images)
   cleanedSignatures = cleanDetectedSignaturesUsingCycleGAN(detectedSignatures)
   preprocessedSignatures = preprocessSignatures(cleanedSignatures)
   signatureFeatures = extractSignatureFeatures(preprocessedSignatures, cnnModel)
   comparisonResults = compareSignatureFeaturesWithDatabase(signatureFeatures, signatureDatabase)
      # Display results
   displayResults(comparisonResults)
# Start the program
main()
```

Fig. 1 Pseudocode of Proposed Method

According to the pseudocode demonstrated in Figure 1, the proposed methodology for offline HSR is initiated through a meticulously structured sequence of steps, beginning with the crucial 'Image Acquisition' phase. This initial phase is foundational, as the system gathers images containing signatures to set the stage for the recognition process. Once the collection of images is securely established, the methodology advances to the 'Detect Signature' stage. At this stage, the widely acclaimed YOLO (You Only Look Once) [20] framework, known for its real-time object detection capabilities, is employed. Utilizing YOLO's powerful algorithms enables the precise localization and identification of signatures within the images, marking an essential stride toward effective signature authentication.

After successfully identifying signatures, the methodology proceeds to the 'Clean Signature' step. At this juncture, CycleGAN [21], a sophisticated form of Generative Adversarial Network known for its image translation capabilities, plays a crucial role. This phase concentrates on the meticulous refinement of detected signatures, potentially involving the reduction of unwanted noise. These refinements aim to create a standardized signature image, simplifying the path forward for more straightforward analysis by enhancing image quality and uniformity.

The next phase, 'Pre-processing,' involves meticulously preparing the cleaned signature images for feature extraction. This preparation contains various processes such as resizing, normalizing, and potentially augmenting the data to ensure uniformity. Preserving consistent format and quality across the signature images is crucial, as it significantly enhances the model's efficiency and accuracy in the feature extraction phase.

The 'Feature Extraction' step is the core component of the method. Here, relevant feature vectors are extracted from the pre-processed signature images using a CNN model that has already been trained. These feature vectors represent the key elements of the comparison signatures. After that, the database of recognized signatures is used to compare the extracted features. This comparison is carried out by calculating the ED between the input signature's feature vector and those in the database. In the ED, the smaller distances denote more remarkable similarity.

Two outcomes are possible from this comparison: "Match" and "Not Match." If the computed ED is smaller than a preset threshold, the system will declare a match, signifying that the signature has been verified and accepted. This threshold determines whether or not the signature is recognized. If the distance is above this threshold, then none of the known signatures in the database match the signature.

This approach leverages the capabilities of cutting-edge neural networks and image processing techniques to provide a reliable solution for HSR tasks, offering a comprehensive and automated system for signature verification.

IV. Results and Discussions

Our experiments utilizing pre-trained CNN models for offline HSR yielded promising outcomes. Initially, the model's performance was benchmarked against the CEDAR dataset [22]. This dataset was developed with the active involvement of 55 participants, each contributing 24 signatures, culminating in a collection of 1320 authentic signatures. Figure 2 presents a sample from this dataset.

Fig. 2 A Sample Signature from CEDAR dataset

The models were developed on a system with an Intel i7 eight-core processor, 32GB of RAM, a 512GB SSD, and an NVIDIA GeForce GTX 3070 Ti GPU using the PyTorch framework [23]. The dataset was divided into training and evaluation sets at random to facilitate the implementation. In a standardized setting, several models were trained during this simulation. Adam's optimization algorithm was selected as the optimization technique, with a learning rate of 0.001. We evaluate the models based on parameters such as loss, accuracy, precision, recall, and model size.

Accuracy is the percentage of correct predictions made by a model out of all predictions made [24]. The formula for accuracy is:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions made}}$$

Precision measures the proportion of positive identification that was actually correct. It is particularly important in situations where the cost of false positives is high [24]. The formula for precision is:

$$\text{Pecision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall, also known as sensitivity, measures the model's ability to detect all actual positives from the data [24]. The formula for the recall is:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Table 1. Comparative Performance Metrics of CNN Models for HSR

	MobilenetV2	**Resnet18**	**EfficientnetB0**	**shufflenetV2**
Loss	0.0558	0.0537	0.0333	0.3577
Accuracy	0.9919	0.9927	0.9954	0.9890
Precision	0.9919	0.9927	0.9953	0.9891
Recall	0.9919	0.9926	0.9953	0.9890
Model Size	10.8 MB	45.4MB	18.0MB	2.8MB

We analyze four distinct pre-trained CNN models: MobileNetV2, ResNet18, EfficientNetB0, and ShuffleNetV2. Table 1 depicts each model's performance across various metrics pivotal to HSR efficacy. A model with higher predictive accuracy has a lower loss score. The difference between the actual data and the predictions made by the model during training is measured by loss. EfficientNetB0's minimal loss value of 0.0333 indicates that it is incredibly accurate at identifying handwritten signatures. On the other hand, ShuffleNetV2's high loss of 0.3577 can point to flaws in the model's generalization capacity or the efficiency of its training schedule. The percentage of accurate predictions among all predictions made by the models is known as accuracy. The best accuracy (0.9954) is obtained by EfficientNetB0, which is followed by ResNet18 (0.9927), MobileNetV2 (0.9919), and ShuffleNetV2 (0.9890). This suggests that EfficientNetB0 is more adept at accurately identifying signatures. Precision measures how well positive predictions—that is, signatures that are correctly recognized as authentic—fit. The models that exhibit the highest precision (0.9953 and 0.9927, respectively) are EfficientNetB0 and ResNet18. This suggests that when these models identify a signature as authentic, there is a high likelihood that it is accurate. Recall evaluates the model's ability to recognize every single valid signature. All models have relatively high recall scores; EfficientNetB0 and ResNet18 have scores of 0. 9953 and 0.9926, respectively, demonstrating their ability to identify real signatures consistently. Each model's storage needs are reflected in its size. At 2.8 MB, ShuffleNetV2 is the smallest model, which might be advantageous when deployment size is limited or in environments with limited storage. In contrast, ResNet18 is the largest at 45.4 MB, which may affect its deployability in resource-constrained settings.

These findings highlight the trade-offs between model complexity, size, and performance. ShuffleNetV2's small size might be advantageous for some applications, but EfficientNetB0 appears to be the best at recognizing offline handwritten signatures due to its high accuracy and low loss. These factors would need to be considered when choosing a model for real-world use, ensuring that the model's advantages match the demands and limitations of the application.

V. CONCLUSION

Finally, this work has effectively illustrated the feasibility of offline HSR using pre-trained CNN models. The advantages and disadvantages of several CNN architectures, such as MobileNetV2, ResNet18, EfficientNetB0, and ShuffleNetV2, have been made clear through in-depth comparative research. The tested models showed that EfficientNetB0 had the lowest loss and the highest accuracy, making it the most accurate in identifying signatures. Despite the impressive performance of EfficientNetB0, the results also highlighted the significance of considering the model size in practical applications, as evidenced by the compact and efficient nature of ShuffleNetV2. The results highlight the complex trade-off DL model deployments must make between accuracy, efficiency, and computational demands. Moreover, this study provides opportunities for future research to improve the resilience of HSR systems.

REFERENCES

[1] A. Sarkar and B. K. Singh, "A review on performance, security and various biometric template protection schemes for biometric authentication systems," Multimedia Tools and Applications, vol. 79, no. 37, pp. 27721-27776, 2020.

[2] M. Diaz, M. A. Ferrer, D. Impedovo, M. I. Malik, G. Pirlo, and R. Plamondon, "A perspective analysis of handwritten signature technology," Acm Computing Surveys (Csur), vol. 51, no. 6, pp. 1-39, 2019.

[3] E. N. Zois, M. Papagiannopoulou, D. Tsourounis, and G. Economou, "Hierarchical dictionary learning and sparse coding for static signature verification," in Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2018, pp. 432-442.

[4] W. Hou, X. Ye, and K. Wang, "A survey of off-line signature verification," in 2004 International Conference on Intelligent Mechatronics and Automation, 2004. Proceedings., 2004: IEEE, pp. 536-541.

[5] E. J. Justino, F. Bortolozzi, and R. Sabourin, "A comparison of SVM and HMM classifiers in the off-line signature verification," Pattern recognition letters, vol. 26, no. 9, pp. 1377-1385, 2005.

[6] R. Ghosh, "A Recurrent Neural Network based deep learning model for offline signature verification and recognition system," Expert Systems with Applications, vol. 168, p. 114249, 2021.

[7] X. Han et al., "Pre-trained models: Past, present and future," AI Open, vol. 2, pp. 225-250, 2021.

[8] K. L. Elmore and M. B. Richman, "Euclidean distance as a similarity metric for principal component analysis," Monthly weather review, vol. 129, no. 3, pp. 540-549, 2001.

[9] A. B. Jagtap and R. S. Hegadi, "Offline handwritten signature recognition based on upper and lower envelope using eigen values," in 2017 World Congress on Computing and Communication Technologies (WCCCT), 2017: IEEE, pp. 223-226.

[10] A. Foroozandeh, A. A. Hemmat, and H. Rabbani, "Offline handwritten signature verification and recognition based on deep transfer learning," in 2020 International Conference on Machine Vision and Image Processing (MVIP), 2020: IEEE, pp. 1-7.

[11] Q. S.-u. Mosaher and M. Hasan, "Offline handwritten signature recognition using deep convolution neural network," European Journal of Engineering and Technology Research, vol. 7, no. 4, pp. 44-47, 2022.

[12] K. Daqrouq, H. Sweidan, A. Balamesh, and M. N. Ajour, "Off-line handwritten signature recognition by wavelet entropy and neural network," Entropy, vol. 19, no. 6, p. 252, 2017.

[13] H.-H. Kao and C.-Y. Wen, "An offline signature verification and forgery detection method based on a single known sample and an explainable deep learning approach," Applied Sciences, vol. 10, no. 11, p. 3716, 2020.

[14] G. Ramesh, G. N. Sharma, J. M. Balaji, and H. Champa, "Offline Kannada handwritten character recognition using convolutional neural networks," in 2019 IEEE international wie conference on electrical and computer engineering (WIECON-ECE), 2019: IEEE, pp. 1-5.

[15] N. Sharma et al., "Offline signature verification using deep neural network with application to computer vision," Journal of Electronic Imaging, vol. 31, no. 4, pp. 041210-041210, 2022.

[16] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770-778.

[18] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in International conference on machine learning, 2019: PMLR, pp. 6105-6114.

[19] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 18-23 June 2018 2018, pp. 6848-6856, doi: 10.1109/CVPR.2018.00716.

[20] M. Hussain, "YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection," Machines, vol. 11, no. 7, p. 677, 2023.

[21] 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 workshops, 2020, pp. 808-809.

[22] D. Gumusbas and T. Yildirim, "Offline signature identification and verification using capsule network," in 2019 IEEE international symposium on innovations in intelligent systems and applications (INISTA), 2019: IEEE, pp. 1-5.

[23] I. Vasilev, D. Slater, G. Spacagna, P. Roelants, and V. Zocca, Python Deep Learning: Exploring deep learning techniques and neural network architectures with Pytorch, Keras, and TensorFlow. Packt Publishing Ltd, 2019.

[24] D. L. Olson and D. Delen, Advanced data mining techniques. Springer Science & Business Media, 2008.