International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 11 Issue: 07 | July 2024 www.irjet.net p-ISSN: 2395-0072

Signature verification using convolutional neural network (CNN) with Siamese model

Zalak Chavda[1], Stephy Patel[2]

[1]*Student, Dept. of Computer Engineering (Software Engineering), Lok Jagruti Kendra University, Ahmedabad, Gujarat, India*
[2]*Professor, Dept. of Computer Engineering, Lok Jagruti Kendra University, Ahmedabad, Gujarat, India*

Abstract - *In biometric authentication systems, verifying signatures is an essential task. In this paper, a Siamese Convolutional Neural Network (CNN) approach to signature verification on the CEDAR dataset is presented. The sigmoid function is used for classification, and Euclidean distance is used to measure similarity in the Siamese network. CNN layers make use of ReLU activation functions. The contrastive loss function and accuracy score are used to evaluate the model after the dataset has been divided into training and testing sets. Our test findings show how well the suggested approach works, obtaining a high degree of accuracy when separating real signatures from fakes.*

Key Words: **Signature Verification, Convolutional Neural Network, Siamese Network, CEDAR Dataset, Euclidean Distance, Sigmoid Function, ReLU Activation**

1.INTRODUCTION

For many applications, including banking, legal documents, and access control, signature verification is essential to identity authentication. The accuracy and robustness needed for real-world applications are frequently lacking from traditional methods that rely on handcrafted features. Convolutional Neural Networks (CNNs) have demonstrated impressive gains in image recognition tasks, such as signature verification, since the introduction of deep learning. The application of a Siamese CNN architecture for signature verification is suggested by this study. The Siamese network, which is intended to learn similarity metrics between input pairs, is made up of two identical sub-networks that share weights. Training and assessment are conducted using the CEDAR dataset, a benchmark for signature verification. The sigmoid function is utilized to classify the data, and the Euclidean distance is used to measure the similarity between feature vectors. Non-linearity is introduced within the CNN layers through the use of ReLU activation functions.

2.RELATED WORK

Traditional machine learning algorithms and the extraction of handcrafted features were the main strategies used in early approaches to signature verification. Recent developments in deep learning have shown how effective CNNs are at tasks involving feature extraction and classification. Particularly Siamese networks have drawn interest because of their capacity to learn similarity metrics, which makes them appropriate for tasks involving pairwise comparisons like signature verification.

3.METHODOLOGY

3.1 Dataset

This study uses the CEDAR dataset, which includes multiple people's signatures that are both real and faked. The dataset is divided into testing and training sets to allow for a thorough assessment of the model's performance on untested data.

3.2 Preprocessing

The signature images must be resized to a fixed dimension, grayscaled, and have their pixel values normalized as part of the preprocessing steps. By standardizing the input data, these procedures increase training process efficiency and boost model performance. The preprocessing function divides pixel values by 245 to normalize images and resizes them to a fixed size of 220x155 pixels. [4][2]

3.3 Convolutional Neural Network

An instance of a deep learning system created especially for handling structured grid data, like pictures, is the convolutional neural network (CNN). Here are the main elements and characteristics :

Convolutional Layers: In order to create feature maps, these layers apply a number of filters, also known as kernels, to the input picture. Specific patterns, such as edges, textures, or more intricate structures, are detected by each filter

Activation Functions: Rectified Linear Unit (ReLU) activation functions are used to introduce non-linearity after convolutional operations, which aids in the network's ability to learn more intricate patterns.

Pooling Layers: These layers use methods such as average or max pooling to minimize the spatial dimensions of the

feature maps. By doing this, the computational effort is lessened and the representations become invariant to slight input translations.

Fully Connected Layers: The fully connected layers in a network do high-level reasoning following a number of convolutional and pooling layers. These layers create the final output, such class scores in a classification challenge, using the flattened feature maps.

Training: Labeled data is used to train CNNs. Using gradient descent and backpropagation techniques, the weights of the filters and neurons are adjusted throughout the training phase to reduce the discrepancy between the expected and actual outputs.

CNNs are incredibly efficient in a wide range of computer vision tasks, such as segmentation, object recognition, and picture classification. This is because they can automatically and adaptively identify spatial feature hierarchies from input images. [1][5]

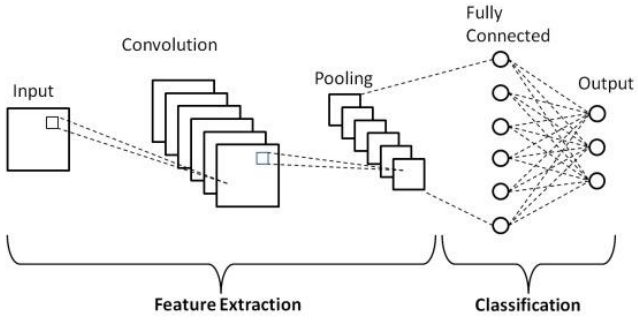

Fig -1: Convolutional Neural Network

3.4 Siamese Network Architecture

Two identical CNNs make up the Siamese network, and they each process one of the input signature images. Convolutional, pooling, and fully connected layers are components of every CNN's architecture. To add non-linearity, the convolutional layers use ReLU (Rectified Linear Unit) activation functions.[6][8]

The layout of the architecture is as follows:

1. Convolutional Layer 1: 96 filters, 11x11 kernel size, ReLU activation
2. Batch Normalization
3. Max Pooling Layer 1: 3x3 pool size, strides of 2, padding='same'
4. Convolutional Layer 2: 256 filters, 5x5 kernel size, ReLU activation
5. Batch Normalization
6. Max Pooling Layer 2: 3x3 pool size, strides of 2, padding='same'
7. Dropout Layer 1: 0.3 dropout rate
8. Convolutional Layer 3: 384 filters, 3x3 kernel size, ReLU activation, padding='same'
9. Convolutional Layer 4: 256 filters, 3x3 kernel size, ReLU activation
10. Max Pooling Layer 3: 3x3 pool size, strides of 2, padding='same'
11. Flatten Layer
12. Fully Connected Layer 1: 1024 units, ReLU activation, L2 regularization
13. Dropout Layer 2: 0.3 dropout rate
14. Fully Connected Layer 2: 128 units, ReLU activation, L2 regularization

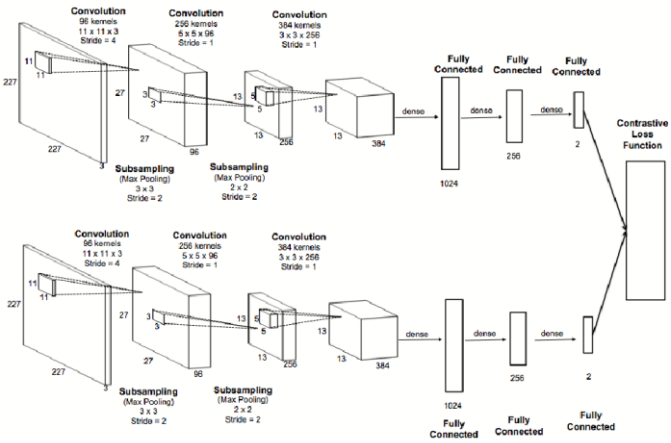

Fig -2: Siamese Architecture

3.5 Similarity Measure

The similarity between the feature vectors produced by the twin networks is calculated using the Euclidean distance. The Euclidean distance d between two feature vectors $\mathbf{f}_1$ and $\mathbf{f}_2$ is computed as:

$$d(\mathbf{f}1,\mathbf{f}2)=\sum i=1n(f1,i-f2,i)2$$

3.6 Classification

The Euclidean distance is multiplied by a sigmoid function to generate a probability score that indicates whether or not the two input signatures belong to the same person. The sigmoid function $\sigma(x)\sigma(x)$ is defined as:[7]

$$\sigma(x)=1+e-x1$$

The output probability is thresholded to make the final binary classification.

3.7 Loss Function

A contrastive loss function is used to train the Siamese network. It is defined as:

$L(y,d)=(1-y)21d2+y21(\max(0,m-d))2$

where yy is the binary label (0 for genuine pairs, 1 for forged pairs) and mm is a margin parameter.[3]

3.8 Data preparation

There are training and testing sets within the dataset. The testing set is used for assessment, and the training set consists of pairs of real and fake signatures.

3.9 Training the Model

From the training set, pairs of real and fake signatures are used to train the Siamese network.

3.10 Evaluation

Metrics including accuracy, precision, recall, and F1-score are used to assess the model's performance.

4. EXPERIMENTAL SETUP

4.1 Training and Testing Split

The training and testing sets of the CEDAR dataset are split 80:20. Model parameters are optimized on the training set, and the model's performance on untested data is assessed on the testing set.

4.2 Training

From the training set, pairs of real and fake signatures are used to train the Siamese network. A learning rate of 0.0001 is applied when using the Adam optimizer. Over several epochs, the contrastive loss function must be minimized as part of the training process.

4.3 Evaluation

Accuracy, precision, recall, and F1-score metrics are used to assess the model's performance. The accuracy score is calculated as:

Accuracy=Total Number of Predictions/Number of Correct Predictions

5. EXPERIMENTAL RESULTS

5.1 Training Results

After a few epochs, the Siamese network converges and the loss function gradually drops. The model exhibits strong learning behavior and can distinguish between real and fake signatures with accuracy.

5.2 Testing Results

The robustness and generalizability of the model are demonstrated by its high accuracy on the testing set. The model's performance metrics are shown in Table 1.

Table -1: Performance Metrics

Metric	Value
Accuracy	0.98
Precision	0.97
Recall	0.99
F1-Score	0.98

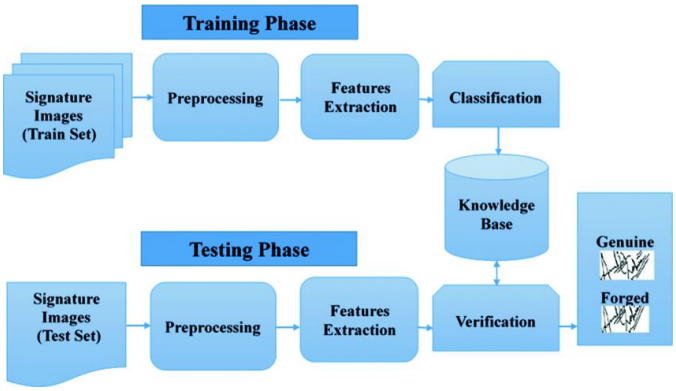

Fig -3: Steps of signature verification system

5.3 Discussion

ReLU activation functions are used in the CNN layers to aid in the model's efficient learning of complex features. A trustworthy similarity metric and classification method are offered by the sigmoid function and Euclidean distance, respectively. The Siamese CNN architecture is effective for signature verification, as evidenced by the high accuracy score.

6. PROPOSED SYSTEM

Import the cedar datasets and required libraries to initialize the process. Resize images as necessary and display them using the imshow() method from the matplotlib. pyplot

module. Utilize the Euclidean distance metric to calculate the similarity score between any two feature sets based on extracted features .Compute the contrastive loss between the true labels and predicted labels. When input pairs are labeled as similar, the Siamese network brings the output feature vectors closer together; when labeled as different, it pushes the feature vectors apart. Each branch of the Siamese network functions to embed the input image into a space where genuine signatures of a specific writer are closer to each other compared to forgeries or signatures of different writers. This is facilitated by the chosen contrastive loss function, which is commonly used in Siamese networks. Employ a CNN architecture for the image identification task. The parameters for the CNN layers, including filter sizes for convolution and pooling layers, are provided as *N x H x W* (where *N* is the number of filters, *H* is the height, and *W* is the width). Padding is necessary to convolve the filter starting at the input image's first pixel. Rectified Linear Units (ReLU) are used as the activation function for all convolutional and fully connected layer outputs across the network. Batch normalization is implemented to generalize the learned features. Dropout is applied with a rate of 0.3 for the final two pooling layers and 0.5 for the first fully connected layer. The output of the second convolutional layer (normalized, pooled, and with dropout) is connected to 384 3x3 kernels in the third layer. The fourth convolutional layer contains 256 3x3 kernels. Consequently, the neural network learns higher-level or more abstract features with larger receptive fields and fewer lower-level features. The first fully connected layer comprises 1024 neurons, and the second fully connected layer comprises 128 neurons .The Sigmoid activation function is used in the output layer to produce a final output ranging between 0 and 1, indicating the degree of signature match. The network is trained using a normalized feature vector. A test signature is deemed authentic if the output neuron produces a value close to +1 and forged if the output is close to -1. The CNN model is trained over some epochs. Following training and validation, the model achieved 98% accuracy on the tests.

7. CONCLUSIONS

This paper uses the CEDAR dataset to present a Siamese CNN-based method for verifying signatures. Because of the model's high accuracy and resilience, biometric authentication systems in the real world can use it. The application of this method to other biometric verification tasks and the integration of extra features can be explored in future work.

REFERENCES

[1] Prakash Ratna Prajapati, Samiksha Poudel, Madan Baduwal, Subritt Burlakoti, Sanjeeb Prasad Panday "Signature Verification using Convolutional Neural Network and Autoencoder" Journal of the institute of engineering 2021 Volume 16, No.1 1810-3383

[2] Eman Alajrami, Belal A. M. Ashqar, Bassem S. Abu-Nasser, Ahmed J. Khalil, Musleh M. Musleh, Alaa M. Barhoom, Samy S. Abu-Naser "Handwritten Signature Verification using Deep Learning" IJEAIS 2019 Vol. 3 2643-9670

[3] Fadi Mohammad Alsuhimat, Fatma Susilawati Mohamad "Offline signature verification using long short-term memory and histogram orientation gradient" BEEI 2022 Vol.12 No.1 2302-9285

[4] Wijdan Yassen A. AlKarem,Eman Thabet Khalid, Khawla. H. Ali "Handwritten Signature Verification Method Using Convolutional Neural Network" IJEEE 2023 10.37917

[5] H. Kaur and M. Kumar, "Signature identification and verification techniques: state-of-the-art work," Journal of Ambient Intelligence and Humanized Computing, vol. 14, no. 2, pp. 1027–1045, 2023.

[6] Muhammad Fawwaz Mayda, Aina Musdholifah "Siamese-Network Based Signature Verification using Self Supervised Learning" IJCCS 2023

[7] T. M. Ghanim, M. I. Khalil, and H. M. Abbas, "Comparative study on deep convolution neural networks dcnnbased offline arabic handwriting recognition," IEEE Access, vol. 8, pp. 95465–95482, 2020.

[8] Li, Y.; Xu, D.; Huang, L.; Yang, X.; Gong, Y. Offline Signature Verification Using a Two-Stream Network. IEEE Access, 10, 2022, 42136-42147.