ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ

УДК 004.93
DOI 10.22213/2410-9304-2023-3-27-39

Exploiting Deep Learning Techniques for the Verification of Handwritten Signatures

F. B. Albasu, Kalashnikov Izhevsk State Technical University, Izhevsk, Russia
M. A. Al Akkad, Ph.D., Kalashnikov Izhevsk State Technical University, Izhevsk, Russia

Biometric featuresare common measures of identity verification where signaturesarethe most used type. The digital technology has given birth to new ways of biometric identification, such as fingerprints, iris and face recognition,while dealing with handwritten signatures is still a challenging task, because handwritten signatures are more prone to forgery than other means of verification due to issues like computer error, insufficient datasets, and loss of information. This work aims to develop a system that takes a signature image as its input and determines whether the signature is genuine written by its author or forged by another individual. The system is based on a neural network algorithm called Convolutional Siamese Neural Networks, which is used for deep learning and computer vision as well as other machine learning tasks such as natural language processing and digital signal processing.A Contrastive Loss function which compares the Euclidean distance of the output feature vectors is used, and a writer-independent model is used for training and image classification. This work's objective is toenhance the precision of signature verification and take it as a base for future work on signature verification and use it in user identification, fraud detection and prevention, and forensic investigation applications. The system can be applied in banking, government and private organizations, and forensic investigation for identity and document verification, impersonation and fraud detection and prevention, crime and judicial investigation, and passport verification.

Keywords: Biometric identification, Pattern recognition, Signature verification, Authentication, Deep learning.

Introduction

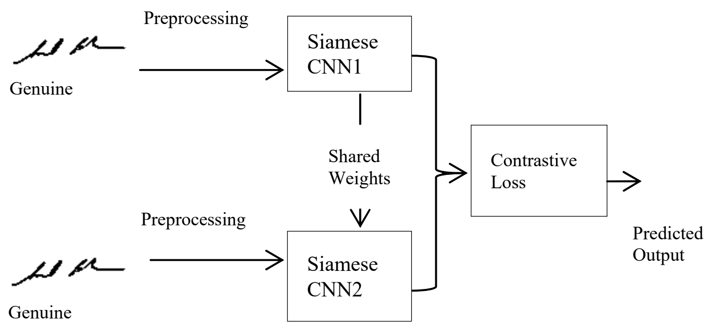
Biometric identification is the process of verifying and authenticating people based on their physiological and behavioral characteristics. Physiological types of biometrics are based on an individual's physical and biological features such as face, iris and fingerprints while behavioral types are based on individual's behavioral traits, mainly their signature. Signature verification in particular is the most common type of biometric identification particularly regarding the verification of official documents, bank cheques, and others. Signature verification is divided into *dynamic signature verification*, and *static signature verification*, where dynamic or online signature verification involves an individual signing on specialized equipment such as a pressure sensitive tablet with a specialized pen designed for the tablet, while static or offline based verification involves an individual signing on a document or a piece of paper and scanning the signature into the computer before verifying the signature [1, 2]. Online signature verification still remains obsolete due to the need for specialized equipment for capturing and recognizing the signatures,besides the need for ways to recognize signatures taken in an offline environment using online systems and able to distinguish genuine signatures accurately.While offline handwritten signature recognition is more adopted than online handwritten signature recognition, it poses more challenges and is far less efficient than its online counterpart, because online handwritten signature recognition takes into account more features of a signature than offline which often lacks information. Offline signature recognition still remains difficult to achieve despite a larger amount of research done than online signature recognition, due to loss of a lot of dynamic information [3]. When dealing with offline signature verification, a set of signatures have to be collected with genuine signatures which are employed in the training of the model which will then be tested using a different set consisting of genuine signatures and forgeries to assess the system's performance [4, 5]. There are *writer-independent* signature verification systems, when a single model is used for image classification [6], and *writer-dependent systems,* when one model is trained for each user [7]. Training writer-independent models raises more challenges than writer-dependent classifiers and performs worse than writer-dependent classifiers, but poses a better chance of generalization.Some of the major challenges with offline handwritten signature recognition include lack of datasetsdue to security purposes, high-intra class variability where handwritten signatures often show large variability between samples of the individual. This often leads to a larger percentage in False Acceptance Rate (FAR), False Rejection Rate (FRR), and Equal Error Rate (EER). When working with forgeries, there is little inter-class variability, making competent forgeries difficult to identify from real signatures. This further exacerbates significant intra-class variability when forgeries are made specifically to target an individual, making the resemblance too convincing[8].The largest publicly available dataset is the GPDS-960 which contains signatures from 881 users with 24 genuine samples and 30 skilled forgeries per user. The process of handwritten signature verification consists of*data acquisition* which involves collecting genuine and forged signatures,*preprocessing* that involves enhancement of images by removing features that may hinder the model's development, and restoring lost information that would enhance the model's accuracy,*feature extraction*of the signature,and *model training*that involves classifying the images.A traditional neural network learns to predict many classes which usually cause an issue when classes are added or removed from the data. In such cases, the neural network (NN) needs to be updated and retrained on an entire dataset. Deep neural networks also require a vast amount of data to train on. In contrast, Convolutional Siamese Neural Networks (CSNN) learns a similarity function. As a result, it can be trained to detect if the inputs are identical which allows for classification of new types of data without having to retrain the network.In this work aCSNNwas used for classification. Convolutional-SiameseNeural Networkshave two or more similar subnetworks, which share the same setup, parameters, and weights. The updating of parameters is duplicated throughout both sub-networks. These networks are utilized in a wide range of applications to determine the similarity of inputs by comparing feature vectors.

Development of the system

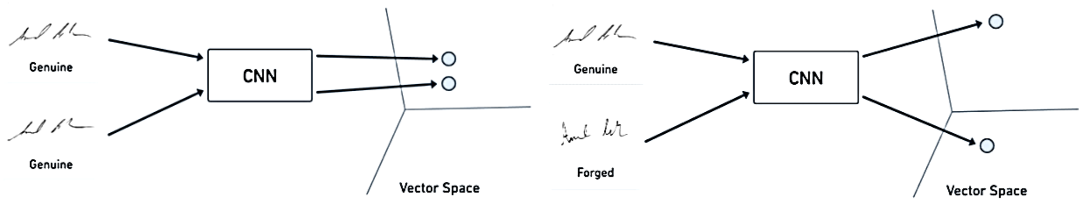
The system aims to enhance precision of signature verification using CSNN along with Contrastive Loss as the loss function which compares the Euclidean distance of the output feature vectors [9, 10].CSNN is an architecture that has two or more similar subnetworkssharing the same setup, parameters, and weights to determine the similarity of inputs by comparing feature vectors. The updating of parameters is duplicated throughout both subnetworks.Contrastive loss calculates the distance between similar and dissimilar input and output pairs of a network by projecting them onthe Euclidean space. This means that signatures of the same class(genuine-genuine) would be placed close to each other as opposed to signatures of different classes(genuine-forged) which would be placed far from each other on the plane as shown on Fig.1.The system can be integrated into different larger systems and serve as a method of verifying and authenticating users and customers digitally with the use of offline signatures. The system can be applied in banking, government and private organizations, and forensic investigationfor identity and document verification, impersonation and fraud detection and prevention, crime and judicial investigation, and passport verification. The block diagram of the system, as shown on Fig. 2, involves *data acquisition* that is usually done in a controlled environment to minimize the amount of noise and maximize the amount of features that could give the best possible accuracy when training the system, *preprocessing* to prepare the dataset being fed into the system by removing any feature that might hinder the learning optimization of the system,*feature extraction and learning* wherein the case of convolutional neural networksthere's no need to extract the features since the network is going to learn the features before classifying the signatures,and *model training and testing*where the images are being fed into the network in order to learn the features and use those features for classifying the signatures, then testingis done with a different set of signatures for analysis.The system activity diagram is shown on Fig. 3.

Data Acquisition Preprocessing. Data acquisition process involves collecting both genuine and forged signatures from users across multiple sessions. The user provides multiple samples of their genuine signatures for each session in a form containing several cells which often have sizes that match commonscenarios such as bank cheques and credit card vouchers [11].

For the forgery, users are provided with genuine samples of other users to imitate multiple times in order to obtain random, simple and skilled forgeries of the signatures. After the signatures have been acquired, they are then uploaded to the system for preprocessing. This is done to remove any factors that might hinder the model's accuracy during training and ensure that the signatures are standard and ready for feature extraction.The main preprocessing steps *extraction of signature* from where it is located in a document, *noise reduction* to remove the noise caused during the scanning of the documents, *resizing and centering* where the image is cropped down to the signature boundaries and then centered on the image [12] maintaining high quality resolution, *binarization* that involves transforming the grayscale images to binary, but it is not taken into account in online verification systems because it’s already been done on the dataset, *thickening or thinning* that is the process of growing selected regions of the foreground by inserting pixels to the objects or removing the pixels from the foreground by making it one pixel thick, while preserving the extent and connectivity of the foreground by forming a 4 or 8-connectivity. Other preprocessing steps include clutter removal which involves removal of unconnected dots by masking and skeletonization which involves removal of selected foreground pixels from the binary image [13, 14].The preprocessing algorithm is shown on Fig.4.In order to feed the images into the network, they need to be organized, labeled and preprocessedusing *Binarization*to transform the image from grayscale to binary image to reduce the complexity and execution time, *Resizing*to resize the image based on a canvas Cof size $H \times W$converting it into tensors that will be fed into the network, and *Image Pairing*to group the images into pairs of genuine and forged, labeling them 1 if both images are genuine and from the same author and 0 if not.



a



b

Fig. 1. a) Overview of the verification process, b) Representation of signatures on Euclidean space

Рис. 1. а) обзор процесса проверки; б) представление подписей на евклидовом пространстве

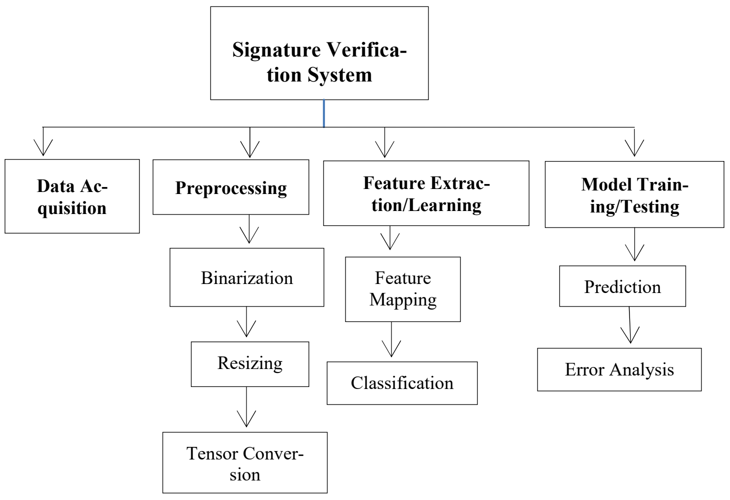


Fig. 2. The block diagram of the system

Рис. 2. Структурная схема системы

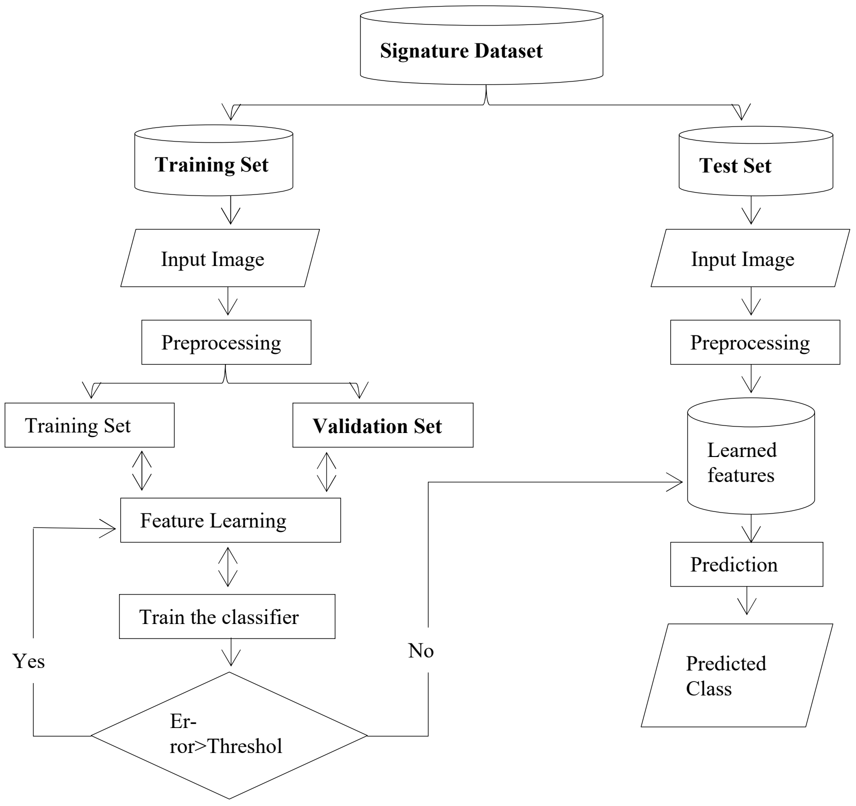


Fig. 3. Thesystem's activity diagram

Рис. 3. Диаграмма активности системы

Feature Extraction and Learning. The main purpose of handwritten signature verification is to minimize the number of features in a dataset, creating new features from current ones, then removing the original characteristics [15, 16]. The feature extraction techniques can be classified as *static* and *pseudo-dynamic* where pseudo-dynamic features attempt to recover dynamic information like speed and pressure from the signature execution process. Another classification involves distinguishing between *global* and *local features* where global features describe the whole signature image factoring in characteristics such as width and height while local features describing specific parts of the images either by segmenting the image or dividing the image in a grid and applying feature extractors to each part of the image [17]. Function and parameter features can be used for signature verification, where *function features* characterize the signature in terms of a time function whose values constitute a feature set while *parameter features* characterize the signature as a vector of elements, each one representing a feature value [18, 19].

The result of a preprocessed signature image is shown on Fig. 5. Parameter features can be categorized into global and local parameters with local parameters further divided into component oriented and pixel oriented. Function features are adopted in the case of online or dynamic signature verification with the most common ones being position, velocity and acceleration. These features are captured during the data acquisition phase, using devices specifically developed for capturing such features. Parameter features such as the pen-down time ratio, number of pen lifts and other global parameters that are numerically derived like the average, the root mean, minimum and maximum values of position, displacement, acceleration and speed are mainly used for dynamic signature verification. Local parameters are more widely used in static or offline signature verification [20]. These are either *component-oriented features* such as geometric, slant, contour, and orientation features or *pixel-oriented features* such as grid, texture and shadow-code features [21, 22]. Lately, Deep learning (DL) approaches are being adopted for feature learning and classification because as opposed to traditional Machine learning (ML) algorithms which break down the problem and solve different parts separately before aggregating the results to give a final output, DL techniques solve problems using an *end-to-end approach*, i.e. take an image, extract several different features from it and deliver an output classifying what type of input it is [23].

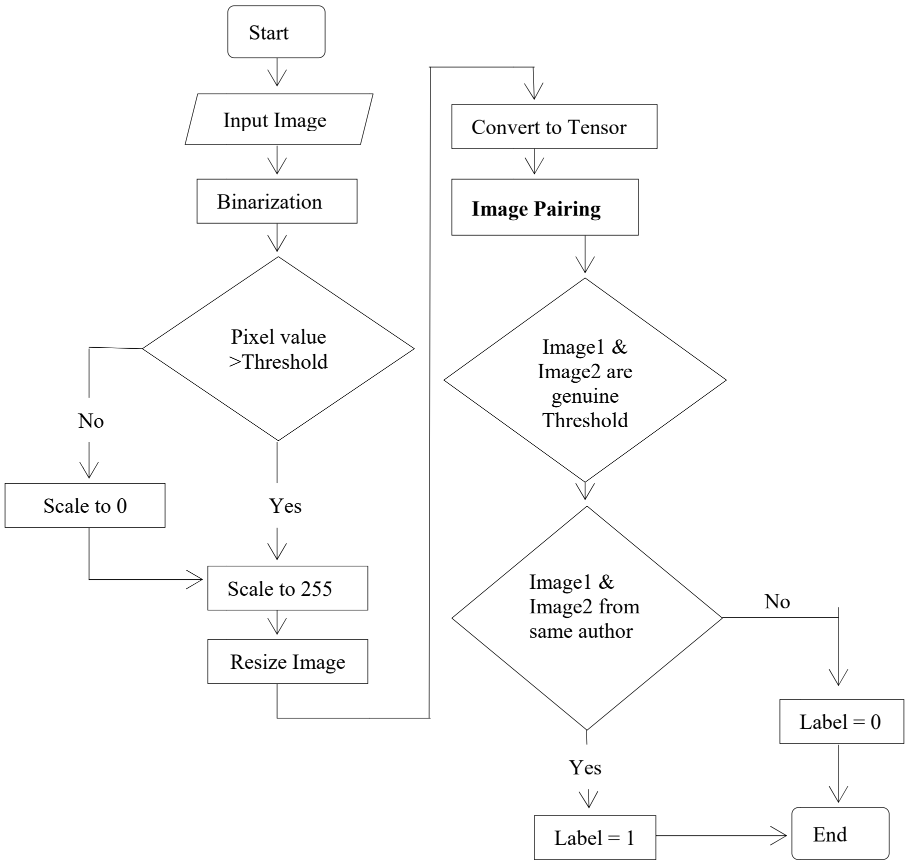


Fig. 4. The preprocessing algorithm

Рис. 4. Алгоритм предварительной обработки

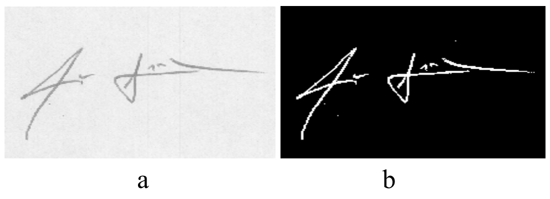


Fig. 5. a) Original signature, b) Preprocessed signature

Рис. 5. а) исходная подпись;
б) предварительно обработанная подпись

Unlike most of the classical ML techniques that are restrictive and use handcrafted feature extractors, DL techniques learn both low-level and high-level features, eliminating the need for handcrafted feature extraction and can be applied on various domains, due to techniques such as *transfer learning* which leverages the use of pre-trained networks for other applications within the same domain. In a writer-dependent approach, each model is trained for a user, using their real signatures and forgeries at random from other signers. This is the more commonly used approach in the literature. The writer-independent approach uses a single classifier for all users [24]. There have been cases where both approaches have been combined in a single hybrid solution [25] [26, 27]. In [28] a ConvNet (CNN) model based on the VGG16 architecture using the ICDAR 2011 SigComp dataset was proposed achieving accuracies between 95 and 97 %. In [29] a CNN model using python, Keras library and TensorFlow, trained on a dataset of 300 signature images (150 real and 150 forged) was used achieving accuracies on different splits of the dataset, where on an 8:2 ratio split, a training accuracy of 99.9% was achieved. In [30] a classifier was proposed,trained ona publicly available datasets MCYT and BHSig260, and achieved accuracy between 95.29% and 97.79%. Hidden Markov Models are used when the state of a system cannot be observed but only the result of some probability function is available [31]. In [32] an off-line system was presented based on the Hidden Markov Models with a total dataset of 4000 signatures, and in [33], afteradding a pixel distribution feature to the model. Other HMM based signature verification techniques are described in [34, 35]. Dynamic Time Warping (DTW) has proven to be as competitive and even outperform the HMM based methods [36]. DTW focuses on online signature verification using template matching techniques. In [37] an enhanced DTW model that computes the distance between two signatures and assigns special parameters to each signer was proposed. Features extracted include quantized directions, curvature changes, speed, and pressure, trained on the SUSIG database with skilled forgeries. The best result for Receiver Operating Characteristic (ROC) area under curvature gotten was 99.5 with equal error rate of 3.48 %. In [38] an off-line system based on SVM models was presented and compared it to Multi-Layer Perceptrons (MLP). In [39] an offline verification system was proposed that converts the images into time series data, using linear scanning and identifying the time series shapelets. Mahalanobis distance measure is used for comparing two time-series data and a dataset containing 1287 questioned signatures and 646 reference signatures were used and an EER of 5.8% was obtained,it was based on an *interval symbolic representation* and *fuzzy similarity measure*, using a dataset of 16200 offline signature images [40, 41]. Feature extractors are mostly built on specific features. Applying, DL approaches to feature extraction have proven to be more reliable, because of their robustness and ability to learn more features. The inputs of a perceptronx_i: i=1, 2, …, n are multiplied by their corresponding weights $w_i$which is then added to the bias$b$(predetermined weight thatallows modifying the output independent of the input)., forming the outputas inEq. 1:

$$f(x) = \sum_{i=1}^{n} w_i x_i + b$$
$$output = \begin{cases} 0 \; if \;\; f(x) \geq 0 \\ 1 \; if \;\; f(x) < 0 \end{cases} \qquad (1)$$

The bias or threshold influences the output depending on whether it's less than or greater than$w_i x_i$.The activation function determines the output of a node, and helpsDL models separate many forms of data in anNN thus activates the output. It is of two types:*linear activation function*which separates the data in a linear fashion, this works normally in a scenario where the difference between the sorts of data are evident enough to be discovered by the model,*non-linear activation function*that separates the data in a non-linear fashion,which helps the network to adapt to more nuanced patterns in the data that might make it difficult to separate with a linear activation function. *Convolution* is a mathematical operation which expresses how the shape of a function is modified by taking two input functions f and $g$and producing a third function$(f*g)$. CNNs are deep multi-layered NNs that primarily take images as input, capture the spatial and temporal dependencies of that image using layered filters in order to differentiate one image from another [42].CNNs consist of three main layers:

Convolutional Layer (kernel/filter):performs the convolution operation as in Eq. 2 on an input producing a feature map and extracting high-level characteristics from the input image, such as edges. This filter moves according to a parameter called *stride* which determines the pixel or unit value the filter will move across the input at each step. A kernel with a stride length of 1 will move 1 pixel across the input until it reaches the end of the matrix and then move down 1 pixel until it reaches the bottom of the matrix.

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \qquad (2)$$

Pooling Layer: lowers the spatial size of the convolved feature in order to decrease computational power required to process data and extract dominant features that are rotational and positional invariant. There are two types, *maximum pooling* and *average pooling*. The former returns the greatest possible value from a covered portion of input by the kernel while the latter computes the average of all values in the covered portion. Max pooling performs relatively better than average pooling because of its capacity to suppress noise along with dimensionality reduction while average pooling uses dimensionality reduction as a noise suppressing mechanism [43].

Fully-Connected Layer: determines the class of a specific input by using the characteristics discovered via the convolutional and the pooling layer. It converts the outputs from the convolutional and pooling layers from 3D to 1D vector using a method called flattening which arranges the 3D volume of numbers into a 1D vector. This layer consists of a set of multilayer perceptrons along with a softmax activation function for classifying the input according to its learned features and parameters.

Other optional layers in CNN are the *non-linearity* layer which is placed between convolutional and pooling layers in order to pass through an activation function such as the Rectified Linear Unit (ReLU). The *Dropout* layer is used for network regularization, preventing overfitting and increasing the model's accuracy by temporarily disabling a certain percentage of the total neurons at each training-phase iteration.

Model Training and Testing. A155×220 grayscale channel input was chosen. The convolutional and pooling layers are given as (N×H×W) where$N$is the number of filters, H the height, and W the width of the filter. The network is made up of a set of convolutional, pooling, normalization, dropout and fully-connected layers which the image is passed through for feature learning and classification. The image is passed through four convolutional layers. The first layer is made of 96 convolutional kernels, of size 11×11 each, and 1 pixel stride. The second contains 256 kernels, of size 5×5 each, and 2 pixels stride. The third and fourth contain 384 and 256 kernels respectively each of size 3×3, and 1 pixel stride, a padding of 1 pixel and are connected to each other with no layer between them. These layers perform a series of operations on the image called filtering by shifting the kernel across the image, performing a matrix multiplication on each portion of the image in order to produce a feature map and extract high-level features from the image such as edges and vertices. A padding of 1 pixel was added to the second, third and fourth layers of the network. The images are passed through 3 max pooling layers. The first layer consists of 96 filters each of size 3×3 and a stride length of 2. The second and third each consists of 256 filters with size 3×3 and stride length of 2. The pooling layers are designed to reduce the spatial size of the convolved feature in order to decrease computational power required to process data and extract dominant features that are rotational and positional invariant. The image passes through two Local Response Normalization(LRN) layers after the first and second convolutional layers. This along with the Rectified Linear Unit(ReLU) activation function, regularize the network and perform lateral inhibition on the network, which is a process in which a neuron subdues its neighbors and increases sensory perception by creating a contrast in the area. The LRN layer consists of the following parameters:k=2, n=5 (where n is the size), α= $1e^{-5}$, β=0.75.The outputs from the convolutional and pooling layer are moved to the fully-connected through a process called flattening, by converting the vectors from 3D to 1D vectors and arranging them as 1D array vector. The first fully-connected layer consists of 1024 neurons and it's attached to a Dropout layer while the second fully-connected layer consists of 128 neurons which is also the dimension of the most taught characteristic of each side's vector in the network. The Dropout layer is used for network regularization, preventing overfitting and enhancing the precision of the network by temporarily disabling a certain percentage of the total neurons at each iteration of the training phase. The first and second dropout layers in the network are set to a value of 0.3 and the last dropout layer which is connected to the first fully-connected layer is set to a value of 0.5, i.e., 30% of the neurons will be dropped during the first and second dropouts and 50% will be dropped during the final dropout. The architectural parameters are shown on Table 1.To determine whether the model is learning well, it needs to be measured using a loss function so it could be optimized using an optimization algorithm, and adjust the learning rate. This is done to reduce *underfitting* when a model learns too little from the data and fails to perform accurate predictions on the samples, due to the lack of appropriate amount of data or when building a linear model with few non-linear data, andto reduce *overfitting* when a model learns too much from the data and fails to adapt when provided with novel data and inaccurate data entries in the datasetdue to which the model learns too many details and noise from the dataset. *Loss functions* are functions that evaluate how an algorithm is modeling its data by calculating the distance between the expected and current output of the algorithm. The most common loss function is cross-entropy that calculates the probability difference between the distribution functions and determines the output.

Table 1. **The architecture's parameters**

Таблица 1. **Параметры архитектуры**

Layer	Size	Parameters
Convolution	96×11×11	Stride = 1
Local Response Normalisation	–	$\alpha = 10^{-4}, \beta = 0.75$, k=2, n=5
Pooling	96×3×3	Stride = 2
Convolution	256×5×5	Stride = 1, padding = 2
LocalResponseNormalisation	—	α=10–4, β=0.75, k=2, n=5
Pooling + Dropout	256×3×3	Stride = 2, ρ = 0.3
Convolution	384×3×3	Stride = 1, padding = 1
Convolution	256×3×3	Stride = 1, padding = 1
Pooling + Dropout	256×3×3	Stride = 2, ρ = 0.3
Fullyconnectedlayer + Dropout	1024	ρ = 0.5
Fullyconnectedlayer	128	

The input image goes through a series of convolutional layers with ReLU activation function, LRN layers, and max pooling layers, each reducing the number of feature vectors learned by the network with two dropout layers in between before finally going through 2 Fully connected layers containing 1024 and 128 neurons respectively.The outputs from the final fully connected layers are joined by the contrastive loss function.Due to its pairwise learning, there are two main loss functions used for training Siamese networks both of which are distance-based losses: the *triplet loss* that uses a base input to determine the output by comparing it to both a positive (true) and a negative (false) input and minimizing the distance between the base and the positive input and maximizing the distance between the base and the negative input, the *contrastive loss* that calculates the distance between similar and dissimilar input and output pairs of a network by projecting them in a Euclidean space. This means that signatures of the same class(genuine-genuine) would be placed close to each as opposed to signatures of different classes(genuine-forged) which would be placed far from each other on the plane. This distance is then used to predict whether a signature is genuine or forged using a threshold value on the distance. It is mathematically denoted as in Eq. 3:

$$L(s_1, s_2, y)=\alpha(1-y)D^2+\beta y max(0,m-D)^2, \quad (3)$$

where s_1 and s_2 are two signature samples, $m$is the margin(equal to 1 in this instance), y is a binary function indicating if the samples are of the same or different classes. The Euclidean distance $D=\|f(s_1, w_1)-f(s_2, w_2)\|^2$ is computed in the embedded feature space with f being an embedding function mapping a signature image to the real vector space through the CNN and the learned weights (w_1, w_2)of the underlying network. The training algorithm is shown on Fig. 6. Training and saving checkpoint stops when the epoch value becomes equal or greater than 20.

Experiments and results

To implement the signature verification system the following tools were used: Python for the implementation of DL tasks [44, 45], Javascript for the implementation of web demo, Pytorch DL framework, React which is a Javascript frontend library, Google Colab which is a cloud Jupyter notebook, Pillow image library for manipulating different image file formats, Flask web framework, Gunicorn web server gateway interface HTTP server, Scikit Learn learning library for regression, classification etc., Numpy, and Scipy.

During the training, an accuracy of 97.5 % was achieved with possible deviations of around 1-2% depending on the threshold which was computed by taking the average of True positive rate and True negative rate using the ROC. The model was trained using the CEDAR dataset which contains signatures from 55 different signers. Each author has 24 genuine signatures and each forger imitated signatures from 3 authors 8 times, producing 24 forged signatures, altogether making 55×24 = 1320 genuine and forged signatures each. The images were grouped as a pair of genuine and forged where a pair is labeled if both samples in the pair are genuine and came from a single writer and 0 if the samples are from different writers or one of the samples is forged. 13500 image pairs were chosen and split to a train test split of 85% and 15% respectively.

This produced a test set size of about 4100 samples. The training parameters are described in Table 2.

Table 2. **The training parameters**

Таблица 2. **Параметры обучения**

Parameter	Value
Optimizer	Adam [35, 36]
Learning rate	1^{-4}
Learning rate scheduler	$Learning\ rate \times 0.1$
Weight decay	0.0001
Batch Size	16
Epochs	20
Shuffle	True
Step	0.001

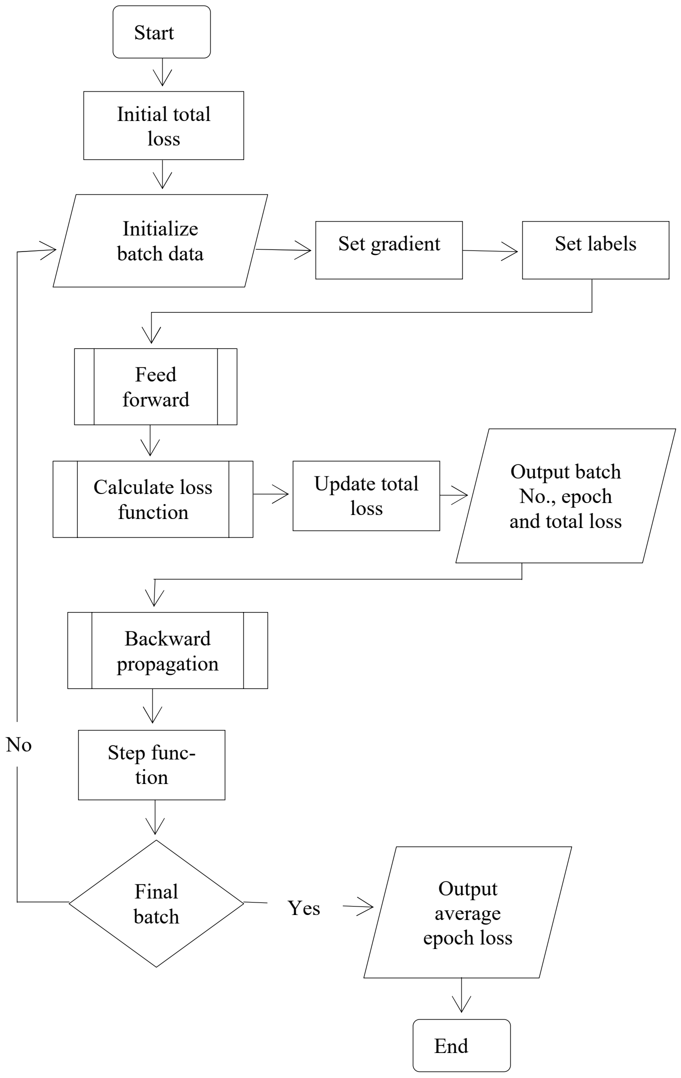


Fig. 6. The model training algorithm

Рис. 6. Алгоритм обучения модели

The accuracy measurement graph is shown on Fig. 7.

The results compared to other works in the literature still proves to be a good result given that it was trained on a significantly small dataset and training period of 20 epochs.This is also among the few end to end approaches in the literature where no manual feature extractors were needed and the network learned all the features needed for the classification on its own.Table 3 shows a comparison with other works in the literature, some of which used manual feature extractors and some of which used a similar approach of automatic feature learning.

Sample results obtained bythe signature verification system are shown on Fig. 8.

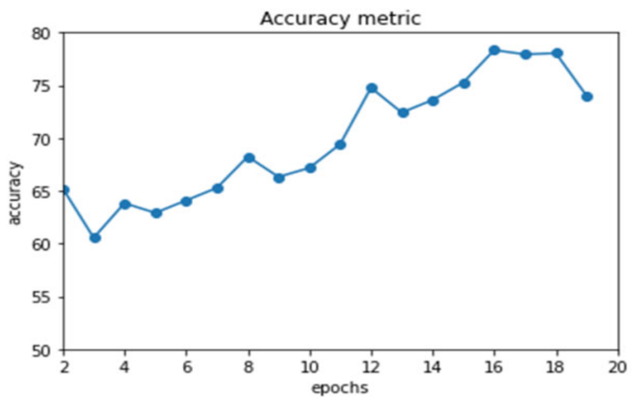


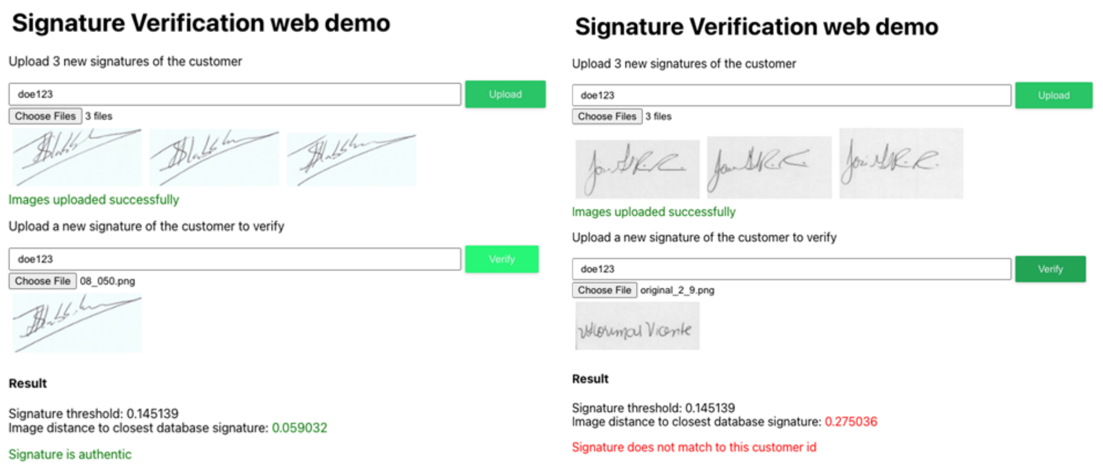
Fig. 7. Acc uracy metric

Рис. 7. Метрика точности

Table 3. **Comparison with similar work**

Таблица 3. **Сравнение с аналогичной работой**

Authors	Approach	Accuracy (%)
Gabe et al.[28]	CNN	Dutch: 97, Chinese: 95
Alajrami et al.[29]	ANN	60-40 split: 99.7, 70-30 split: 98 80-20 split: 99.7
Anamikaetal. [30]	ANN	Hindi: 95.29, Bengali: 97.79
Martinezetal. [38]	SVMs and MLP	SVM(characteristic): 66.5 SVM(bitmap): 71.2 MLP, (characteristic): 45.2 MLP(bitmap): 46.8
Albasu et al. (This work)	CSNN	97.5


Signature Verification web demo

Upload 3 new signatures of the customer

doe123

Choose Files 3 files

Images uploaded successfully

Upload a new signature of the customer to verify

doe123

Choose File 08_050.png

Result

Signature threshold: 0.145139
Image distance to closest database signature: 0.059032

Signature is authentic

Signature Verification web demo

Upload 3 new signatures of the customer

doe123

Choose Files 3 files

Images uploaded successfully

Upload a new signature of the customer to verify

doe123

Choose File original_2_9.png

Result

Signature threshold: 0.145139
Image distance to closest database signature: 0.275036

Signature does not match to this customer id

Fig. 8. The results of signature verification

Рис. 8. Результаты проверки подписи

Conclusion

This work presents a writer-independent handwritten-signatureverification system which is developed using CSNNArchitecture. Unlike most other works in the literature that use handcrafted feature extractors, the model in the experiment learns from the data fed into it in a writer-independent scenario. The model has shown a pretty positive result given the tight budget and resources, and can be improved further to obtain even better results if provided better and more efficient resources. Theadvantage of this system lies in the possibility of usingit in many tasks such as user verification, fraud detection and prevention, and forensic investigation and can be integrated with systems in different applications such as banking, travel, legal and more. Theadopted approach aims to improve writer-independent signature verification. There have been alternatives proposed to circumvent the issue by training writer-independent feature extractors and writer-dependent classifiers but only a few works use writer-independent classifiers.

References

1. Hafemann, L. G., et al. (2017). *Offline Handwritten Signature Verification — Literature Review*. Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA), IEEE, pp. 1–8. URL: https://doi.org/10.1109/IPTA.2017.8310112.

2. Khalil, M. I., et al. (2009). *Enhanced DTW Based On-Line Signature Verification*. 16th IEEE International Conference on Image Processing (ICIP), IEEE, pp. 2713–2716. URL: https://doi.org/10.1109/ICIP.2009.5414166.

3. Arathi, M., and Govardhan, A.(2014). *An Efficient Offline Signature Verification System*. International Journal of Machine Learning and Computing, vol. 4, no. 6, pp. 533–537. URL: https://doi.org/10.7763/IJMLC.-2014.V6.468.

4. *Alajrami*, E., et al. (2020). *Handwritten Signature Verification Using Deep Learning*. International Journal of Academic Multidisciplinary Research (IJAMR), vol. 3, то. 12, pp. 39-44.

5. Alaei, A., et al. (2017). *An Efficient Signature Verification Method Based on an Interval Symbolic Representation and a Fuzzy Similarity Measure*. IEEE Transactions on Information Forensics and Security, vol. 12, no. 10, pp. 2360–2372. URL: https://doi.org/-10.1109/TIFS.2017.2707332.

6. Anamika, J., et al. (2021). *Signature Verification Using Geometrical Features and Artificial Neural Network Classifier*. Neural Computing and Applications, vol. 33, no. 12, pp. 6999–7010. *arXiv.org*. URL: https://doi.org/10.1007/s00521-020-05473-7.

7. Ippolito, P. P. (2019). *Feature Extraction Techniques*. Mediumю URL: https://towardsdatascience.com/-feature-extraction-techniques-d619b56e31be.

8. Donato, I. and Pirlo, G. (2008). *Automatic Signature Verification: The State of the Art*. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 38, no. 5, pp. 609–35. URL: https://doi.org/10.1109/TSMCC.2008.923866.

9. Zhang, Z., et al. (2016). *Multi-Phase Offline Signature Verification System Using Deep Convolutional Generative Adversarial Networks*.9th International Symposium on Computational Intelligence and Design (ISCID), vol. 2, pp. 103–107. *IEEE Xplore*ю URL: https://doi.org/10.1109/ISCID.2016.2033.

10. Eskander, G. S., et al. (2013). *Hybrid Writer-independent–Writer-dependent Offline Signature Verification System*. IET Biometrics, vol. 2, no. 4, pp. 169–181. URL: https://doi.org/10.1049/iet-bmt.2013.0024.

11. Yılmaz, M. B., and Berrin Y. (2016) *Score Level Fusion of Classifiers in Off-Line Signature Verification*.Information Fusion, vol. 32, pp. 109–19. URL: https://doi.org/10.1016/j.inffus.2016.02.003.

12. Alvarez, G., et al. (2016). *Offline Signature Verification with Convolutional Neural Networks*. Technical report, Stanford University.

13. Justino, E. J. R., et al. (2002). *An Off-Line Signature Verification System Using HMM and Graphometric Features*. 4th IAPR International Workshop on Document Analysis Systemsp. pp. 211-222.

14. Faúndez-Zanuy, M. (2007). *On-Line Signature Recognition Based on VQ-DTW*. Pattern Recognition, vol. 40, no. 3, pp. 981-992. URL: https://doi.org/10.1016/j.patcog.2006.06.007.

15. Frias-Martinez, E., et al. (2006). *Support Vector Machines versus Multi-Layer Perceptrons for Efficient off-Line Signature Recognition*. Engineering Applications of Artificial Intelligence, vol. 19, no. 6, pp. 693–704. URL: https://doi.org/10.1016/j.engappai.2005.12.006.

16. Chavan, M., et al. (2017). *Handwritten Signature Verification Using Hidden Markov Model with Hybrid Wavelet Transform*. International Conference on Computing, Communication, Control and Automation (ICCUBEA), pp. 1–6. *IEEE Xplore*ю URL: https://doi.org/10.1109/ICCUBEA.2017.8463675.

17. Farimani, S. A., and Majid V. J. (2018). *An HMM for Online Signature Verification Based on Velocity and Hand Movement Directions*. 6th Iranian Joint Congress on Fuzzy and Intelligent Systems *(CFIS)*, pp. 205–09. *IEEE Xplore*ю URL: https://doi.org/10.1109/CFIS.2018.8336639.

18. Saha, S.(2018). *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 Way*. Mediumю URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

19. Hafemann, L. G., et al. (2016). *Writer-Independent Feature Learning for Offline Signature Verification Using Deep Convolutional Neural Networks*. International Joint Conference on Neural Networks (IJCNN), pp. 2576–83. *arXiv.org*ю URL: https://doi.org/10.1109/IJCNN.2016.7727521.

20. Oliveira, L. S., et al. *The Graphology Applied to Signature Verification*. 12th Conf. Int. Graphonomics Soc. (IGS2005), pp. 286-290.

21. Sangekar, S. S., and Dhanwani D. C. (2012).*Survey of Various Techniques for Signature Recognition and Verification*. International Journal of Science and Research (IJSR), vol. 3,no. 11, pp. 1520-1522.

22. Dey, S., et al. (2017). *SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification*. arXiv:1707.02131ю URL: http://arxiv.org/abs/1707.02131.

23. Koch, G., et al. (2015). *Siamese Neural Networks for One-Shot Image Recognition*. ICML deep learning workshop, vol. 2.

24. Karouni, A., et al. (2011). *Offline Signature Recognition Using Neural Networks Approach*. Procedia Computer Science, vol. 3, pp. 155–61. URL: https://doi.org/10.1016/j.procs.2010.12.027.

25. Pal, S., et al. (2011). *Off-Line Signature Verification Systems: A Survey*. Proceedings of the International Conference & Workshop on Emerging Trends in Technology – ICWET'11, ACM Press, pp. 652–657. URL: https://doi.org/10.1145/1980022.1980163.

26. Justino, E. J. R., et al. (2001). *Off-Line Signature Verification Using HMM for Random, Simple and Skilled Forgeries*. Proceedings of Sixth International Conference on Document Analysis and Recognition, IEEE Comput. Soc, pp. 1031-1034. URL: https://doi.org/10.1109/ICDAR.2001.953942.

27. Hafemann, L. G., et al. (2017). *Learning Features for Offline Handwritten Signature Verification Using Deep Convolutional Neural Networks*. Pattern Recognition, vol. 70, pp. 163–176. *arXiv.org*ю URL: https://doi.org/10.1016/j.patcog.2017.05.012.

28. Krizhevsky, A., et al. (2017). *ImageNet Classification with Deep Convolutional Neural Networks*. Communications of the ACM, vol. 60, no. 6, pp. 84–90. URL: https://doi.org/10.1145/3065386.

29. Hafemann, L. G., et al. (2018). *Fixed-Sized Representation Learning from Offline Handwritten Signatures of Different Sizes*. International Journal on Document Analysis and Recognition (IJDAR), vol. 21, no. 3, pp. 219–32. *arXiv.org*. URL: https://doi.org/10.1007/s10032-018-0301-6.

30. Hadsell, R., et al. (2006). *Dimensionality Reduction by Learning an Invariant Mapping*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 2, pp. 1735–1742. https://doi.org/10.1109/CVPR.2006.100.

31. Madasu, Vamsi K., et al. *Automatic Handwritten Signature Verification System for Australian Passports*. Science, Engineering and Technology Summit on Counter-Terrorism Technology, pp. 53-66.

32. Odeh, S., and Khalil, M.(2011). *Apply Multi-Layer Perceptrons Neural Network for Off-Line Signature Verification and Recognition*. International Journal of Computer Science Issues (IJCSI),vol. 8, no. 6, pp. 261-266.

33. Shah, A., et al. (2016). *An Offline Signature Verification Technique Using Pixels Intensity Levels*. International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 9, no. 8, pp. 205-22. URL: https://doi.org/10.14257/ijsip.2016.9.8.18.

34. Benhur, S. J. (2021). *A Friendly Introduction to Siamese Networks*. Medium. URL: https://towardsdatascience.com/a-friendly-introduction-to-siamese-networks-85ab17522942.

35. Kingma, D. P., and Ba, J.(2017). *Adam: A Method for Stochastic Optimization*. arXiv:1412.6980. URL: http://arxiv.org/abs/1412.6980.

36. Sanghvirajit. (2021). *A Complete Guide to Adam and RMSprop Optimizer*.Analytics Vidhya. URL: https://medium.com/analytics-vidhya/a-complete-guide-to-adam-and-rmsprop-optimizer-75f4502d83be.

37. Kang, N. (2019). *Introducing Deep Learning and Neural Networks – Deep Learning for Rookies (1)*. Medium. URL: https://towardsdatascience.com/introducing-deep-learning-and-neural-networks-deep-learning-for-rookies-1-bd68f9cf5883.

38. Bindal, A. (2019). *Normalization Techniques in Deep Neural Networks*. Techspace. URL: https://medium.com/techspace-usict/normalization-techniques-in-deep-neural-networks-9121bf100d8.

39. Bertolini, D., et al. (2010). *Reducing Forgeries in Writer-Independent off-Line Signature Verification through Ensemble of Classifiers*. Pattern Recognition, vol. 43, no. 1, pp. 387-96. URL: https://doi.org/10.1016/j.patcog.2009.05.009.

40. Vinushanth, C. V. (2020). *Markov and Hidden Markov Model*. Medium. URL: https://towardsdatascience.com/markov-and-hidden-markov-model-3eec42298d75.

41. Manohar, S. (2017). *Mastering Machine Learning with Python in Six Steps: A Practical Implementation Guide to Predictive Data Analytics Using Python*. Apress.

42. Sandipan, D. (2018). *Hands-on Image Processing with Python: Expert Techniques for Advanced Image Analysis and Effective Interpretation of Image Data*. Packt Publishing. Open WorldCat. URL: http://proquestcombo.safaribooksonline.com/9781789343731.

43. Ansari, A., et al. (2014). *Online signature verification using segment-level fuzzy modeling*. IET biometrics,vol. 3, no. 3, pp. 113-127.

44. Masoudnia, S., et al. (2019). *Multi-Representational Learning for Offline Signature Verification Using Multi-Loss Snapshot Ensemble of CNNs*. Expert Systems with Applications, vol. 133, pp. 317-30. *arXiv.org*. URL: https://doi.org/10.1016/j.eswa.2019.03.040.

45. Sarker, I. H. (2021). *Machine Learning: Algorithms, Real-World Applications and Research Directions*. SN Computer Science, vol. 2, no. 3, p. 160. Springer Link. URL: https://doi.org/10.1007/s42979-021-00592-x.

* * *

Использование методов глубокого обучения для верификации подписей

Ф. Б. Албасу, студент, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия
М. А. Аль Аккад, кандидат технических наук,доцент, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия

Биометрические характеристики являются распространенными мерами проверки личности, где наиболее часто используются подписи. Цифровые технологии породили новые способы биометрической идентификации, такие как отпечатки пальцев, радужная оболочка глаза и распознавание лиц, в то время как работа с рукописными подписями по-прежнему остается сложной задачей, поскольку рукописные подписи более подвержены подделке, чем другие средства проверки, из-за таких проблем, как компьютерная ошибка, недостаточный набор данных и потеря информации. Целью этой работы является разработка системы, которая использует изображение подписи в качестве входных данных и определяет, является ли подпись подлинной, написанной ее автором, или подделана другим лицом. Система основана на алгоритме нейронной сети под названием «сверхточные сиамские нейронные сети», которые используются для глубокого обучения и компьютерного зрения, а также для других задач машинного обучения, таких как обработка естественного языка и цифровых сигналов. Используется функция контрастных потерь, которая сравнивает евклидово расстояние выходных векторов признаков, а для обучения и классификации изображений используется независимая от записи модель. Цель этой работы состоит в том, чтобы повысить точность проверки подписи и взять ее за основу для будущей работы по проверке подписей и использовать в приложениях для идентификации пользователей, обнаружения и предотвращения мошенничества, а также для проведения судебно-медицинских расследований. Система может быть применена в банковских, государственных и частных организациях, а также в судебно-медицинской экспертизе для проверки личности и документов, выявления и предотвращения мошенничества, преступлений и судебных расследований, а также проверки паспортов.

Ключевые слова: биометрическая идентификация, распознавание подписи, проверка подписи, аутентификация, глубокое обучение.

Получено: 17.18.23

Образец цитирования

Albasu F. B., *Al Akkad M. A.*, Exploiting Deep Learning Techniques for the Verification of Handwritten Signatures // Интеллектуальные системы в производстве. 2023. Т. 21, № 3. С. 27-39. DOI: 10.22213/2410-9304-2023-3-27-39.

For Citation

Albasu F.B., *Al Akkad M.A.* [Exploiting Deep Learning Techniques for the Verification of Handwritten Signatures]. *Intellektual'nye sistemy v proizvodstve.* 2023, vol. 21, no. 3, pp. 27-39. DOI: 10.22213/2410-9304-2023-3-27-39.