Application of Convolutional Neural Networks in Biometric Identification Problems

Liubomyr Monastyrskyi, Yaroslav Boyko, Volodymyr Lozynskyi, Taras Kropyvka
Department of Radioelectronic and Computer Systems
Faculty of Electronics and Computer Technologies
Ivan Franko National University of Lviv
Lviv, Ukraine
yaroslav.boyko@lnu.edu.ua

Abstract—**The prototype of a neural network system of biometric identification based on fingerprints using a convolutional neural network is implemented. An important feature of the proposed methodology is the use of a very limited set of biometric data to train the neural network. The results of the calculations show that the use of carefully selected neural network hyperparameters and data augmentation methods makes it possible to apply the developed prototype for practical purposes.**

Index Terms—**Machine learning; convolutional neural networks; biometrical identification.**

I. Introduction

Biometrics is called the set of technologies for the quantitative description of the bodies of humans or animals. In recent decades computer biometric systems have become widely used in security systems for identification, authorization, access control and verification [1].

Although biometric systems offer solutions to many authentication and identification problems, biometric systems themselves are complex, and a variety of factors affect their ability to function successfully. The use of specific biometric techniques often depends to a large extent on the application or conditions in which it will be used. For example, fingerprint recognition is an ideal choice for use on mobile devices and access control systems, but has significant limitations for remote or hidden identity. There is also a large variety of other factors that need to be considered, such as the presence of biometric characteristics in the population under study, the uniqueness of the sample, the time period of stability, ease of use, economic feasibility and resistance to attacks. No biometric technology has an ideal set of characteristics, therefore careful selection, design and implementation of advanced systems is required to ensure success [1].

In the previous period, considerable progress was made in applying traditional mathematical methods to the development of computer systems for biometric identification. However, as shown in [2] on the example of fingerprint recognition, a variety of image acquisition methods leads to problems of unification of image recognition methods, that is, it imposes significant limitations on the possibility of quantifying the characteristic properties of data sets. Therefore, an extremely important area of research in this field is the development of unlimited methods for processing biometric information, which would have a high degree of universality. In [2], it is convincingly shown that methods of deep machine learning, such as those that have proven their unmatched accuracy and flexibility, should first be attributed to such methods. The first section [2] gives a general overview of the methods of machine learning, which are widely used in biometric studies and practical implementations, and the subsequent sections cover the results of specific studies in this field. A characteristic feature of almost all work on the use of machine learning in biometrics is the standard training process on large data sets. This approach is fully methodically grounded, but in some cases it is difficult to implement. In particular, the creation of fingerprint databases for the purpose of identifying a person does not usually involve the receipt of a large number of prints of one person. However, one can hope that the use of perfect algorithms of machine learning will allow and in such tasks achieve sufficient for practical use of the level of accuracy and reproducibility of the results. The purpose of this work is to determine the possibility of using convolutional neural networks (CNN) for the purpose of biometric identification under the conditions of limited inputs on an example of fingerprint recognition.

II. Methods

As indicated in [3], despite the great efforts in computer vision community, the hand-engineered features were not able to properly model large classes of natural objects. Advent of CNN, large datasets and parallel computing hardware changed the course of computer vision. Instead of designing feature vectors by hand, convolutional neural networks learn a composite feature transformation function that makes classes of objects linearly separable in the feature space. CNN or ConvNet is a class of deep, feed-forward artificial neural networks, most commonly applied to analyzing visual imagery. Convolutional networks were inspired by biological processes [4] in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field. CNNs use relatively little pre-processing compared to other image classification algorithms. This means

that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage. They have applications in image and video recognition, recommender systems and natural language processing [4]. A typical schematic representation of CNN is shown in the Figure 1.

A high-level library Keras was used to implement the custom CNN [5]. Keras is written in Python and capable of running on top of TensorFlow, CNTK, or Theano libraries. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research. Keras allows for easy and fast prototyping (through user friendliness, modularity, and extensibility), supports both convolutional networks and recurrent networks, as well as combinations of the two, runs seamlessly on CPU and GPU.

We used TensorFlow library as a backend in our calculations [6]. TensorFlow is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. Originally developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with strong support for machine learning and deep learning and the flexible numerical computation core is used across many other scientific domains.

TensorFlow uses computational data-flow graphs to represent complicated neural-network architecture. The nodes in the graph denote mathematical computations, also called ops (operations), whereas the edges denote the data tensors transferred between them. Also, the relevant gradients are stored at each node of the computational graph, and during backpropagation these are combined to get the gradients with respect to each weight. Tensors are multi-dimensional data arrays used by TensorFlow [7].

Fundamentally, TensorFlow C++ engine consists of following two things:

• Efficient implementations for operations like convolution, max pool, sigmoid, and so on.

• Derivatives of forwarding mode operation.

TensorFlow graph has two types of edges:

• Normal: They carry the data structures between the nodes. The output of one operation from one node, becomes input for another operation.The edge connecting two nodes carries the values.

• Special: This edge doesn't carry values, but only represents a control dependency between two nodes, say X and Y. It means that the node Y will be executed only if the operation in X is executed already, but before the relationship between operations on the data.

The TensorFlow implementation defines control dependencies to enforce orderings between otherwise independent operations as a way of controlling the peak memory usage [8].

III. Results and discussion

As input to the training of the neural network, a fragment of the biometric information database CASIA Fingerprint Image Database Version 5.0 [9] was used. As previously mentioned, the purpose of the study was to determine the possibility of classifying fingerprint images based on training on a small number of images. The size of the input vector for training was 30 images, validation - 10 images. To realize this goal, a convolutional neural network was created, the structure of which is depicted by the output of the Keras model.summary() function:

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 118, 148, 32)	320
activation_1 (Activation)	(None, 118, 148, 32)	0
max_pooling2d_1 (MaxPooling2	(None, 59, 74, 32)	0
conv2d_2 (Conv2D)	(None, 57, 72, 64)	18496

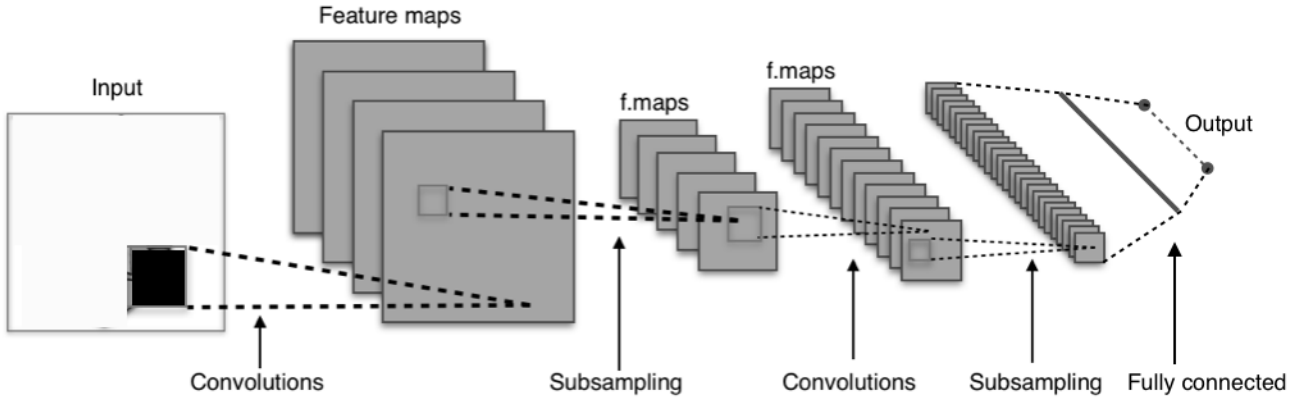

Figure 1. Typical schematic representation of CNN

activation_2 (Activation)	(None, 57, 72, 64)	0
max_pooling2d_2 (MaxPooling2	(None, 28, 36, 64)	0
conv2d_3 (Conv2D)	(None, 26, 34, 128)	73856
activation_3 (Activation)	(None, 26, 34, 128)	0
conv2d_4 (Conv2D)	(None, 24, 32, 256)	295168
activation_4 (Activation)	(None, 24, 32, 256)	0
max_pooling2d_3 (MaxPooling2	(None, 12, 16, 256)	0
flatten_1 (Flatten)	(None, 49152)	0
dense_1 (Dense)	(None, 128)	6291584
activation_5 (Activation)	(None, 128)	0
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 2)	258
activation_6 (Activation)	(None, 2)	0

Total params: 6,679,682
Trainable params: 6,679,682
Non-trainable params: 0

As can be seen, the neural network consists of four convolutional layers, which have the appropriate rectified linear units and pooling operations. In addition, at the output of the network, there are two dense (fully connected) layers for the binary classification

In order to improve the accuracy of the training of the neural network, methods of the ImageDataGenerator class that implement data augmentation were applied: sample-wise standardization, feature-wise standardization, random rotation, shifts, shear and flips, dimension reordering.

In the process of training the network used the Adaptive Moment Estimation (Adam) algorithm [5] that computes adaptive learning rates for all parameters. In order to reduce the overfitting effect of the network, the L2-regularization procedure is applied [5].

The learning process of the created neural network is shown in Figures 2, 3. In particular, Figure 2 shows the dependence of the values of the loss function (upper part) and the accuracy of the classification (lower part of the figure) during the training of the neural network. One can see that, despite the critically small amount of input data, it was possible to achieve acceptable level of accuracy for the classifier for practical applications. The work of the convolutional network is also illustrated with a drawing 3. It shows a map of properties obtained as a result of the action of the second convolutional layer. The use of a set of filters allows you to

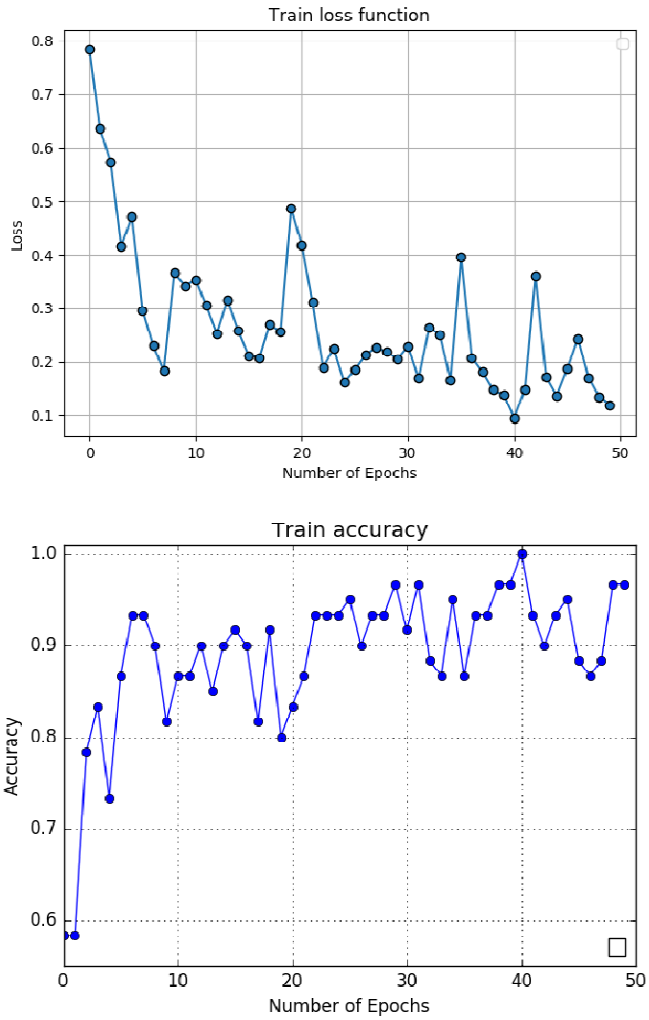

Figure 2..Loss function and accuracy during the training process

distinguish from the input image the set of features that will be classified.

The confusion matrix is a yet another convenient way to present and evaluate the correctness of machine learning algorithms. Confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa). The name stems from the fact that it makes it easy to see if the system is confusing two classes (i.e. commonly mislabeling one as another) [10].

Figure 4 presents a confusing matrix with the results of our research. You can make sure that the well-chosen hyper-parameters of CNN allow you to get reliable results.

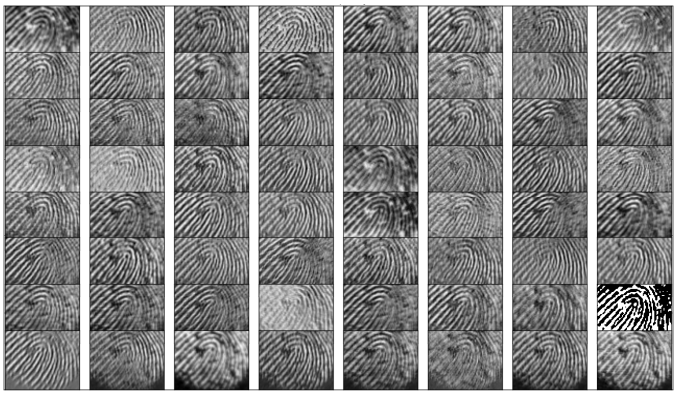

Figure 3. Feature matrix at the convolutional layer No. 2

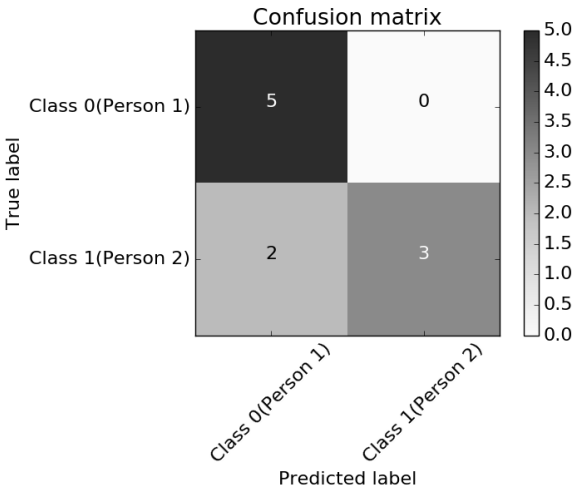

Figure 4. Confusion matrix based on the results of prediction process

IV. CONCLUSIONS

On the basis of the obtained results it is possible to make unambiguous conclusions about the considerable potential of convolutional neural networks in solving important problems of biometrics and security systems. In particular, it has been shown that on the basis of a small set of biometric data it is possible to construct a realistic biometric identification system that is relatively simple to implement. Such a system can be combined, namely: training is carried out on relatively powerful computing systems (servers, clusters), and the classification is carried out on systems with limited resources (microcomputers, microcontrollers). | In addition, in order to increase the authenticity of identification, it is expedient to implement multivariate verification based on different biometric data.

REFERENCES

[1] Information Security Foundations, Technologies and Applications. Edited by Ali Ismail Awad and Michael Fairhurst. The Institution of Engineering and Technology, 2018, 404 p.

[2] Deep Learning in Biometrics. Edited by Mayank Vatsa Richa Singh Angshul Majumdar. CRC Press, 2018, 316 p.

[3] Hamed Habibi Aghdam, Elnaz Jahani Heravi, Guide to Convolutional Neural Networks. A Practical Application to Traffic-Sign Detection and Classification. Springer, 2017, 282 p.

[4] Convolutional Neural Networks [Electronical resource]. – Access mode: https://en.wikipedia.org/wiki/Convolutional_neural_network

[5] Keras Library [Electronical resource]. – Access mode: https://keras.io/

[6] TensorFlow Library [Electronical resource]. – Access mode: https://www.tensorflow.org/

[7] Santanu Pattanayak. Pro Deep Learning with TensorFlow. A Mathematical Approach to Advanced Artificial Intelligence in Python. Apress, 2017, 398 p.

[8] Md. Rezaul Karim. TensorFlow: Powerful Predictive Analytics with TensorFlow. Packt Publishing, 2018, 152 p.

[9] CASIA Database [Electronical resource]. – Access mode: http://biometrics.idealtest.org/findTotalDbByMode.do?mode=Fingerprint

[10] Stephen V.Stehman. Selecting and interpreting measures of thematic classification accuracy. Remote Sensing of Environment Volume 62, Issue 1, October 1997, pp 77–89