A New Method for Face Recognition Using Convolutional Neural Network

Patrik KAMENCAY, Miroslav BENCO,
Tomas MIZDOS, Roman RADIL

Department of Multimedia and Information-Communication Technologies, Faculty of Electrical Engineering,
University of Zilina, Univerzitna 8215/1, 010 26 Zilina, Slovakia

patrik.kamencay@fel.uniza.sk, miroslav.benco@fel.uniza.sk, tomas.mizdos@fel.uniza.sk,
roman.radil@fel.uniza.sk

DOI: 10.15598/aeee.v15i4.2389

Abstract. *In this paper, the performance of the proposed Convolutional Neural Network (CNN) with three well-known image recognition methods such as Principal Component Analysis (PCA), Local Binary Patterns Histograms (LBPH) and K–Nearest Neighbour (KNN) is tested. In our experiments, the overall recognition accuracy of the PCA, LBPH, KNN and proposed CNN is demonstrated. All the experiments were implemented on the ORL database and the obtained experimental results were shown and evaluated. This face database consists of 400 different subjects (40 classes/ 10 images for each class). The experimental result shows that the LBPH provide better results than PCA and KNN. These experimental results on the ORL database demonstrated the effectiveness of the proposed method for face recognition. For proposed CNN we have obtained a best recognition accuracy of 98.3 %. The proposed method based on CNN outperforms the state of the art methods.*

Keywords

Face recognition system, KNN, LBPH, neural networks, PCA.

1. Introduction

The idea of face recognition [1] is to give a computer system the ability of finding and recognizing human faces fast and precisely in images or videos. Numerous algorithms and techniques have been developed for improving the performance of face recognition. Recently Deep learning has been highly explored for computer vision applications. Human brain can automatically and instantly detect and recognize multiple faces. But when it comes to computer, it is very difficult to do all the challenging tasks on the level of human brain.

The face recognition is an integral part of biometrics. In biometrics, basic traits of human are matched to the existing data. Facial features are extracted and implemented through algorithms, which are efficient and some modifications are done to improve the existing algorithm models. Computers that detect and recognize faces could be applied to a wide variety of practical applications including criminal identification, security systems, identity verification etc. The face recognition system [1] generally involves two stages:

- Face Detection – where the input image is searched to find any face, then image processing cleans up the facial image for easier recognition.
- Face Recognition – where the detected and processed face is compared to the database of known faces to decide who that person is.

The difference between face detection and recognition is that in detection we just need to determine if there is some face in the image, but in recognition we want to determine whose face it is. Features extracted from a face are processed and compared with similarly processed faces present in the database [1] and [2]. In general, face recognition techniques can be divided into two groups:

- Face representation techniques – these techniques use holistic texture features and are applied to either whole-face or specific regions in a face image.
- Feature-based techniques – these techniques use geometric facial features (mouth, eyes, brows, etc.), and geometric relationships between them.

The outline of this paper is organized as follows. In the Sec. 2. , the face recognition system is discussed. The obtained experimental results are listed in Sec. 3. Finally, the Sec. 4. concludes and suggests the future work.

2. Face Recognition System – State of the Art

The process of the face recognition starts with two image sets: the gallery, and the reference set. The gallery consists of frontal face images. The goal is to rank the gallery images based on their similarity to a given probe image (see Fig. 1).

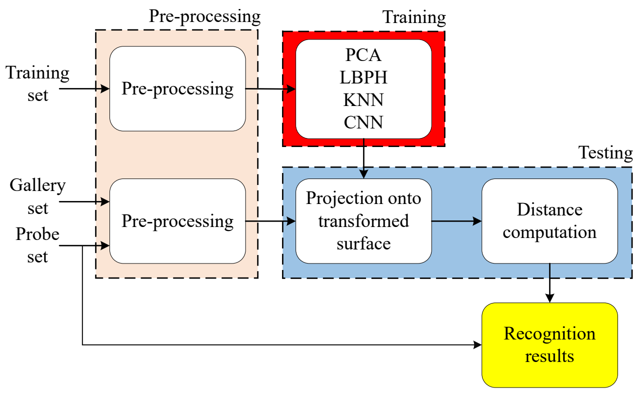

Fig. 1: Example of the face recognition system.

After pre-processing, the reference-based descriptors for the gallery images are computed by measuring similarity between the gallery and reference set feature vectors. The face recognition algorithms have to deal with significant amounts of illumination variations between gallery and probe images. The reference-based descriptor for the probe image is generated by computing the similarity between the probe and reference set images. The gallery images are ranked based on the similarity scores between reference-based descriptors of the probe and gallery images. The concepts of gallery and probe sets are defined as follows:

- Each probe image is matched against those in a gallery, and the ranked matches can be analyzed to produce recognition performance measures such as the cumulative match score for identification, and the receiver operating characteristic for verification applications.
- Projections are extracted from all the samples in the gallery and the probe set.
- The scores between each gallery sample and probe is defined as the distance between two projections after alignment.

Interestingly, many traditional computer vision image classification algorithms follow this pipeline (see Fig. 1), while Deep Learning based algorithms bypass the feature extraction step completely [1], [2], [3], [4], [5], [6], [7], [8] and [9].

In all our experiments, the feature extraction (PCA, LBPH) and classifications (KNN and proposed CNN) methods have been used to predict test face images.

2.1. Face Recognition Using Eigenfaces

The Eigenfaces is the name given to a set of Eigenvectors when they are used in the computer vision problem of human face recognition [2]. It involves pixel intensity features and uses the Principal Component Analysis (PCA) of the distribution of faces, or Eigenvectors, which are a kind of set of features characterizing faces variations where each face image contributes more or less to each eigenvector (see Fig. 2). Thus, an Eigenvector can be seen as an Eigenface. The Eigenfaces themselves form a basic set of all images used to construct the covariance matrix. This produces dimension reduction by allowing the smaller set of basis images to represent the original training images.

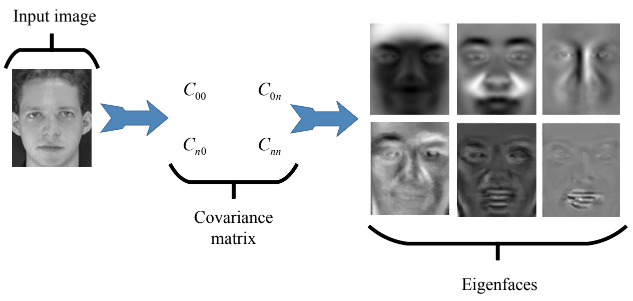

Fig. 2: Block diagram of a PCA algorithm.

The general steps for performing the Eigenfaces [2]:

- First, insert a set of images into a database (these images are called the training set).
- Second, compute covariance matrix of input face images.
- Next create the Eigenfaces. In our case, the Eigenfaces are extracted from the image data using a PCA [3].
- When the Eigenfaces have been created, each image is represented as a vector of weights (represent all face images in the dataset as linear combination of Eigenfaces).
- Finally, the weight of the unknown image is found and then compared to the weights of those already

in the system. If the input image weight is over a given threshold it is considered to be unknown. The identification of the input image is done by finding the image in the database, the weights of which are the closest to the weights of the input image.

Classification can be achieved by comparing how faces are represented by the basis set.

2.2. LBP Approach to Face Recognition

The Local Binary Patterns (LBP) is a texture descriptor that can also be used to represent faces, since a face image can be seen as a composition of micro-texture-patterns. Briefly, the procedure consists of dividing a facial image into several regions where the LBP features are extracted and concatenated into a feature vector that will be used as facial descriptor [4] later.

The LBP originally appeared as a generic texture descriptor. The operator assigns a label to each pixel of an image by thresholding a 3×3 neighborhood with the center pixel value and considering the result as a binary number. In this research, the binary result has been obtained by reading the values clockwise, starting from the top left neighbor, as can be seen in the following figure.

In other words, given a pixel position $(x,\ y)$, LBP is defined as an ordered set of binary comparisons of pixel intensities of the central pixel and its surrounding pixels. The resulting decimal label value of the 8-bit word can be expressed as follows [4] and [5]:

$$LBP(x, y) = \sum_{n=0}^{7} 2^n \cdot s(l_n(x, y) - l_c(x, y)), \qquad (1)$$

where l_c corresponds to the grey value of the center pixel $(x,\ y)$, l_n to the grey values of the 8 surrounding pixels, and function $s(k)$ is defined as:

$$s(k) = \begin{cases} 1 & \text{if } k \geq 0, \\ 0 & \text{if else.} \end{cases} \qquad (2)$$

The LBP operator works with the eight neighbors of a pixel, using the value of this center pixel as a threshold [6]. If a neighbor pixel has a higher gray value than the center pixel (or the same gray value) than one is assigned to that pixel, else it gets zero. The LBP code for the center pixel is then produced by concatenating the eight ones or zeros to a binary code (see Fig. 3).

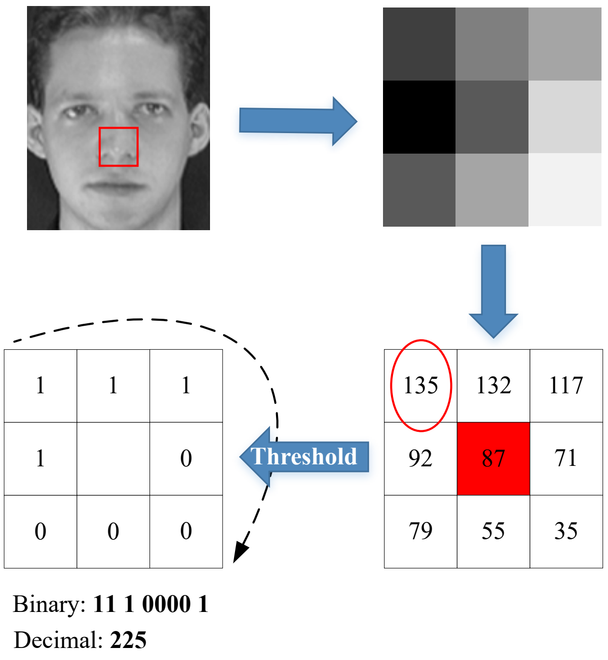

Fig. 3: Example of a LBP calculation (feature extraction).

During the training stage (see Fig. 4), samples of faces and non-faces generate one unique matrix of eigenvectors that will represent the feature space (LBP space). Then for each class (faces and non-faces) one representative model is selected (e.g. the mean of all training samples of each class). These both models are projected to the LBP space [7].

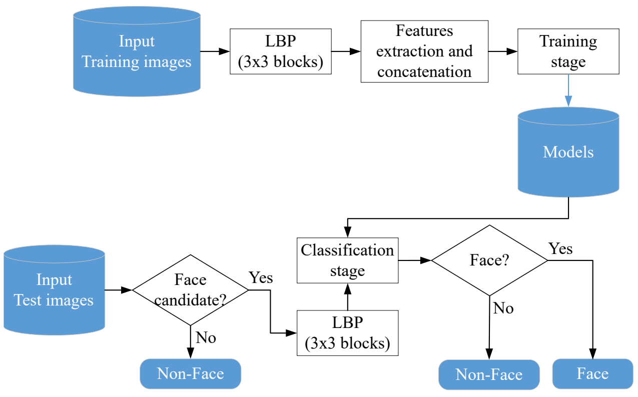

Fig. 4: Block diagram of a LBP algorithm.

In the test stage (see Fig. 4), a feature vector on a new input test sample (if given) is projected to the orthonormal basis. Then, projected version of the input sample is compared with the two projected models (faces and non-faces). The minimum distance provides the shape of the input vector [7].

2.3. K – Nearest Neighbour Classifier

The K–Nearest Neighbour classifier is by far the simplest machine learning/image classification algorithm. Inside, this algorithm simply relies on the distance between feature vectors. Simply put, the K–NN algorithm classifies unknown data points by finding the most common class among the K–closest examples. Each data point in the K-closest examples casts a vote and the category with the most votes wins [8].

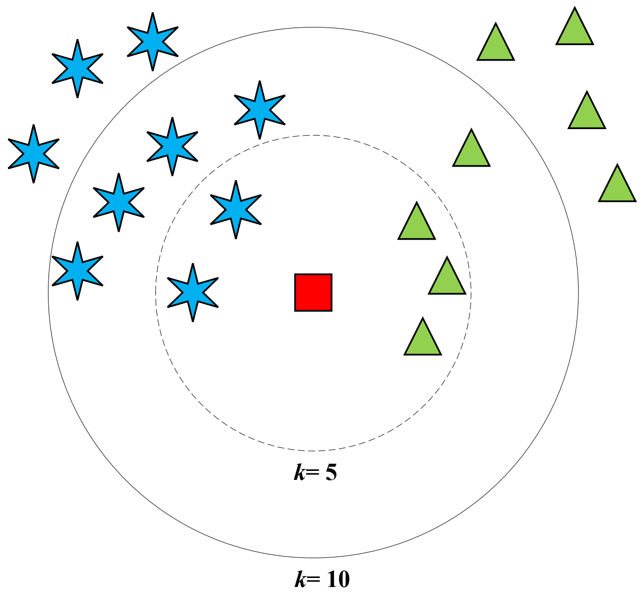

Fig. 5: The principle diagram of the K-NN classification algorithm.

As shown in Fig. 5, the red square represents the sample to be classified. It needs to be classified into blue star or green triangle. It is obvious that it is classified to green triangle while k is set to 5, since the probability of classifying it into green triangle is 60 %, which is higher than that of classifying it to blue star (40 %). While k is set to 10, the red square is classified into blue star, since the probability of classifying it into blue star is 60 %, higher than the probability of classifying it into green triangle (40 %). In order to apply the k-nearest Neighbour classification, we need to define a distance metric or similarity function. Common choices include the Euclidean distance:

$$d(p, q) = \sqrt{\sum_{i=1}^{N} (q_i - p_i)^2}, \tag{3}$$

where N is the number of variables, and q_i and p_i are the values of the ith variable at points p and q respectively.

Other distance metrics/similarity functions can be used depending on the type of data (the chi-squared distance is often used for distributions (histograms)). In our case we have been using the Euclidean distance to compare images for similarity [8] and [9].

2.4. Convolutional Neural Network

The Convolutional Neural Networks (CNN) are very similar to ordinary Neural Networks. They are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. They still have a loss function (e.g. SVM/ Softmax) on the last (fully-connected) layer and all the tips/tricks we have developed for learning regular Neural Networks still apply [10].

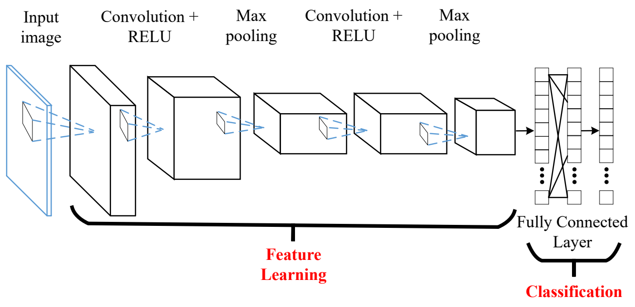

Fig. 6: A convolutional neural networks (CNN).

The CNN consists of multiple layers (see Fig. 6 and Fig. 7). Each layer takes a multi-dimensional array of numbers as input and produces another multi-dimensional array of numbers as output (which then becomes the input of the next layer). When classifying images, the input to the first layer is the input image (32×32), while the output of the final layer is a set of likelihoods of the different categories (i.e., $1 \times 1 \times 10$ numbers if there are 10 categories). A simple CNN is a sequence of layers, and every layer of a CNN transforms one volume of activations to another through a differentiable function. We have used three main types of layers to build CNN architectures: Convolution (CONV) Layer, Pooling Layer, and Fully-Connected Layer (exactly as seen in regular Neural Networks). We have stacked these layers to form a full CNN architecture:

- INPUT [32×32] holds the raw pixel values of the image, in this case an image of width 32, height 32.
- CONV layer computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. This may result in volume such as [$32 \times 32 \times 12$] if we decided to use 12 filters.

- RELU layer applies an elementwise activation function, such as the $\max(0, x)$ thresholding at zero. This leaves the size of the volume unchanged ($[32 \times 32 \times 12]$).
- POOL layer performs a down-sampling operation along the spatial dimensions (width, height), resulting in volume such as $[16 \times 16 \times 12]$.
- FC (Fully-Connected) layer computes the class scores, resulting in volume of size $[1 \times 1 \times 10]$, where each of the 10 numbers corresponds to a class score. As with ordinary Neural Networks and as the name implies, each neuron in this layer is connected to all the neurons in the previous volume.

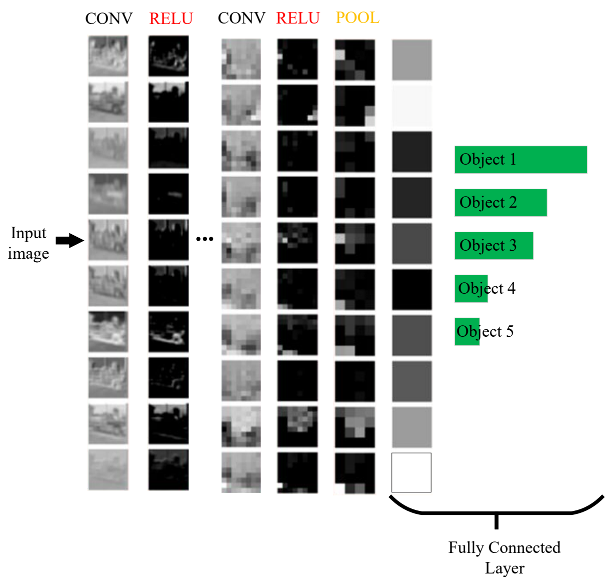

Fig. 7: The activations of an example CNN architecture.

Pooling layer (see Fig. 8) downsamples the volume spatially, independently in each depth slice of the input volume. In this example, the input volume of size $[224 \times 224 \times 64]$ is pooled with filter size 2, stride 2 into output volume of size $[112 \times 112 \times 64]$ (see Fig. 8 top). Notice that the volume depth is preserved. The most common downsampling operation is max, giving rise to max pooling (see Fig. 8 down). That is, each max is taken over 4 numbers (little 2×2 square) [11].

In this way, CNN transforms the original image layer by layer from the original pixel values to the final class scores. Note that some layers contain parameters and other don't. In particular, the CONV/FC layers perform transformations that are a function of not only the activations in the input volume, but of the parameters (the weights and biases of the neurons) as well. On the other hand, the RELU/POOL layers will implement a fixed function [10], [11] and [12].

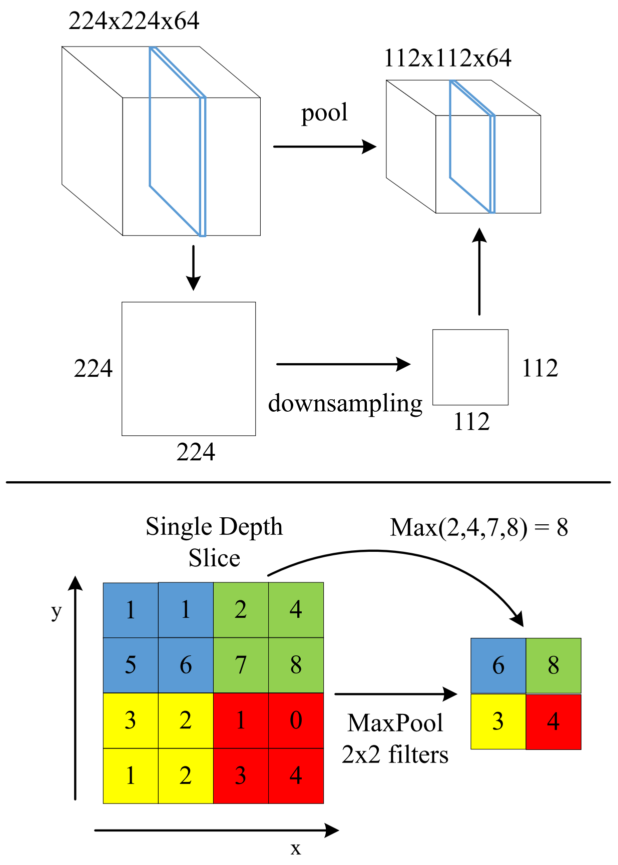

Fig. 8: Max Pooling operation on feature map (2×2 window).

The parameters in the CONV/FC layers have been trained with gradient descent so that the class scores that the CNN computes are consistent with the labels in the training set for each image [12].

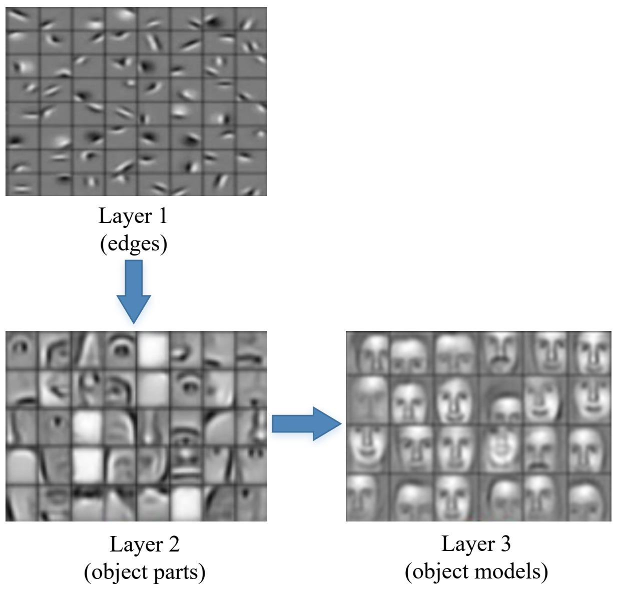

Fig. 9: Layers in convolutional neural network.

A CNN architecture is in the simplest case a list of Layers that transform the image volume into an output volume (e.g. holding the class scores) [12]:

- There are a few distinct types of Layers (e.g. CONV/FC/RELU/POOL are by far the most popular).
- Each Layer accepts an input 3D volume and transforms it to an output 3D volume through a differentiable function.
- Each Layer may (CONV/FC) or may not have (RELU/POOL) parameters.
- Each Layer may (CONV/FC/POOL) or may not have (RELU) additional hyper parameters.

In Fig. 9 you see representation of layers and filters in network for detecting face. First layer can detect basic edges. Second layer detects features from previous layers, thus it is able to detect more complex shapes like eye, nose or mouth. The third and last layer can detect whole faces.

3. Experiments and Results

In this section, we evaluate the performance of our proposed method on ORL face database. In all our experiments, all the images were aligned and normalized based on the positions of human eyes. All the tested methods (PCA, LBPH, KNN and proposed CNN) were implemented in MATLAB and C++/Python programming language.

3.1. Face Dataset

The ORL Database of Faces (see Fig. 10), contains ten different images of each of 40 distinct subjects (400 different images). For some subjects, the images were taken at different times and under varying the lighting [13].

In this database the different moods of the images (faces) such as open and closed eyes and laugh or without laugh with other details like having beard or being beardless, with or without glasses are presented. All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). The forehead and hair of people can be observable in the related pictures. The face situation towards the camera angle is variable from top to bottom and left to right side. All the pictures are black and white with 112×92 pixels. The files are in PGM format.

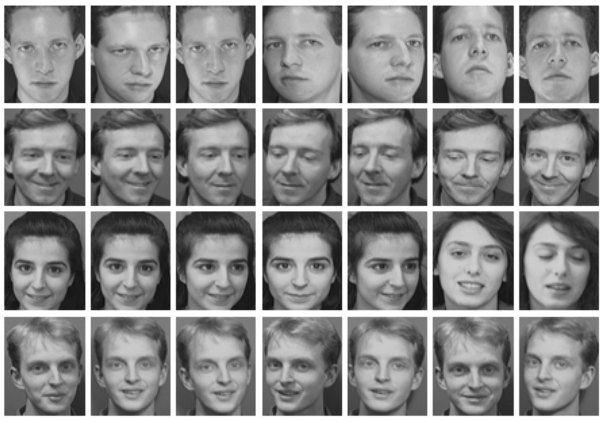

Fig. 10: The example of the ORL face database.

3.2. Experiments

The example of input images from the training database is shown in Fig. 10. All the tested methods follow the principle scheme of the image recognition process (see Fig. 1). Training images and test images as a vector are transformed and stored. These training and testing data form the whole face database (see Fig. 10). The Euclidean distance for the designation of feature vector was used (accuracy of the face recognition algorithm between the test images and all the training images).

Tab. 1: The CNN layers.

Layer	Type	Properties
Layer 1	Input	32×32
Layer 2	1 Convolutional	16 feature maps 3×3 kernel dimension
Layer 3	Pooling	2×2 kernel dimension probability 0.25
Layer 4	2 Convolutional	16 feature maps 3×3 kernel dimension
Layer 5	Pooling	2×2 kernel dimension probability 0.25
Layer 6	Fully-connected	3000 neurons
Layer 7	Fully-connected (Softmax)	40 neurons (classes)

In order to evaluate the effectiveness of our proposed CNN (see Tab. 1 and Fig. 11), we compare the face recognition rate with 3 well-known algorithms (PCA, LBPH, and KNN). After the system is trained using the training data, the feature space "Eigenfaces" are found through PCA. Unlike Eigenfaces, Local Binary Patterns Histograms (LBPH) extract local features of the object and have its roots in 2D texture analysis. The spatial information must be incorporated in the face recognition model. Then, the spatially enhanced feature vector is obtained by concatenating the histograms, not merging them. In our experiments, the KNN classifier used two data types. To create a classification model, training data are used. To test and evaluate trained model accuracy, testing data are used.

The proposal of the Convolutional Neural Network (CNN) is shown in Fig. 11. The input image contains 1024 pixels (32×32). The convolutional layer is followed by the Pooling Layer. Each of the layers gets a 3D-input volume, called the feature map, and transforms it to another by means of convolutions and a non-linearity. By stacking layers and downsampling their outputs, CNNs extract more complex and abstract feature maps, which are, at the same time, invariant to distortions and translations. The last layers of CNN are standard fully connected layers. These layers compute final descriptors of the input images, which can be considered as global representations of images, or they classify the input images into classes depending on an objective function.

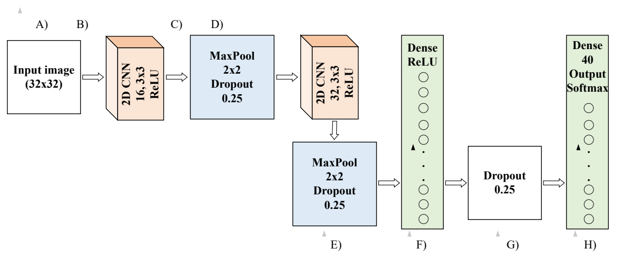

Fig. 11: Block diagram of proposed CNN.

The CNN is a feedforward network composed of layers (see Tab. 1) that transform an input image from the original pixel values to the final class scores by forwarding it layer by layer. The proposed CNN has 2 convolutional layers excluding the source input data layer, fully-connected layers, ReLU layers, and 2 overlapping pooling layers. Each layer has a plurality of feature maps. Each feature map could extract one selected feature through a convolution filter and contains multiple neurons. The data layer contains the image after pre-processing. This convolutional network is divided into 8 blocks:

- A) As input data, faces from ORL dataset were used. Each face was resized into 32×32 pixel to improve the computation time.
- B) The second block was 2D CNN layer which has 16 feature maps with 3×3 kernel dimension. L2 regularization was used due to small dataset. As activation function, Rectifier linear unit (ReLU) was used. This effect improved the sparse features of the whole network and avoided the dependency for passing parameters among the neurons.
- C) For the MaxPooling layers, kernel with dimension 2×2 was used and output was dropped out with probability 0.25. The down sampling layer used the max-pooling method which could have kept the useful information and cut the amount of data which needed to be processed on the upper level.
- D) The second 2D CNN was used with the same parameters as the first one, but amount of feature maps was doubled to 32.
- E) Again, MaxPooling layer and Dropout with same value as in block C were used.
- F) As next layer, the standard dense layer was used, which had 3000 neurons and as activation function Relu was used. L2 regularization was used again to better control the weights.
- G) The output of the last dropout layer was passed to the Softmax of the loss layer.
- H) The final output was a classified distribution with respect to the 40 different classes and Softmax activation function. For validation of the training progress, the Softmax regression was used as the output layer.

For feature extraction, we used the last FC layer as the output. In our experiments, we visualized the activation values of the second convolutional layers of the proposed CNN, which are shown in Fig. 12. In the proposed CNN (see Fig. 11), the pooling operation

Tab. 2: The course of learning by proposed CNN.

Epoch	Iteration	Time Elapsed (seconds)	Mini-batch Loss	Mini-batch Accuracy	Base Learning Rate
1	1	1.68	4.0947	1.67 %	0.0100
2	10	18.69	4.0931	0.00 %	0.0100
4	20	38.03	4.0889	1.67 %	0.0100
6	30	56.68	4.0742	22.50 %	0.0100
8	40	74.59	3.9835	25.00 %	0.0100
10	50	89.78	2.3881	56.67 %	0.0100
12	60	105.14	0.6209	76.67 %	0.0010
14	70	120.30	0.0182	100.00 %	0.0010
16	80	136.81	0.0045	100.00 %	0.0010
18	90	153.30	0.0080	100.00 %	0.0010
20	100	169.10	0.0049	100.00 %	0.0010

was applied separately to each feature map. In general, the more convolutional steps we have, the more complicated features of our proposed network is able to learn to recognize. For example, in image classification a CNN may learn to detect edges from raw pixels in the first layer, then use the edges to detect simple shapes in the second layer, and then use these shapes to deter higher-level features, such as facial shapes in higher layers.

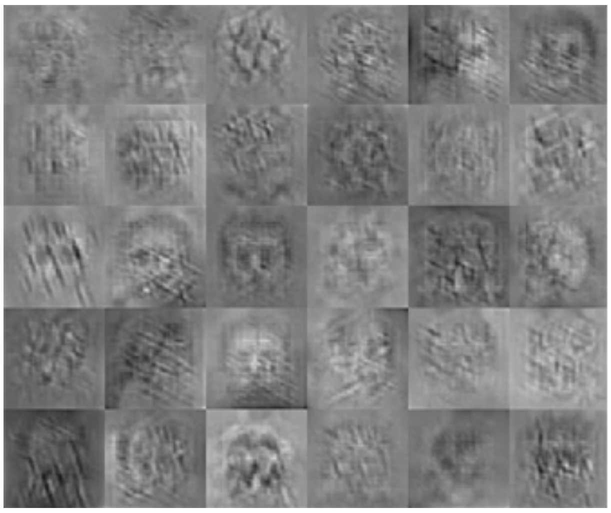

Fig. 12: The example of features of the second convolutional layers.

3.3. Results

The obtained experimental results are presented in this section. The first row in Tab. 3 presents recognition accuracy using PCA algorithm. The next row (see Tab. 3) presents overall accuracy of the LBPH algorithm. The overall accuracy of the KNN algorithm is described in a third row. The last row of the Tab. 3 describes the experimental results of the proposed CNN (overall accuracy).

Tab. 3: The overall face recognition rate for different number of training images.

	Number of training and test images			
	A	B	C	D
PCA (%)	75.2	77.5	82.1	85.6
LBPH (%)	78.1	81.3	86.7	88.9
KNN (%)	71.4	73.8	79.2	81.4
Proposed CNN (%)	93.9	95.7	97.5	98.3

The all performed experimental results are divided into four main parts:

- A – 40:400 (10 % of the data was used for training).
- B – 120:400 (30 % of the data was used for training).
- C – 200:400 (50 % of the data was used for training).
- D – 320:400 (80 % of the data was used for training).

The first part of our performed experiments consists of 40 training images and 400 test images. The second part consists of 120 training images and 400 test images. The next part consists of 200 training images and 400 test images. Finally, the last part of our experiments consists of 320 training images and 400 test images. All the training data in A, B, C and D parts were used for testing too. The obtained experimental results are presented in Tab. 3.

The best results (accuracy of 98.3 %) using the proposed CNN was obtained for the 320 training images (see Tab. 3). On the other hand, the worst results (accuracy of 71.4 %) using KNN algorithm were obtained for the 40 training images.

The mini-batch accuracy reported during training corresponds to the accuracy of the particular mini-batch at the given iteration (see Tab. 2). It is not a running average over iterations. An iteration corresponds to the calculation of the network's gradients for each mini-batch. An epoch corresponds to moving through every available mini-batch.

4. Conclusion

In this work, we presented an experimental evaluation of the performance of proposed CNN. The overall performances were obtained using the different number of training images and test images. The convolutional neural networks achieve the best results so far. Using complex architectures, it is possible to reach accuracy rates of about 98 %. Despite this impressing outcome, CNNs cannot work without negative impacts. Very huge training datasets lead to a high computation load and memory usage, which then needs high processing power to be able to be applied usefully. In our case, the largest tested face dataset consists of 1521 greyscale images with a resolution of 384 × 286 pixel (BioID Face Database) [14]. This database contained 23 different test images (persons). The obtained experimental results based on this database will be published in our next work. Thanks to the development of better and faster hardware, it is no problem anymore to cope with the vast amount of parameters. It can be seen, that every object algorithm has different advantages and disadvantages. Hence, it is almost not possible to create a complete, meaningful ranking, as too many different aspects have to be considered. It depends on the target application which algorithm shall be used.

For the future work, we can use more different kinds of categories that would be difficult for the computer to classify and compare more sophisticated classifiers.

Acknowledgment

This publication is the result of the project implementation: Centre of excellence for systems and services of intelligent transport II, ITMS 26220120050 supported by the Research & Development Operational Programme funded by the ERDF.

References

[1] MEENA, D. and R. SHARAN. An approach to face detection and recognition. In: *International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*. Jaipur: IEEE, 2016, pp. 1–6. ISBN 978-1-5090-2807-8. DOI: 10.1109/ICRAIE.2016.7939462.

[2] REKHA, E. and P. RAMAPRASAD. An efficient automated attendance management system based on Eigen Face recognition. In: *7th International Conference on Cloud Computing, Data Science & Engineering - Confluence*. Noida: IEEE, 2017, pp. 605–608. ISBN 978-1-5090-3519-9. DOI: 10.1109/CONFLUENCE.2017.7943223.

[3] ABUROMMAN, A. A. and M. B. I. REAZ. Ensemble SVM classifiers based on PCA and LDA for IDS. In: *International Conference on Advances in Electrical, Electronic and Systems Engineering (ICAEES)*. Putrajaya: IEEE, 2016, pp. 95–99. ISBN 978-1-5090-2889-4. DOI: 10.1109/ICAEES.2016.7888016.

[4] OLIVARES-MERCADO, J., K. TOSCANO-MEDINA, G. SANCHEZ-PEREZ, H. PEREZ-MEANA and M. NAKANO-MIYATAKE. Face recognition system for smartphone based on LBP. In: *5th International Workshop on Biometrics and Forensics (IWBF)*. Coventry: IEEE, 2017, pp. 1–6. ISBN 978-1-5090-5791-7. DOI: 10.1109/IWBF.2017.7935111.

[5] KAMENCAY, P., T. TRNOVSZKY, M. BENCO, R. HUDEC, P. SYKORA and A. SATNIK. Accurate wild animal recognition using PCA, LDA and LBPH. In: *ELEKTRO*. Strbske Pleso: IEEE, 2016, pp. 62–67. ISBN 978-1-4673-8698-2. DOI: 10.1109/ELEKTRO.2016.7512036.

[6] AMEUR, B., S. MASMOUDI, A. G. DERBEL and A. BEN HAMIDA. Fusing Gabor and LBP feature sets for KNN and SRC-based face recognition. In: *International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*. Monastir: IEEE, 2016, pp. 453–458. ISBN 978-1-4673-8526-8. DOI: 10.1109/ATSIP.2016.7523134.

[7] STEKAS, N. and D. HEUVEL. Face Recognition Using Local Binary Patterns Histograms (LBPH) on an FPGA-Based System on Chip (SoC). In: *IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. Chicago: IEEE, 2016, pp. 300–304. ISBN 978-1-5090-3682-0. DOI: 10.1109/IPDPSW.2016.67.

[8] WANG, Q., K. JIA and P. LIU. Design and Implementation of Remote Facial Expression Recognition Surveillance System Based on PCA and KNN Algorithms. In: *International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*. Adelaide: IEEE, 2015, pp. 314–317. ISBN 978-1-5090-0188-0. DOI: 10.1109/IIH-MSP.2015.54.

[9] NUGRAHAENI, R. A. and K. MUTIJARSA. Comparative analysis of machine learning KNN, SVM, and random forests algorithm for facial expression classification. In: *International Seminar on Application for Technology of Information and Communication (ISemantic)*. Semarang: IEEE, 2016, pp. 163–168. ISBN 978-1-5090-2326-4. DOI: 10.1109/ISEMANTIC.2016.7873831.

[10] BRITZ, D. Understanding convolutional neural networks. In: *WILDML* [Online]. 2015. Available at: `http://www.wildml.com/2015/11/understanding-convolutional\-neural-networks-for-nlp/`.

[11] TOBIAS, L., A. DUCOURNAU, F. ROUSSEAU, G. MERCIER and R. FABLET. Convolutional Neural Networks for object recognition on mobile devices: A case study. In: *23rd International Conference on Pattern Recognition (ICPR)*. Cancun: IEEE, 2016, pp. 3530–3535. ISBN 978-1-5090-4847-2. DOI: 10.1109/ICPR.2016.7900181.

[12] GUO, S., S. CHEN and Y. LI. Face recognition based on convolutional neural network and support vector machine. In: *IEEE International Conference on Information and Automation (ICIA)*. Ningbo: IEEE, 2016, pp. 1787–1792. ISBN 978-1-5090-4102-2. DOI: 10.1109/ICInfA.2016.7832107.

[13] The Database of Faces (ORL Face Database). In: *AT&T Laboratories Cambridge* [Online]. 2002. Available at: `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`.

[14] The BioID Face Database – FaceDB. In: *BioID* [Online]. 2017. Available at: `https://www.bioid.com/About/BioID-Face-Database`.

About Authors

Patrik KAMENCAY was born in Topolcany in 1985, Slovakia. He received his M.Sc. and Ph.D. degrees in Telecommunications from the University of Zilina, Slovakia, in 2009 and 2012, respectively. His Ph.D. research work was oriented to reconstruction of 3D images from stereo pictures. Since October 2012 he is researcher at the Department of Multimedia and Information-Communication Technologies (MICT), University of Zilina. His research interest includes holography for 3D display and construction of 3D objects of the real scene.

Miroslav BENCO was born in Vranov nad Toplou in 1981, Slovakia. He received his M.Sc. degree in 2005 at the Department of Control and Information System and Ph.D. degree in 2009 at the Department of MICT, University of Zilina. Since January 2009 he is researcher at the Department of MICT, University of Zilina. His research interest includes digital image processing, and semantic analysis of multimedia content.

Tomas MIZDOS was born in 1993 in Poprad, Slovakia. He received his M.Sc. degrees in Multimedia engineering at the Department of Multimedia and Information-Communication Technologies, Faculty of Electrical Engineering, at the University of Zilina in 2017. Nowadays he is Ph.D. student at the same department. His main area of interest is functionality and quality of multimedia services.

Roman RADIL was born in Trencin, Slovakia, in 1984. Graduated from the Faculty of Electrical Engineering, University of Zilina, in 2008 from Biomedical Engineering and received the Ph.D. degree in Theoretical Electromagnetics at the same university, in 2012. At present he works as a researcher at the Department of Electromagnetic and Biomedical Engineering, Faculty of Electrical Engineering, University of Zilina. His research activities are focused on investigation of low frequency electromagnetic field effects on biological samples and biomedical signal (image) processing.