IAPR ICPR 2018
INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION
In Beijing, China — August 20th-24th 2018

Efficient Statistical Face Recognition Using Trigonometric Series and CNN Features

Andrey V. Savchenko
Dr. of Sci., PhD,
- Full Professor, Senior researcher in Laboratory of Algorithms and Technologies for Network Analysis, National Research University Higher School of Economics, Nizhny Novgorod, Russia
- Senior researcher in Samsung-PDMI Joint AI Center

E-mail: avsavchenko@hse.ru

August, 2018

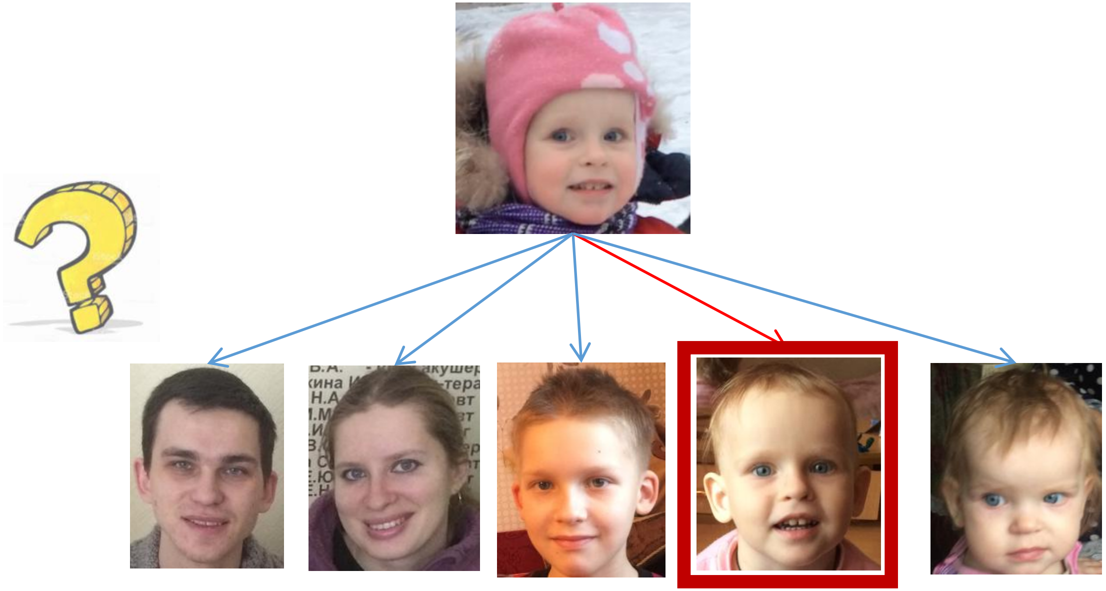
Face identification

What for?

Assign an observed facial image X to one of $C>1$ **identities** specified by N **reference** images $\{X_n\}$ from the gallery set.

Small-sample-size case: *number of reference photos per class is small* (5-20 photos per subject).The number of subjects can be very large

Input photo



Gallery set

Key idea

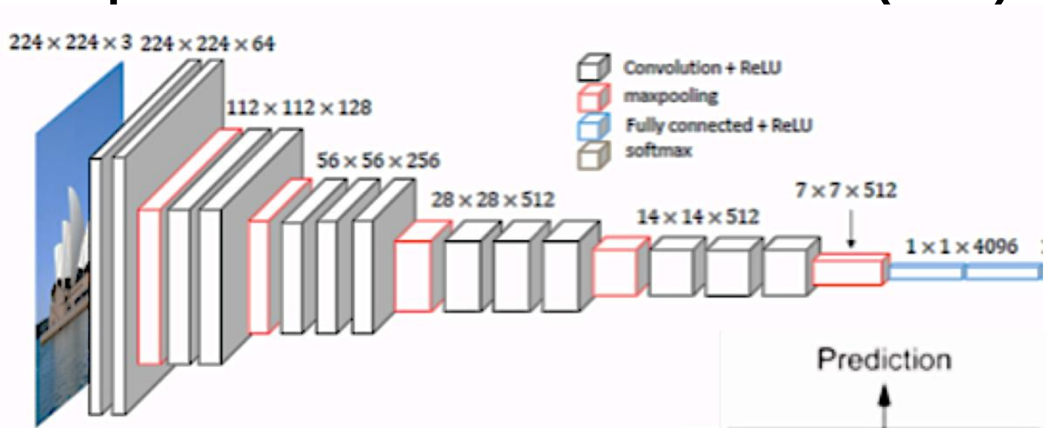
Improve the **running time** of instance-based learning by using estimates of class densities with the trigonometric series expansion

And now we introduce the agenda of our talk

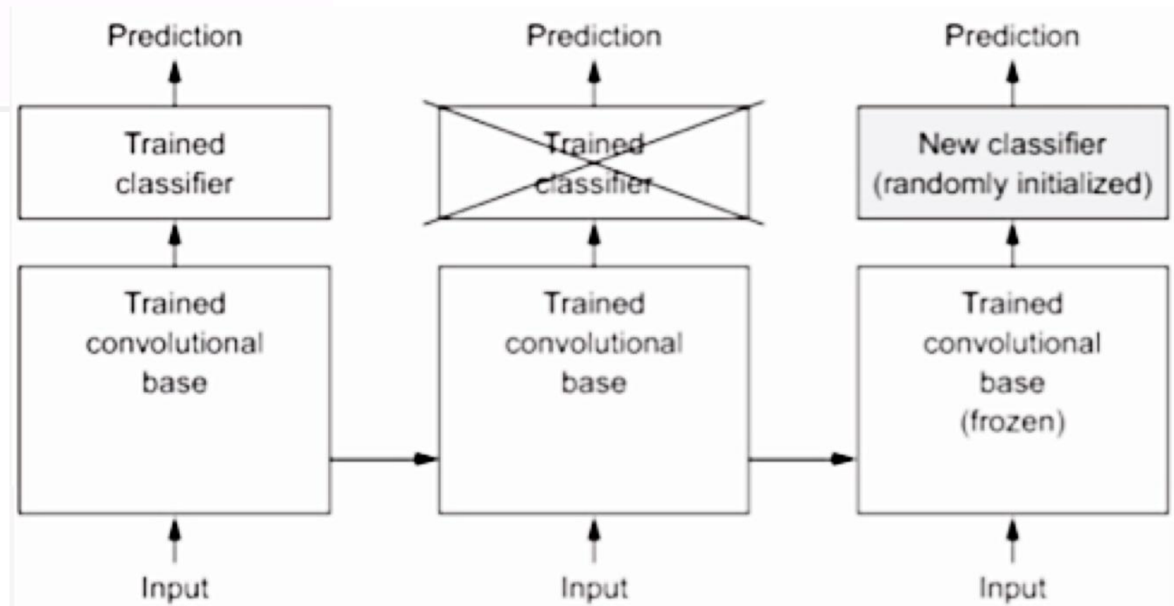
1. Instance-based Learning in Face Recognition
2. Proposed Method
3. Experimental results
4. Conclusion and future work

Facial features: Deep Embeddings

Deep Convolutional Neural Networks (CNN)



Deep embeddings: CNN bottleneck features -
high*(D)*-dimensional feature vectors



Contemporary CNNs

VGGFace, FaceNet, CenterFace, DeepID (2/3) LightCNN, VGGFace2, SphereFace, ArcFace,…

Accuracy: 85-99%

Instance-based Learning

Nearest neighbor (NN) classifier

$$\min_{c\in\{1,\dots,C\}} \min_{n\in\{1,\dots,N(c)\}} \rho(\mathbf{x}, \mathbf{x}_n(c)),$$

Statistical approach: empirical Bayesian classifier

$$\max_{c\in\{1,\dots,C\}} p_c \hat{f}(\mathbf{x} \mid H_c)$$

Advantages

1. High accuracy for small sample size (SSS): *N*/*C* is small
2. Very high training speed

Disadvantages

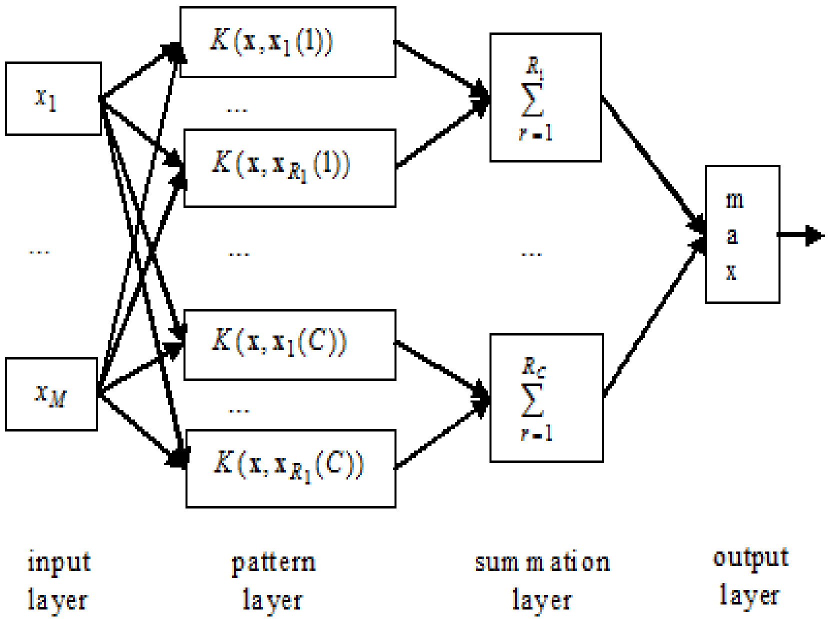
1. Classification performance is low: O(*DN*)
2. Memory-based approach: space complexity is also linear

Probabilistic neural network (PNN)

- The prior probability is estimated as $p_c = N(c)/N$
- The Rosenblatt-Parzen kernel with the Gaussian window is used

$$\hat{f}\left(\mathbf{x} \middle| H_c\right) = \frac{1}{N(c)} \sum_{n=1}^{N(c)} K\left(\mathbf{x}, \mathbf{x}_n(c)\right)$$

$$K\left(\mathbf{x}, \mathbf{x}_n(c)\right) = \frac{1}{\left(2\pi\sigma^2\right)^{D/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{d=1}^{D} \left(x_d - x_{n;d}(c)\right)^2\right)$$



PNN has the same advantages and disadvantages as typical instance-based learning methods

Orthogonal series expansion of probability density

1 **Orthogonal series** (Fourier, Hermite or Legendre) instead of the Parzen kernel. Runtime complexity **exponentially** depends on dimensionality!

Possible solution: naïve Bayesian rule (Perform **PCA**)

$$\hat{f}(x_d \mid H_c) = \sum_{j=0}^{J} a_j \varphi_j(x_d), J \to \infty$$

2 Recognition procedure. Features are **normalized**: Fourier series

$$\varphi_j^{(1)}(x_d) = \cos(\pi j x_d), \varphi_j^{(2)}(x_d) = \sin(\pi j x_d)$$

The Dirichlet kernel is used instead of the Parzen kernel

$$f_j(x_d \mid H_c) = \frac{1}{2N(c)} \sum_{n=1}^{N(c)} \frac{\sin\left(\pi(J+0.5)(x_d - x_{n;d}(c))\right)}{2\sin(\pi(x_d - x_{n;d}(c)))} \qquad (*)$$

Disadvantages

1. The Dirichlet kernel is not always non-negative!
2. Computational complexity remains identical to k-NN/PNN

Proposed Approach (1)

1 Estimate the likelihood as the **average of the first *J* partial sums**

$$f(x_d|H_c) = \frac{1}{J+1}\sum_{n=1}^{N(c)} f_J(x_d|H_c)$$

The right-hand side is the non-negative Fejér kernel

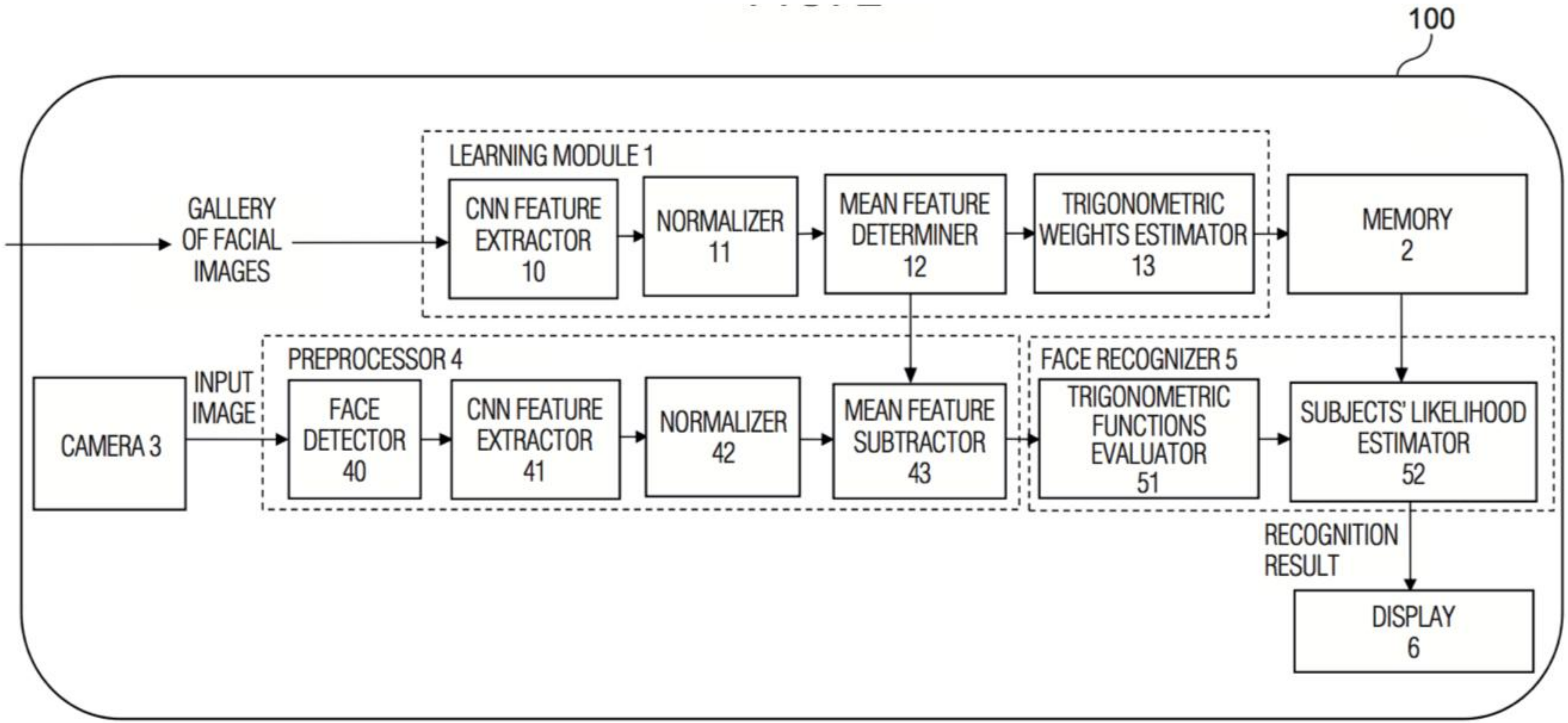
$$F_{J+1} = \frac{1}{J+1}\left(\frac{\sin((J+1)\pi(x_d - x_{r;d}(c))/2)}{\sin(\pi(x_d - x_{r;d}(c))/2)}\right)^2$$

2 Canonical form of density estimate (*) is replaced to the equivalent form

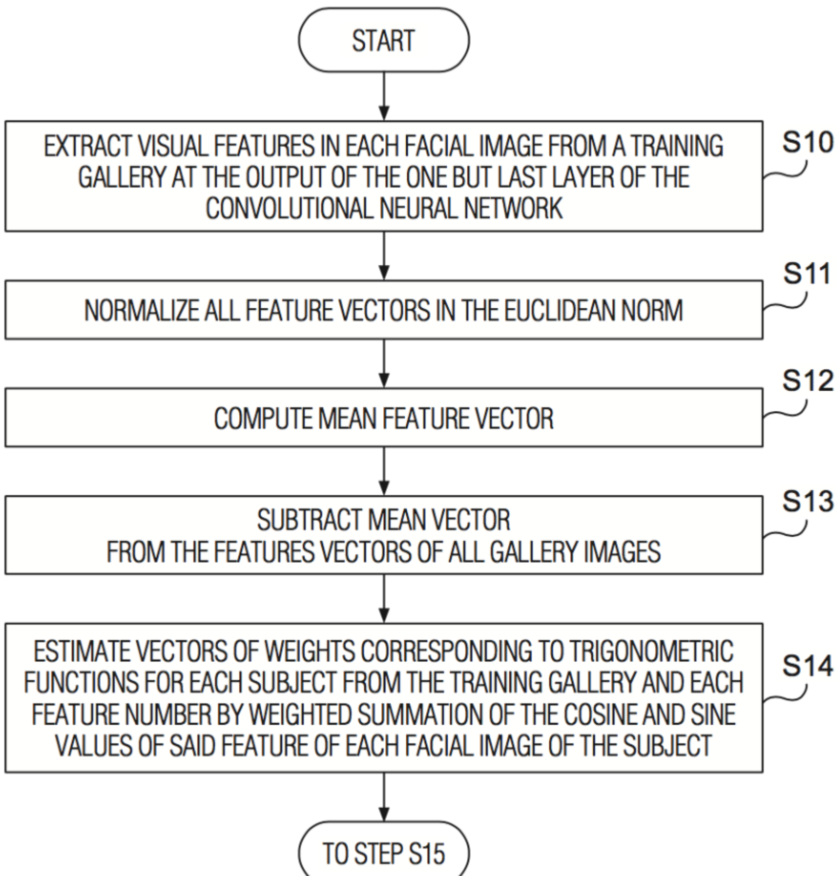
$$f(x_d|H_c) = \frac{1}{2N(c)} + \sum_{j=1}^{J} w_{j;d}^{(1)}(c)\phi_j^{(1)}(x_d) + w_{j;d}^{(2)}(c)\phi_j^{(1)}(x_d)$$

No requirement for an exhaustive search in the training set

Proposed Approach (2)

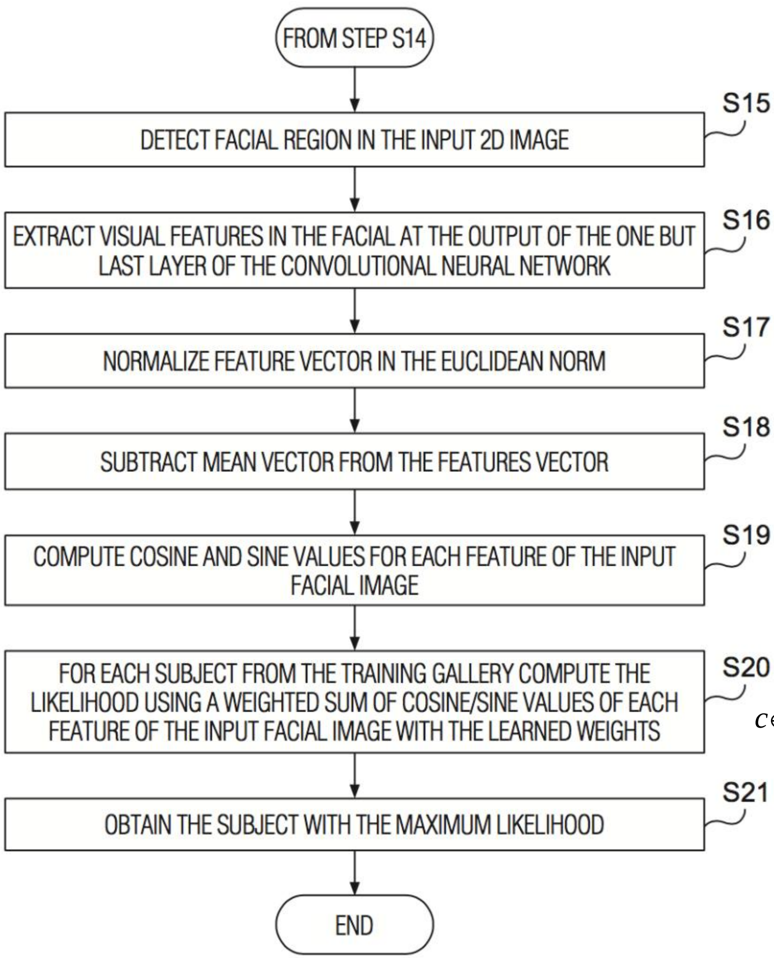


Training



$$w_{j;d}^{(l)}(c) = \frac{J+1-j}{(J+1)N(c)} \sum_{n=1}^{N(c)} \phi_j^{(l)}(x_{n;d}(c)), l \in \{1,2\}.$$

Inference



Fast inference:

$$\varphi_j^{(1)}(x_d) = \varphi_{j-1}^{(1)}(x_d)\cdot\varphi_1^{(1)}(x_d) - \varphi_{j-1}^{(2)}(x_d)\cdot\varphi_1^{(2)}(x_d),$$

$$\varphi_j^{(2)}(x_d) = \varphi_{j-1}^{(1)}(x_d)\cdot\varphi_1^{(2)}(x_d) + \varphi_{j-1}^{(2)}(x_d)\cdot\varphi_1^{(1)}(x_d),$$

$$\varphi_1^{(1)}(x_d) = \cos(\pi x_d), \varphi_1^{(2)}(x_d) = \sin(\pi x_d)$$

$$\max_{c\in\{1,\ldots,C\}}\left(\log N(c) + \sum_{d=1}^{D}\log\left(0.5 + \sum_{j=1}^{J}\left(w_{j;d}^{(1)}(c)\cdot\varphi_j^{(1)}(x_d) + w_{j;d}^{(2)}(c)\cdot\varphi_j^{(2)}(x_d)\right)\right)\right)$$

Advantages of the proposed classifier

1. Converges to the optimal Bayesian decision.
2. Can be implemented completely in parallel.
3. Excellent training speed. The new instance can be added in real time:

$$\frac{N(c)w_{j;d}^{(l)}(c) + \phi_j^{(l)}(x_{N(c)+1;d}(c))}{N(c)+1},$$

4. Orthogonal series estimate converge when J=O($N^{1/3}$):

$$J = 2\left\lceil \sqrt[3]{N/C} \right\rceil$$

Runtime complexity and memory space complexity: O($DN^{1/3}C^{1/3}$)

- Classification is approximately $N^{2/3}$–times faster than the instance-based learning
- More efficient than the instance-based learning, if the following condition holds:

$$\frac{N/C}{4\left\lceil \sqrt[3]{N/C} \right\rceil + 1} \geq 1 \Rightarrow N \geq 5C$$

Experiments

1 Pre-trained deep CNNs (Caffe and Tensorflow/Keras frameworks)

1.1 **LightCNN**: D = 256 embeddings at “eltwise_fc2” layer

2.2 **VGG-Face** (VGG-16 Network): D = 4096 embeddings at “fc8” layer

2.3 **VGGFace2_ft** (ResNet-50 trained on VGGFace2): D = 2048 embeddings at “pool5/7x7_s1” layer

2.4 Our own **MobileNet-192** fine-tuned on the VGGFace2 dataset: D = 1024 embeddings

2 Datasets

2.1 **PubFig83**

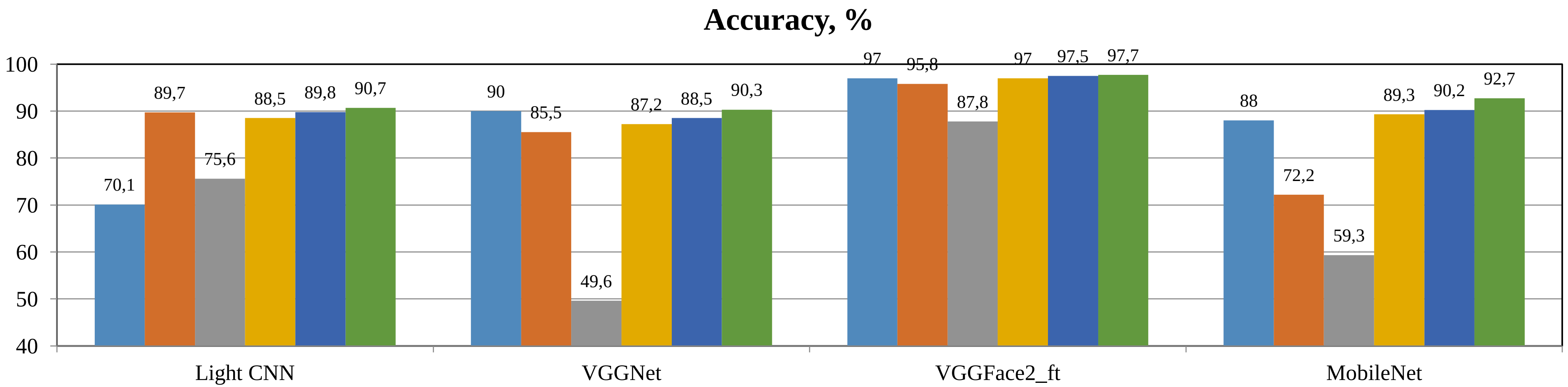
Number of subjects C 83 Total number of photos 13811

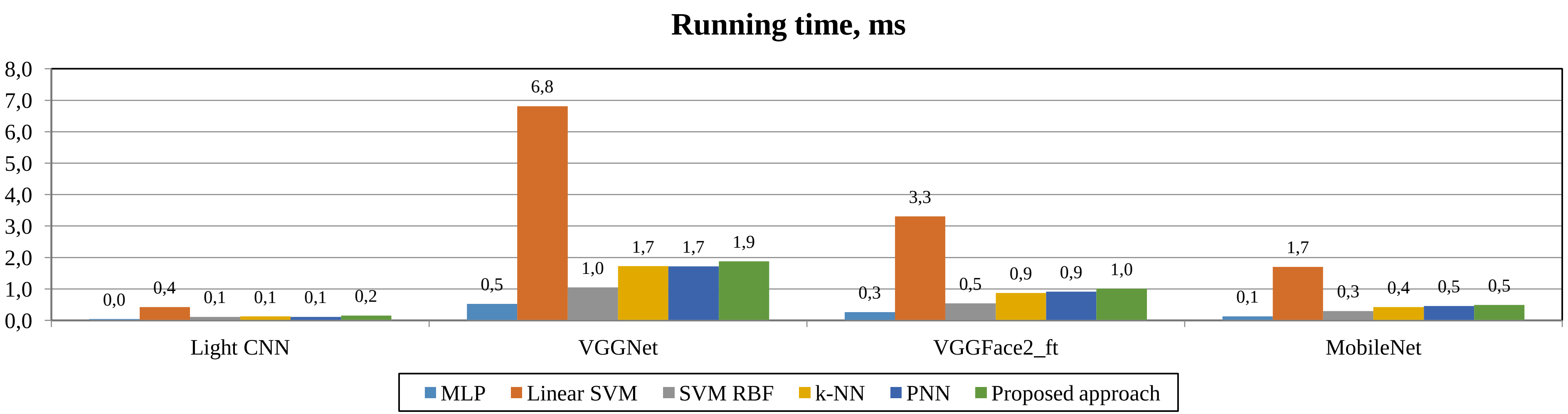
2.2 First 1000 persons from **CASIA WebFace dataset**

Number of subjects C 1000 Total number of photos 79709

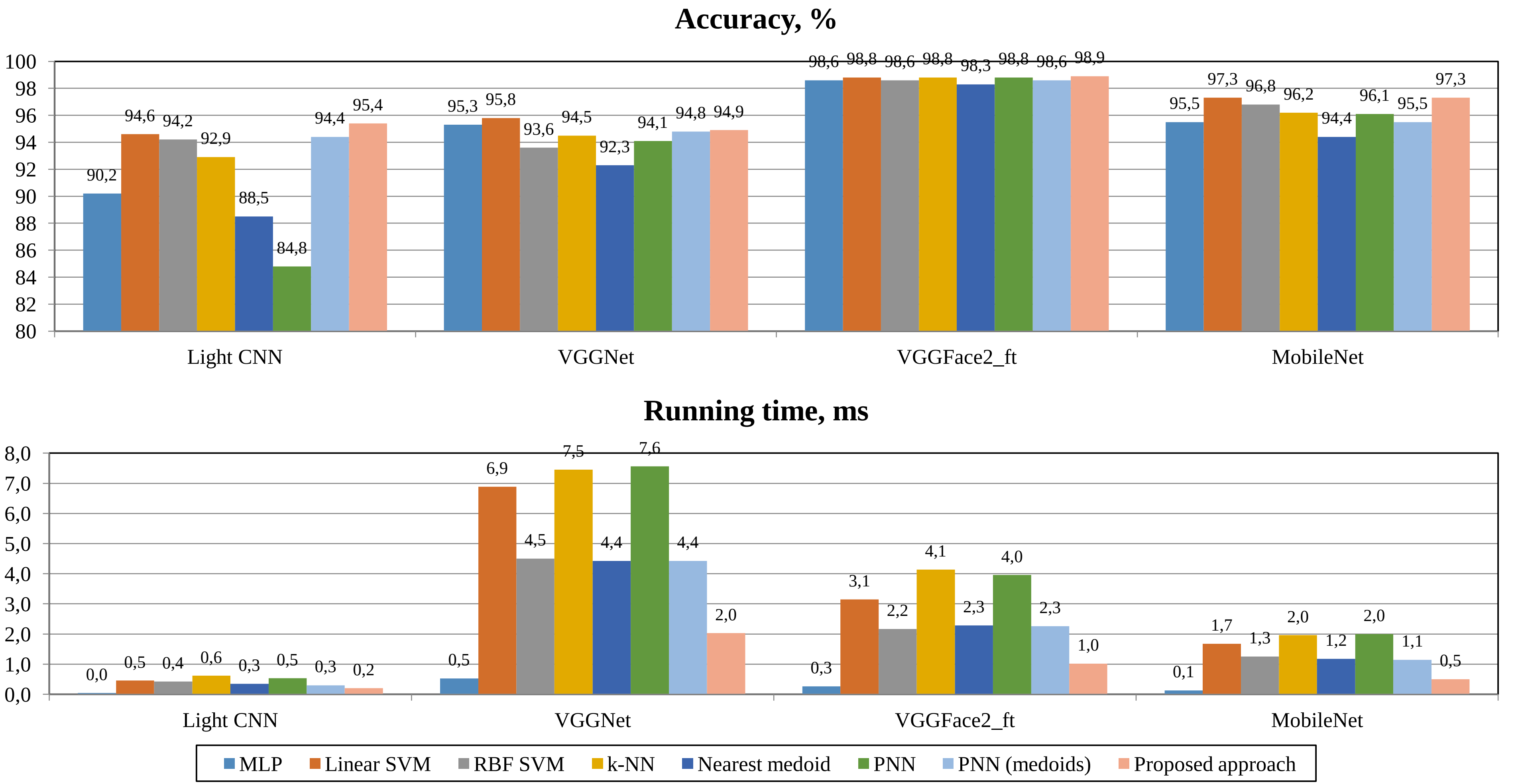
Hardware: MacBook 2015 laptop (8 core 2.2 GHz Intel Core i7, 16 Gb RAM)

PubFig83. N=332

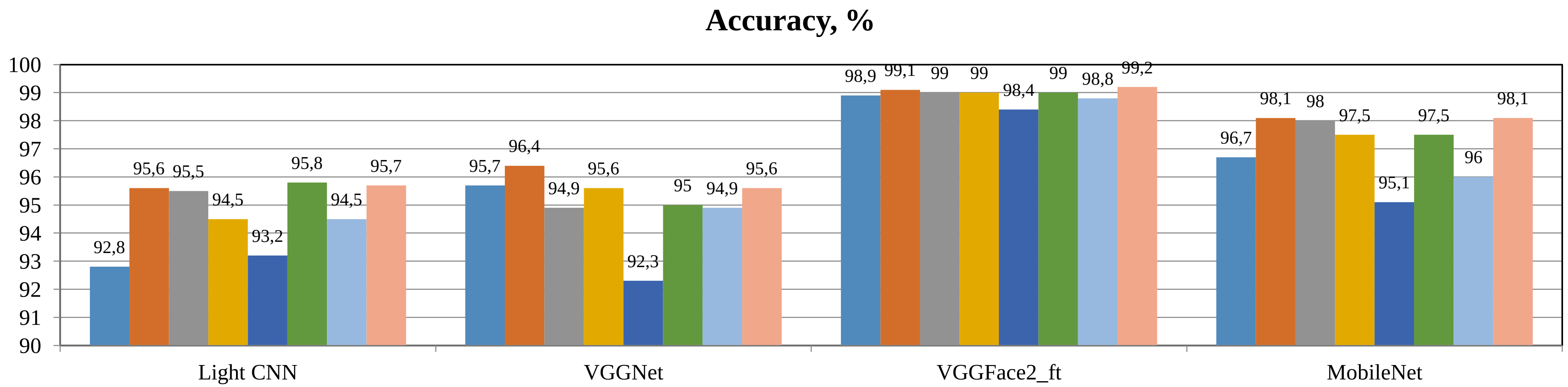


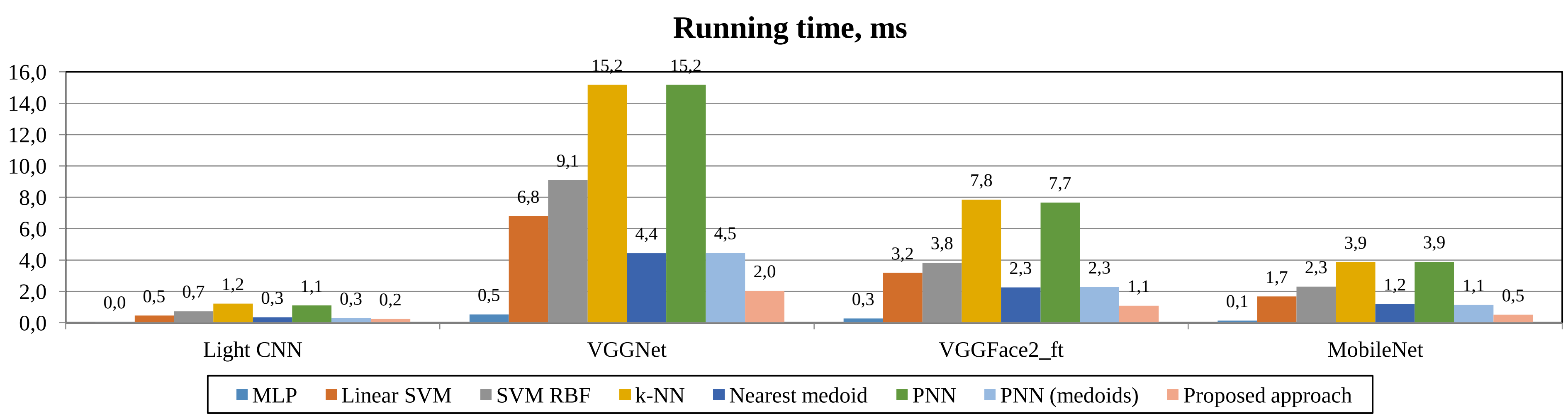


PubFig83. N=1411

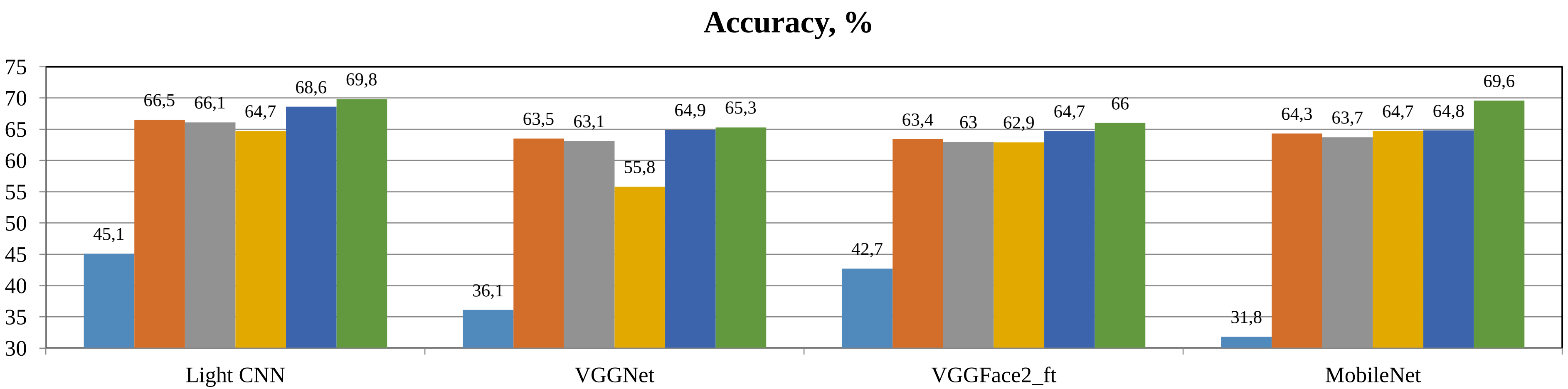


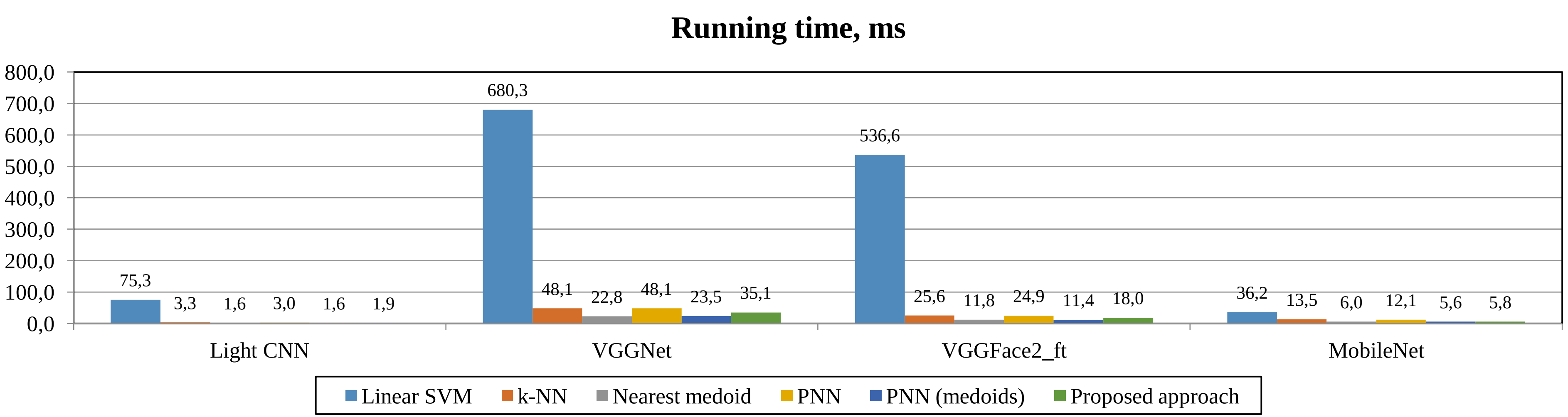
PubFig83. N=2739



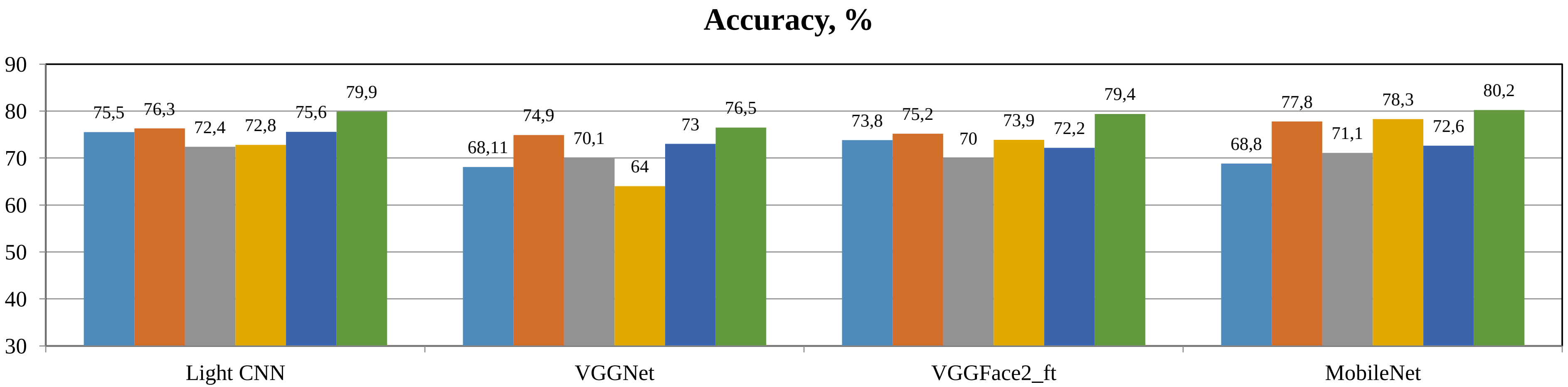


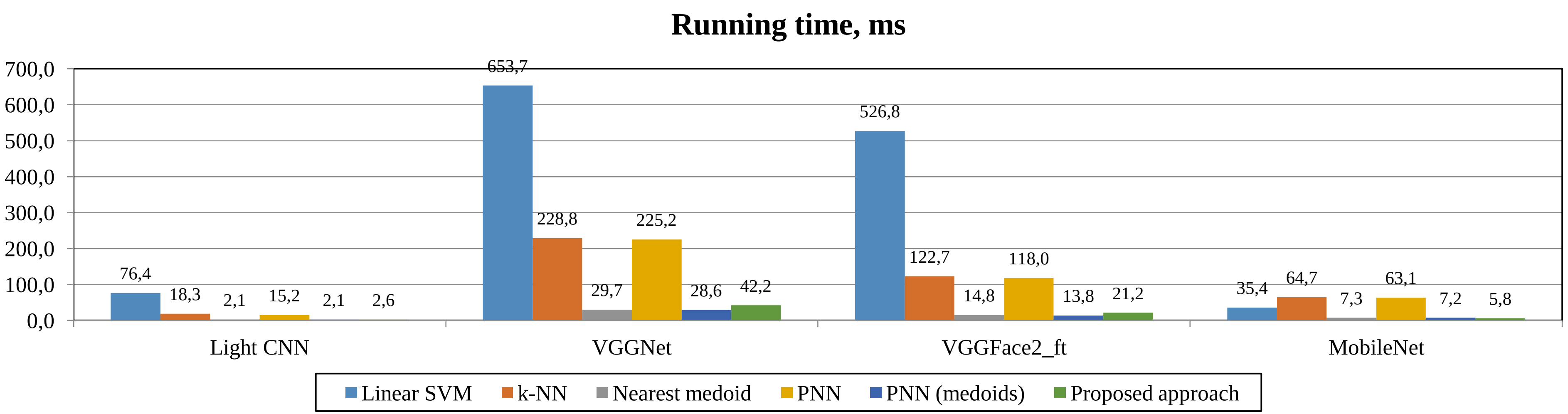
CASIA WebFace. N=8000





CASIA WebFace. N=40000





And summarizing our results we have the following conclusions

Proposed approach has a list of **advantages**

1. Saves all advantages of the PNN including convergence to the Bayesian decision
2. Significantly improves the classification performance of instance-based learning
3. No need to optimize the smoothing parameter in the PNN
4. C++ implementation is freely available:
   - https://github.com/HSE-asavchenko/HSEFaceRec/tree/master/src/recognition_testing
   - https://github.com/HSE-asavchenko/HSE FaceRec_tf

And **disadvantages**

1. Naïve assumption about independence of features
2. No distance calculation as in the Gaussian kernel in the PNN

What we are going to do in the future

1 Application of our approach to other tasks, e.g. image categorization

2 Imbalanced face recognition

- 1.1 Our prior probability estimation does not influence the decision too much. Platt scaling?
- 1.2 Proper choice of the cut-off parameter?

3 Uncorrelated features from the range [-1; 1] are required.
How to normalize various features properly?

4 Implement the Bayesian networks with our classifier in order to weaken the requirement of feature independence

Thank you for your attention

Any Questions?