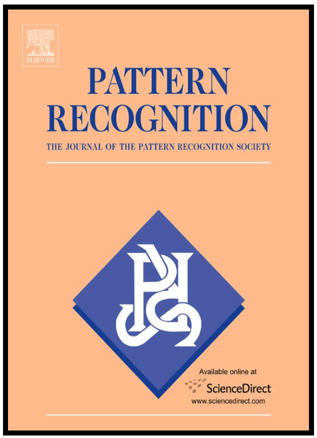
# Maximum-Likelihood Approximate Nearest Neighbor Method in Real-time Image Recognition

A.V. Savchenko



www.elsevier.com/locate/pr

PII: S0031-3203(16)30228-X
DOI: http://dx.doi.org/10.1016/j.patcog.2016.08.015
Reference: PR5844

To appear in: *Pattern Recognition*

Received date: 19 August 2015
Revised date: 20 April 2016
Accepted date: 15 August 2016

Cite this article as: A.V. Savchenko, Maximum-Likelihood Approximate Neares Neighbor Method in Real-time Image Recognition, *Pattern Recognitior* http://dx.doi.org/10.1016/j.patcog.2016.08.015

# Maximum-Likelihood Approximate Nearest Neighbor Method in Real-time Image Recognition

A. V. Savchenko

National Research University Higher School of Economics, Laboratory of Algorithms and Technologies for Network Analysis, 136 Rodionova St., Nizhny Novgorod 603093, Russia

avsavchenko@hse.ru

## ABSTRACT

An exhaustive search of all classes in pattern recognition methods cannot be implemented in real-time, if the database contains a large number of classes. In this paper we introduce a novel probabilistic approximate nearest-neighbor (NN) method. Despite the most of known fast approximate NN algorithms, our method is not heuristic. The joint probabilistic densities (likelihoods) of the distances to previously checked reference objects are estimated for each class. The next reference instance is selected from the class with the maximal likelihood. To deal with the quadratic memory requirement of this approach, we propose its modification, which processes the distances from all instances to a small set of pivots chosen with the farthest-first traversal. Experimental study in face recognition with the histograms of oriented gradients and the deep neural network-based image features shows that the proposed method is much faster than the known approximate NN algorithms for medium databases.

## KEYWORDS

Approximate nearest neighbor method; large database; maximum likelihood; real-time pattern recognition; image recognition; probabilistic neural network; HOG (histograms of oriented gradients); deep neural networks.

## Abbreviations

AESA, Approximating and Eliminating Search Algorithm; ANN, approximate nearest neighbor; DEM, directed enumeration method; DNN, deep neural network; HOG, histograms of oriented gradients; HT-PNN, homogeneity testing probabilistic neural network; ML-ANN, maximal-likelihood approximate nearest neighbor; NN, nearest neighbor; P-ML-ANN, pivot-based maximal-likelihood approximate nearest neighbor; SVM, support vector machine

## 1. Introduction

Image recognition is a challenging problem due to the variability of either objects presented in images or conditions of observations, e.g., illumination, noise, etc. [1]. The input image and all reference images are usually processed to extract either local features, e.g., SIFT [2] and SURF [3], or a global descriptor, e.g., the histograms of oriented gradients (HOG) [4], the pyramid HOGs [5] and the features based on Deep Neural Network (DNN) [6]. After that, machine learning techniques can be applied, e.g., conventional Support Vector Machines (SVM) [4,7] or modern DNNs [8,9]. These image recognition methods have reached a certain level of maturity when the size of the training set is large with respect to the number of different classes. These methods proved to be very efficient in such tasks, as optical character recognition [9], pedestrian identification [4,10], traffic sign classification, etc. [11]. However, the situation becomes much more complicated if the training database contains a small number of reference images per each class [12]. This small sample size problem is crucial in several recognition tasks, e.g., in face recognition when only one reference image of each person is sometimes available [13,14]. In such a case, the nearest neighbor (NN) methods with an appropriate dissimilarity measure can be applied after feature extraction procedure [15]. A variant of this approach is implemented in the DeepFace method [6] which solves the image verification task [16] by training a DNN on an external large face database. The outputs of the last layer of the DNN for the input image and every reference image from the database are used as the feature vectors in the NN method [6,17]. Thus, if the count of reference images for each class is low, practically all image recognition methods are based on the NN search [18]. If the images have to be recognized in real-time (e.g., in video-based face recognition [19]), the feature vector is high-dimensional [20], the dissimilarity measure is complex [21] and only standard hardware (e.g., a laptop) is available, the performance of the brute-force NN search is not enough even for medium (thousands of classes) and very large databases. A lot of approximate NN (ANN) methods have been proposed to solve this problem and speed up the recognition procedure [22]. It is known that these methods are specially optimized for very large databases [23], so they are not efficient when the number of classes does not exceed a few thousands [24].

Thus, our goal is an improvement of the performance of the NN search in the case of medium databases by using the asymptotic properties of the Bayesian classifier [25]. The novelty of this paper is summarized as follows.

1) The novel Maximal-Likelihood ANN (ML-ANN) method is presented based on the statistical approach. While looking for the NN for a particular input image, we estimate conditional probability of the previously checked images belonging to each class. The next reference image from the database is selected from the class with the maximal probability.

2) The pivot-based modification of the ML-ANN with the pivot indexing [26] is proposed to overcome the disadvantages of the ML-ANN, namely, the storage of the full matrix of distances and high computational complexity of each step.

The rest of the paper is organized as follows. In Section 2, related work is briefly described. In Section 3, we explore the task of statistical image recognition and briefly recall the baseline ANN algorithm, namely, the directed enumeration method (DEM) [27]. In Subsection 4.1, we introduce our ML-ANN. Its pivot-based modification is presented in Subsection 4.2. In Section 5, we demonstrate experimental results of comparing our method with several ANN algorithms in face recognition for the FERET, the Essex and the LFW (Labeled Faces in the Wild) datasets. Finally, concluding comments are given in Section 6.

## 2. Related Works

There are a large number of papers devoted to fast algorithms of the NN search. All existing methods are divided into two parts: exact and approximate search. Exact methods make it possible to obtain an exact NN without an exhaustive search of the whole database. The most popular of them are various modifications of the kd-trees [28]. This data structure is used to organize k-dimensional data so that an exact NN is obtained as in the ordinary binary search tree but with the addition of backtracking. Backtracking eliminates branches of the tree if the hyper sphere around the search point does not intersect the splitting plane. As a result, the kd-tree can be created only for dissimilarity measures, which satisfy the triangle inequality. Though the Euclidean metric is the most popular in pattern recognition [2], several other dissimilarity measures can increase the recognition accuracy [6,21]. The search procedure in the kd-tree is efficient only for low-dimensional data; otherwise practically no branches are eliminated during the backtracking. This effect is known as the "curse of dimensionality" [29,30].

To deal with this issue, approximate search of the NN was proposed. The ANN methods return an exact NN with high probability and an appropriate reference point in other cases. The simplest option here is to perform a range query and look for a reference point with the distance to the query point not exceeding a low threshold [27,30]. Another possible approach is the sequential processing of multi-resolution image features (pyramids) [31], e.g., the pyramid HOG descriptor [5]. The next hierarchical level with the higher resolution is processed, only if the decision at the current level is not reliable enough [32,18]. However, such a sequential analysis causes the *decrease* of recognition speed, if the rate of unreliable decisions at the lower resolution layers is high. Moreover, this approach is still based on the brute force. Hence, it can be inappropriate for image recognition with medium databases.

Most of ANN methods are based on the effective pruning by using the triangle inequality. An interesting approach is the AESA (Approximating and Eliminating Search Algorithm) [33]. During the search, the distance between the query point and any reference instance $X_r$ is computed. The algorithm prunes all reference images for which the absolute value of the difference of this distance and the distance to $X_r$ exceeds a threshold. The implementation of the AESA requires quadratic memory space to store the distance matrix. There are several modifications of the AESA that perform slightly worse, but require only linear memory space [34]. A similar approach to classification with the non-metric distances is the directed enumeration method (DEM), which outperforms popular ANN methods in face recognition task [27]. The DEM examines the reference images, whose distances to the previously checked instance $X_r$ are approximately equal to the distance between input image and $X_r$.

Another possible way to look for ANN is the modification of the exact methods by applying an early termination strategy. For example, the Best-Bin First method [35] combines a kd-tree with the priority queue of already checked reference image feature vectors, e.g., the SIFT descriptors [2]. It has been demonstrated [36] that the same approach works well for M-trees.

Another well-known ANN method is the locality sensitive hashing [37], originally used with binary data and the Hamming distance. The key idea is to apply several hash functions that return closed hashes for similar objects. It has recently been modified to work with an arbitrary metric and even with a symmetric dissimilarity measure which does not satisfy the triangle inequality [38]. Similar asymmetric hashing techniques are used in Google Correlate [23]. It is necessary to tune several parameters to use this method in an efficient way with integer features. The situation is much more difficult for floating-point features. For instance, in the popular FLANN library [39], it is possible to apply this method only to numerical data.

It is known that most of ANN techniques show good performance only if the first NN is quite different from other reference images [35]. Hence, their usage is restricted in many real-world applications. For instance, faces have a similar shape and common features (eyes, nose, mouth, etc.). Moreover, these methods are usually developed to approximately match a very large number (100 000) of image descriptors of extracted keypoints [39]. Hence, their performance is comparable with the brute-force method for medium databases (thousands of classes) of high-dimensional feature vectors.  A promising approach here is our directed enumeration method (DEM), which outperforms the known ANN methods in face recognition [27]. Its main disadvantage is the storage and processing of the full matrix of distances between all reference images.

The pivot-based methods can be applied to decrease the recognition speed for the thousands of classes [40]. They are implemented even for non-metric dissimilarity measures. In these methods several reference points (pivots) are chosen and distances from them to all reference images are computed. An interesting approach is the stretching of the triangle inequality [41]. Another pivot-based algorithm is the ordered permutations (perm-sort or permutation indexing) method [40]. According to it, several pivots are chosen, and a permutation of pivots is created for every reference data point. To classify the query point, one firstly explores the reference objects with the highest similarity between the permutation of this reference object and the permutation of the vector of distances between the query object and all pivots. Unfortunately, most popular ANN methods including the pivot-based search have a heuristic nature, especially in the case of non-metric distances. It is practically impossible to prove that a particular algorithm is optimal (in some sense) and nothing can be done to improve it.

It seems there is a lack of efficient ANN algorithms for non-traditional measures of image dissimilarity, which can be used in such difficult classification tasks as face recognition with a medium database. Hence, in this paper we investigate the possibility to apply statistical approach to our DEM [27] and look for the reference image with the maximal likelihood of the distances to previously checked reference images. We prove that this method implements a statistically optimal ANN search procedure and can be used with an arbitrary continuous dissimilarity measure, which satisfies identity of indiscernibles and non-negativity metric properties.

## 3. Statistical Image Recognition with the Directed Enumeration Method

In image recognition, it is required to assign an observed grayscale image *X* to one of *R* classes specified by the database of reference images (instances) $\{X_r\}, r \in \{1,\ldots,R\}$. Let all images have the same width *U* and height *V*. We compute the histogram $H_r = [h_1^{(r)},\ldots,h_N^{(r)}]$ of appropriate simple features for each reference image $X_r$. Here *N* is the number of bins in the histogram. Gradient orientation is a widely used simple feature of an image point, which is partially invariant to illumination changes. In such a case, $H_r$ is the HOG, perhaps weighted with the gradient magnitude [4]. It is possible to achieve scale and rotation invariance of this feature vector by using the known normalizing procedures applied in the SIFT algorithm [2]. The same procedure is repeated to compute the histogram $H = [h_1,\ldots,h_N]$ corresponding to the input image *X*. In this paper we assume that all feature vectors $H_r$ and *H* are normalized, so that they may be treated as probability distributions.

Let us *assume* that: 1) the reference images from different classes are independent; and 2) the probability distributions of the feature vectors from the same class are identical. In this case, statistical approach can be applied to solve image recognition problem. Having observed tendencies present in literature within the last decades we should say that most of successful statistical pattern recognition methods use the nonparametric approach to estimate the unknown probability densities of each class [42]. In this approach, a functional form of probability distributions is unknown and should be estimated by nonparametric techniques like the Parzen kernel [15]. In our paper [21] it was shown that statistical pattern recognition task should be reduced to the testing of *R complex* hypothesis $W_r, r \in \{1,\ldots,R\}$ about homogeneity of simple features (e.g., gradient orientations). In that case, the optimal (asymptotic minimax) maximal likelihood solution of statistical image recognition task is achieved with the NN rule

$$W_\nu : \nu = \underset{r \in \{1,..R\}}{\operatorname{argmin}} \rho(X, X_r), (1)$$

where the distance $\rho(X, X_r)$ is given from the output of the homogeneity testing probabilistic neural network (HT-PNN) [18,21]:

$$\rho_{HT-PNN}(X, X_r) = \sum_{i=1}^{N} \left( h_i \ln \frac{2h_{K;i}}{h_{K;i} + h_{K;i}^{(r)}} + h_i^{(r)} \ln \frac{2h_{K;i}^{(r)}}{h_{K;i} + h_{K;i}^{(r)}} \right) - \ln p_r, (2)$$

Here $p_r$ is the prior probability of hypothesis $W_r$, and $h_{K;i}^{(r)} = \sum_{J=1}^{N} K_{ij} h_j^{(r)}$ and $h_{K;i} = \sum_{J=1}^{N} K_{ij} h_j$ are the convolution of the feature vectors with the Parzen kernel $K_{ij} = K(\theta_i, \theta_j), i, j \in \{1,\ldots,N\}$, where $\theta_i, i \in \{1,\ldots,N\}$ is the quantized value of the primitive feature, e.g., the gradient orientation. The HT-PNN (2) represents the kernel-based generalization of the Jensen-Shannon divergence widely applied in various image processing tasks [43].

An obvious implementation of criterion (1) requires the brute force of the whole database $\{X_r\}$. To speed-up recognition, ANN methods can be applied [22], e.g., the DEM [27], in which the criterion (1) is converted to a range query [30]:

$$W_\nu : \rho(X, X_\nu) < \rho_0. \qquad (3)$$

This criterion defines the termination condition of the ANN method. According to the DEM [11] the first reference image $X_{r_1}, r_1 \in \{1,\ldots,R\}$ is randomly chosen. Next, it is put into the priority queue of reference images sorted by the distance to *X* similarly to the Best-Bin First kd-tree [35]. Next, the highest priority item $X_i$ is pulled from the queue and the set of the reference images $X_i^{(M)}$ is determined by using the following expression

$$\left(\forall X_k \in X_i^{(M)}\right)\left(\forall X_j \notin X_i^{(M)}\right) \left|\rho_{i,k} - \rho(X, X_k)\right| \le \left|\rho_{i,j} - \rho(X, X_j)\right|. \qquad (4)$$

The distance matrix $\mathrm{P} = [\rho_{i,j}]$ is computed at the preliminarily step of the DEM (similarly to the AESA [33]) to obtain the set $X_i^{(M)}$ (4). The distances between previously unchecked reference image from the set $X_i^{(M)}$ and the input image are computed. After that, every image from this set is put into the priority queue. The DEM is terminated if the condition (3) is satisfied for the next selected reference image (4) or after checking for $E_{\max}$ reference images. If the priority queue is implemented using heaps, and the maximal heap size is equal to $E_{\max} \cdot M$ , then the worst-case run-time complexity of the DEM can be estimated as follows: $O\left(E_{\max} \cdot (N + M \log(E_{\max} \cdot M)\right)$.

The search procedure (4) of the DEM uses the following heuristic: if there exists a reference image $X^*$ for which $\rho\left(X, X^*\right) < \rho_0 << 1$, then the following condition $\left|\rho\left(X, X_r\right) - \rho\left(X^*, X_r\right)\right| << 1$ is true with high probability for an arbitrary *r*-th reference image. However, the probability that the reference image is the NN of *X* can be directly estimated for the HT-PNN (2). We explore this idea in the next section.

## 4. Proposed Approach

### 4.1. Maximum-Likelihood Approximate Nearest Neighbor Method

Let the reference images $X_{r_1},\ldots,X_{r_k}$ have been checked before the *k*-th step, i.e. the distances $\rho\left(X, X_{r_1}\right),\ldots,\rho\left(X, X_{r_k}\right)$ have been computed. In this paper we propose to choose the next most probable instance $X_{r_{k+1}}$ with the maximum-likelihood method [15]. By using our assumptions about identical prior probabilities and independence of the reference images from different classes, the maximal likelihood choice can be written as follows:

$$r_{k+1} = \underset{\nu \in \{1,\dots,R\} - \{r_1,\dots,r_k\}}{\operatorname{argmax}} \left( p_\nu \cdot \prod_{i=1}^{k} f\left( \rho(X, X_{r_i}) \middle| W_\nu \right) \right), \qquad (5)$$

where $f\left(\rho(X, X_{r_i}) \middle| W_\nu\right)$ is the conditional density of the distance $\rho\left(X, X_{r_i}\right)$ if the hypothesis $W_\nu$ is true (the class label of the observation *X* is $\nu$ ). According to criterion (5), if the prior probabilities of all classes are not identical, then the next instance will be selected from the majority classes with high probability. To estimate the likelihood $f\left(\rho(X, X_{r_i}) \middle| W_\nu\right)$, the asymptotic properties of the HT-PNN distance can be used.

Based on the known asymptotic distribution of the maximal likelihood statistics [25], it is possible to show that if hypothesis $W_\nu$ is true, then the HT-PNN distance (2) $UV \cdot \rho_{HT-PNN}(X, X_r)$ will be asymptotically distributed as a noncentral chi-square with (*N*-1) degrees of freedom and the noncentrality parameter $UV \cdot \rho_{\nu,r}$ [25]. Here *UV* is the total size of the sample used to estimate the probability density *H*. In fact, the noncentral chi-squared distribution is widely used in the model evaluation and power analysis of the likelihood ratio tests [44]. Though the noncentral chi-squared distribution of the maximal likelihood test statistics is only reached in asymptotic, the simulation results have shown that this approximation is valid under reasonable misspecifications and models, even for small (50-100) samples [45]. According to the normal approximation of the non-central chi-squared distribution [46], we can assume, that the distance $\rho\left(X, X_r\right)$ is normally distributed $N\left( \rho_{\nu,r} + \frac{N-1}{UV}; \frac{4UV \cdot \rho_{\nu,r} + 2(N-1)}{(UV)^2} \right)$ in asymptotic, if *X* corresponds to the class $\nu$ . The normal approximation of the distribution of distance between feature vectors is quite famous and can be applied in various dissimilarity-based pattern recognition tasks [47], including the matching of SIFT descriptors [48]. Thus, the likelihood $f\left(\rho(X, X_{r_i}) \middle| W_\nu\right)$ in (5) can be simply written as the probability density function of the normal distribution with mean $\rho_{\nu,r_i} + \frac{N-1}{UV}$ and standard deviation $\frac{\sqrt{4UV \cdot \rho_{\nu,r_i} + 2(N-1)}}{UV}$

:

$$f\left( \rho(X, X_{r_i}) \middle| W_\nu \right) = \frac{UV}{\sqrt{2\pi \cdot (4UV \cdot \rho_{\nu,r_i} + 2(N-1))}} \exp\left[ - \frac{UV\left( \rho(X, X_{r_i}) - \rho_{\nu,r_i} - \frac{N-1}{UV} \right)^2}{8\rho_{\nu,r_i} + \frac{4(N-1)}{UV}} \right].$$

By using an obvious identity $\sqrt{4UV \cdot \rho_{\nu,r_i} + 2(N-1)} = \exp\left[\frac{1}{2}\ln\left(4UV \cdot \rho_{\nu,r_i} + 2(N-1)\right)\right]$, this equation can be written as follows:

$$f\left(\rho(X,X_{r_i})\middle| W_\nu\right) = \frac{UV}{\sqrt{2\pi}}\exp\left[-\frac{1}{2}\ln\left(4UV \cdot \rho_{\nu,r_i} + 2(N-1)\right) - \frac{UV\left(\rho(X,X_{r_i}) - \rho_{\nu,r_i} - \frac{N-1}{UV}\right)^2}{8\rho_{\nu,r_i} + \frac{4(N-1)}{UV}}\right]. \quad (6)$$

If we divide equation (6) by a constant $\left(UV/\sqrt{2\pi}\right)^k$, take a natural logarithm and divide it by *UV*/8, expression (5) is finally transformed to the following form

$$r_{k+1} = \underset{\mu \in \{1,\ldots,R\} - \{r_1,\ldots,r_k\}}{\operatorname{argmin}} \left(\sum_{i=1}^{k} \varphi_\mu(r_i) - \ln p_\mu\right). \quad (7)$$

where

$$\varphi_\mu(r_i) = \frac{\left(\rho(X,X_{r_i}) - \rho_{\mu,r_i} - \frac{N-1}{UV}\right)^2}{\rho_{\mu,r_i} + \frac{N-1}{2UV}} + \frac{4}{UV}\ln\left(2\rho_{\mu,r_i} + \frac{N-1}{UV}\right). \quad (8)$$

As the average image size is usually much higher than the number of parameters *UV*>>(*N*-1), then the function in (8) can be simplified:

$$\varphi_\mu(r_i) \approx \frac{\left(\rho(X,X_{r_i}) - \rho_{\mu,r_i}\right)^2}{\rho_{\mu,r_i}}. \quad (9)$$

This equation is in good agreement with the heuristic from the original DEM [27]: the closer are the distances $\rho(X,X_{r_i})$ and $\rho_{\mu,r_i}$ and the higher is the distance between reference images $X_\mu$ and $X_{r_i}$, the lower is $\varphi_\mu(r_i)$ .

If the distance $\rho(X,X_{r_{k+1}})$ is lower than the threshold $\rho_0$ (3), then the search procedure is terminated at the $K_{checks} = k+1$ step. Otherwise, the reference image $X_{r_{k+1}}$ is put into the set of previously checked reference images and the maximal likelihood search procedure (7), (9) is repeated.

The proposed maximum-likelihood ANN (ML-ANN) method (3), (7), (9) for the HT-PNN discrimination (2) is different from the baseline DEM in the rule of choosing the next reference image (7), (9). In the DEM, *M*>>1 instances are chosen (4), while in the proposed ML-ANN, only one reference image is selected (7). Finally, unlike the heuristic DEM, the maximal likelihood reference image is selected at each step (5) of the proposed ML-ANN based on the available information about the previously computed distances $\rho\left(X, X_{r_1}\right),\ldots,\rho\left(X, X_{r_k}\right)$.

### 4.2. Pivot-Based Maximum-Likelihood Approximate Nearest Neighbor Method

Though the proposed ML-ANN method makes it possible to minimize the average number of distance computations $K_{checks}$, it has two significant disadvantages:

1) It is necessary to compute and store the distances between all reference images, i.e. the ML-ANN requires quadratic memory space [34];

2) The complexity of extra computation at each step is rather high. For instance, $\varphi_\mu(r_i)$ (9) is calculated for each unchecked $\mu$-th reference image, i.e., the total number of computations of $\varphi_\mu(r_i)$ during the whole procedure is equal to $(R-1)+\ldots+(R-K_{checks}+1)=\frac{2R-K_{checks}}{2}\cdot(K_{checks}-1)$. As a result, the total recognition speed of the ML-ANN can be low, especially for simple dissimilarity measures and/or medium dimension size of the feature vector. Thus, in this section we propose a modification of the ML-ANN, which does not provide the best number of distance computations, but is much more suitable for practical applications.

First, let us choose *p*<<*R* reference images (pivots) $\left\{X_{r_1},\ldots,X_{r_p}\right\}$ from the database. It is known that the original DEM shows better performance if the reference images are far from each other [27]. Hence, it is necessary to choose the pivots $\left\{X_{r_1},\ldots,X_{r_p}\right\}$ to be as distant from each other as possible. This task can be solved with the known clustering algorithms. As the pivots should be chosen from the given database, the k-medoids method seems to be the best choice [15]. Unfortunately, this method requires computation of the complete distance matrix $\mathrm{P}=[\rho_{i,j}]$. Moreover, the k-medoids method focuses on looking for clusters of points close to each other. Hence, the inter-cluster distance is ignored.

According to Bustos et al. [26], an incremental selection of pivots is the most effective solution. Hence, in this paper we propose applying the simple farthest-first traversal procedure [49]. The first pivot $X_{r_1}$ is chosen randomly. Next, the distances between this reference image and all other reference images are computed, and the most distant reference image is selected as the second pivot. The procedure is repeated so that every pivot is characterized with the highest sum of distances to the previous pivots:

$$r_{i+1}=\underset{\mu\in\{1,\ldots,R\}-\{r_1,\ldots,r_i\}}{\arg\max}\sum_{j=1}^{i}\rho(X_\mu,X_{r_j}),\, i\in\{1,\ldots,p-1\}. \tag{10}$$

The result of the initialization phase is a list of pivots and $R \times p$ matrix of distances between all reference images and the pivots.

The following algorithm is applied to recognize the image *X*. First, the distances to all pivots $\rho(X, X_{r_i}), i=\overline{1,p}$ are computed. If none of the distances exceeds the threshold (3), the search procedure is terminated. Otherwise, the log-likelihoods for all other reference images are estimated similar to (7), (9):

$$\varphi_{\Sigma;\mu} = \sum_{i=1}^{p} v_{\mu}(r_i) - \ln p_{\mu}, \mu \in \{1,\dots,R\} - \{r_1,\dots,r_p\}. \qquad (11)$$

An array $\{\varphi_{\Sigma;\mu}\}$ is sorted in ascending order, and the maximal likelihood reference images are chosen sequentially as long as the number of distance computations does not exceed $E_{\max}$ or the condition (3) is satisfied for the next checked reference image.

The complete pivot-based ML-ANN (P-ML-ANN) algorithm is described in Table 1.

The worst-case runtime and space (memory) complexities of described methods are shown in Table 2. Here *R* is the number of images stored in the database, *N* is the size of the feature vectors, and *p* is the number of pivots. The complexity of the state-of-the-art Ordering permutations method from paper [40] is also presented in Table 2. In fact, the pivot selection procedure in this algorithm is very similar to the proposed approach (Table 1).

As one can see, the main loop (step 7 of the algorithm in Table 1) includes only the computation of the distances between the next instance and the image *X*, i.e. it is as fast as the brute force (NN search). In addition to the $R \times p$ distances to all pivots, the P-ML-ANN requires $2(R-p)$ extra memory for storing the floating-point and integer numbers for $\{\varphi_{\Sigma;\mu}\}$ and the ordered indices $\{r_i\}$, respectively. In fact, it is much lower in comparison with the quadratic space requirements (Table 2) of the ML-ANN described in the previous subsection. Hence, the proposed modification of the ML-ANN deals with both its drawbacks mentioned above.

We show only worst-case time complexity of the proposed procedures in Table 2. The average-case time complexity of the P-ML-ANN $O(K_{checks} \cdot N + p \log p)$should be much lower, because $K_{checks} << E_{\max}$ in general case. However, the P-ML-ANN method does not provide the greedy maximal likelihood solution anymore, because it does not use all the incoming information about the distances to other reference images except the pivots. Hence, the average number of distance computations $K_{checks}$ of the ML-ANN should be slightly lower than the number of checks $K_{checks}$ in the proposed modification (Table 1). Nevertheless, as the main loop in the P-ML-ANN is very simple and the number of pivots *p* is rather low, the image recognition with the P-ML-ANN should be approximately (1+*R*/*N*)-times faster than the ML-ANN (Table 2). The next section provides experimental evidence to support this claim.

### 5. Experimental Results

Our experimental study is devoted to face recognition [14]. We compare the performance of the proposed ML-ANN and its modification P-ML-ANN (Table 1) with the baseline DEM (3), (4), brute force and several ANN methods from the FLANN [39] and NonMetricSpaceLib [30] libraries that showed the best speed, namely:

1. Randomized kd-tree with 4 trees [50] from the FLANN library;
2. Ordering permutations (perm-sort) from NonMetricSpaceLib which is known to increase the recognition speed for medium databases (thousands of reference images) [40].

We evaluate the error rate (%) and the average time (ms) to recognize one test image by using a laptop (4 core i7 2 GHz CPU, 6 Gb RAM) and Visual C++ 2013 Express compiler (x64 environment) with optimization by speed. We explore an obvious way to improve performance by using parallel computing [18]. Namely, the whole training set is divided into *T* non-overlapping parts and each part is processed in its own task. All tasks work in parallel and terminate right after any task finds the solution (3). Each task is implemented as a separate thread by using the Windows ThreadPool API. We analyze both the nonparallel ($T$=1) and the parallel cases ($T$=8).

The faces were detected with the Local Binary Pattern cascade classifier [16] from OpenCV library. The median filter with the window size (3x3) was applied to remove noise in detected faces. The faces were divided into 100 (10x10) disjoint segments ( $K_1 = 10$ rows and $K_2 = 10$ columns) of the same size. The weighted HOG features are estimated with the procedure described in detail in [27]. These features are scale and illumination invariant. The rotation invariance is not usually required in face recognition, because the orientations of the frontal faces are practically identical, and small rotation variations can be handled by quantization of gradient orientations. In this experiment the number of bins in the HOG is equal to 8. Hence, the feature vector size $N = 8 \cdot 10 \cdot 10 = 800$ . The Gaussian Parzen kernel with the smoothing parameter σ=0.577 was used to estimate the unknown likelihoods in the HT-PNN (2). The prior probabilities of each class are assumed to be identical. The false accept rate of termination condition (3) is set to 1%. The parameter $E_{max}$ for all ANN methods is chosen to achieve the recognition accuracy, which is not 0.5% less than the accuracy of an exhaustive search. If such accuracy cannot be achieved, $E_{max}$ is set to be equal to the count *R* of the reference images assigned to each task.

The distance in the NN rule (1) is computed with the mutual alignment of the HOGs in the Δ-neighborhood in order to take into account the small spatial deviations due to misalignment after face detection [18]:

$$\rho(X, X_r) = \frac{1}{K_1 K_2} \sum_{k_1=1}^{K_1} \sum_{k_2=1}^{K_2} \min_{|\Delta_1| \le \Delta, |\Delta_2| \le \Delta} \rho(H(k_1, k_2), H_r(k_1 + \Delta_1, k_2 + \Delta_2)) \quad (12)$$

The following neighborhood sizes were tested: Δ=0 and Δ=1 [18]. The Euclidean metric and the HT-PNN (2) distance were used to match the HOGs of each segment in (12).

At first, the FERET (http://www.itl.nist.gov/iad/humanid/feret/feret_master.html) dataset was used. This dataset is the standard-de-facto in comparison of the *constrained* face recognition algorithms [51]. The FA set of $R=1365$ frontal images of 994 persons populates the database (i.e. the training set), the FB set of 1355 frontal photos of the same persons formed a test set. The average error rates and recognition time of the NN rule and the state-of-the-art SVM classifier from libSVM library are shown in Table 3.

Table 3. Face recognition results, FERET dataset

An alignment of the HOGs ($\Delta$=1) provides an excellent error rate for the FERET FA/FB task (compare with 2.1% for boosted recognition with the Local Binary Features in [16]). Moreover, the accuracy of SVM is 4% lower, than the accuracy of the NN rule with the Euclidean distance. In fact, the one-versus-all implementation of multiclass SVM requires the brute force of all classes, hence, it is an order of magnitude slower (Table 3), than the simple NN classifier, if the number of classes $C$ is rather high and the number of reference instances per each class is very low (only 1 instance for most of the classes). Finally, we experimentally support the statement [21] that the error rate for the state-of-the-art Euclidean distance exceeds the error rate of the HT-PNN (2).

In the first experiment, we examine the dependence of the recognition time on the choice of pivots in the P-ML-ANN. We implement three strategies: 1) random choice of pivots; 2) k-medoids; and 3) farthest-first traversal (10). The dependence of the recognition time on the number of pivots $p$ for the HT-PNN (2), (12) in the nonparallel case ($T$=1) is shown in Fig. 1 and Fig. 2. We should note that the ANN (3) is always returned in the P-ML-ANN if $E_{max}=R$. As we stated earlier, the parameter $E_{max}$ is chosen to provide appropriate accuracy. Hence, it is not necessary to demonstrate recognition accuracy, as it is approximately equal in all cases (3.91%-4.13% for $\Delta = 0$ and 1.77%-1.99% for $\Delta = 1$).

Fig. 1. Dependence of the average recognition time (ms) on the number of pivots $p$, HT-PNN (2) distance, FERET dataset, $\Delta$=0.

Fig. 2. Dependence of the average recognition time (ms) on the number of pivots $p$, HT-PNN (2) distance, FERET dataset, $\Delta$=1.

First of all, the farthest-first traversal strategy provides the lowest recognition time in most cases. On average, random choice seems to be appropriate. However, the standard deviation of recognition time in this case is usually 2-3 times higher in comparison with the k-medoids and the farthest-first traversal. The state-of-the-art k-medoids clustering is not the best choice as it ignores the inter-cluster distances. Moreover, the pivot selection time in the k-medoids is much higher. Even if the distance matrix is preliminarily calculated, the k-medoids usually requires 30-

40 ms to create the set of pivots. On the contrary, the processing time in the farthest-first traversal is equal to 2-6 ms. However, in practice the distance matrix $\mathrm{P}=[\rho_{i,j}]$ should be computed together with the pivot selection. In this case, the k-medoids method requires 11.8 seconds (Δ=0) and 82.8 seconds (Δ=1). The farthest-first traversal needs only 3.2 ms (Δ=0) and 4.4 ms (Δ=1) for selection of $p$=32 pivots.

Second, in both cases (Δ=0 and Δ=1) the optimal number of pivots for the farthest-first traversal strategy is approximately equal to 24 (1.8% of the number of images $R$ in the training set). However, these diagrams (Fig. 1, 2) are quite different: the loss in performance in comparison with the optimal case is not as crucial in the simplest case (Δ=0) as in the complex one (Δ=1) for a low number of pivots. At the same time, the recognition speed varies not as dramatically in the latter case as for the former one for large values of $p$. Moreover, if Δ=1, one can see some improvement in the speed for $p$=64. Our final conclusion here is that, without a doubt, a proper choice of the number of pivots significantly affects the performance of the P-ML-ANN.

In the next experiment, we compare in detail the performance of our ML-ANN and P-ML-ANN (Table 1) with the baseline DEM [27]. The number of reference images $M$ to check at each step (4) is equal to 64 and 16 in the nonparallel case ($T$=1) and the case of $T$=8 threads, respectively. The number of pivots is equal to $p$=24 in the nonparallel case and $p$=5 in the parallel one. The count of computed distances $K_{checks}$, recognition time and error rate depending on the maximal number of checks ($E_{max}$) are shown in Fig. 3-5, respectively.

Fig. 3. Dependence of the count of computed distances $K_{checks}$ on $E_{max}$, HT-PNN distance (2), FERET dataset, Δ=1.

Fig. 4. Dependence of the average recognition time (ms) on $E_{max}$, HT-PNN distance (2), FERET dataset, Δ=1.

Fig. 5. Dependence of the error rate (ms) on $E_{max}$, HT-PNN distance (2), FERET dataset, Δ=1.

Here, first, the average number of reference images to be checked by the ML-ANN is the lowest in *all* cases (Fig. 3). Namely, $K_{checks}$ of the P-ML-ANN is only 1.1-1.3-times higher, than $K_{checks}$ of the ML-ANN. As the P-ML-ANN is a slight modification of the ML-ANN, it is 1.1-1.5-times faster, than the heuristic DEM.

Second, the complexity of additional processing (11) at each stage of the ML-ANN is rather high (Table 2). Hence, an average recognition time (Fig. 4) of the ML-ANN in most cases is 5-15% higher in comparison with the P-ML-ANN. The ML-ANN's performance is slightly better only if threshold $E_{\max}$ is very low (10% of the training set size $R$). It can be explained by extra processing needed in the initialization of our method (Table 1), namely, the computations of the distances to all pivots and the sorting of $\varphi_{\Sigma;\mu}$ (11). The run-time complexity of these computations is equal to $O(p\cdot(E_{\max}+\log p))$. However, the error rate (Fig. 5) of the ML-ANN in this case (

$E_{\max} = 0.1 \cdot R$ ) – 4.47% – is much higher, than the error rate 1.47% (Table 3) of the NN rule (1). In the same case, the proposed P-ML-ANN provides a lower error rate (3.58%).

Finally, all methods (DEM, ML-ANN and P-ML-ANN) implement the search procedure, which is similar to the most of known ANN algorithms [40]. Namely, two steps are repeated: selection of the next reference image from the database, and termination if the number of distance computations exceeds $E_{\max}$. Hence, all curves in Fig. 3,4 are nearly equal. The higher is the threshold $E_{\max}$, the faster is the ANN search, and so the linear increase is expected here. However, the count of distance computations $K_{checks}$ is usually lower than $E_{\max}$ (e.g., $K_{checks} \approx 0.25R$ for the P-ML-ANN, when $E_{\max} = R$, see Fig. 5), because the search procedure is terminated in most cases, when the reliable decision (3) is found. At the same time, it is important to notice, that the DEM's error rate (Fig. 5) decreases quite slowly in comparison with the proposed approach (ML-ANN and P-ML-ANN). In fact, our maximum-likelihood search procedure (7), (9) converges to an appropriate solution much faster than the heuristics of the known ANN methods.

In the next experiment, we compare the proposed method with other state-of-the-art ANN algorithms. The average recognition time per one test image (in ms) for Δ=0 and Δ=1 is shown in Fig. 6 and Fig. 7, respectively.

Fig. 6. Average face recognition time (ms), HT-PNN (2) distance, FERET dataset, Δ=0.

Fig. 7. Average face recognition time (ms), HT-PNN (2) distance, FERET dataset, Δ=1.

Here the randomized kd-tree is characterized with even worse performance than the brute force in both cases (Δ=0 and Δ=1). The parallel implementation makes it possible to improve the recognition speed. It is obvious that the gain in performance is maximal for the simplest algorithm, namely, the exact NN search (12). However, even in this case the parallel implementation is only 3.8 times better, although we used $T$=8 parallel tasks.

As it was expected for medium databases, the ordered permutations (perm-sort) method in the nonparallel case showed twice lower recognition time than the brute force for Δ=0 and it is even 4.5 times faster than the exhaustive search if Δ=1. The performance of the proposed ML-ANN is 1.7-2 times better than for the perm-sort and 3-7.5 times better than for the brute force. However, the recognition time is only 20% better than for the ordered permutations in the parallel case with the histogram alignment (Δ=1).

The performances of the proposed ML-ANN and the original DEM are similar in the simplest case (Δ=0, $T$=1). However, the recognition time of the ML-ANN is twice lower than the DEM time for their parallel implementations. The situation changes dramatically for the HOGs alignment (Δ=1). In this case, the DEM's performance is even 1.5-times lower than the performance of the ordered permutations if $T$=1. Both methods are heuristic and are specially designed for medium databases. Hence, they are expected to compete with each other. In fact, the ML-ANN here is 2.5 times better than the baseline DEM.

Finally, the most significant note here is that the proposed P-ML-ANN is 6-12 times faster than the brute force in the nonparallel case and 4 times faster for $T$=8 tasks. Moreover, it is 30-80% faster than the ML-ANN, although the number of computed distances in the ML-ANN is 14% and 8% higher than $K_{checks}$ in the P-ML-ANN for Δ=0 and Δ=1, respectively. In fact, this result can be explained by the simplification of the main cycle (Table 1) in the P-ML-ANN and the usage of the small part ( $p/R \cdot 100\% \approx 1.8\%$ ) of the matrix of distances between the reference facial images.

The recognition time for the Euclidean metric is shown in Fig. 8 and Fig. 9 for Δ=0 and Δ=1, respectively.

Fig. 8. Average face recognition time (ms), Euclidean distance, FERET dataset, Δ=0.

Fig. 9. Average face recognition time (ms), Euclidean distance, FERET dataset, Δ=1.

Unlike the previous experiment with the HT-PNN distance, the implementation of the Euclidean metric is much faster. Hence, an additional processing in the ML-ANN is high in comparison with the complexity of the distance computation (12). As a result, its performance is equal to the randomized kd-trees for the simplest case (Δ=0). Even in the parallel case, the ML-ANN is slower than either the ordered permutations or the original DEM. However, when the distance becomes more complex (Δ=1), the lower number of distance computations $K_{checks}$ leads to the better performance of the ML-ANN. However, the ordered permutations method seems to be the best choice even in this case.

Second, the recognition speed of the exact NN search (12) is 30-40% higher than the speed of the randomized kd-tree, i.e. the former approximate method may be preferred in this case. Third, the performance of the perm-sort is twice better than the ML-ANN in the nonparallel case, although the number of distance computations for the ML-ANN is 1.8 times lower. Fourth, the speed of the proposed modification (P-ML-ANN) is maximal (3-8 times higher than the speed of brute force). Its performance is 50-80% better than the performance of the ordered permutations for the complex case of the HOGs alignment (Δ=1) and is slightly (21%) higher for the conventional dissimilarity computation (Δ=0) and the nonparallel case.

In the next experiment we examine the behavior of the proposed ANN method for the Essex dataset (http://cswww.essex.ac.uk/mv/allfaces/index.html). The training set contains $R=5187$ frontal images of 323 persons, the testing set consists of 1222 other photos. The recognition results of the 20-times repeated random subsampling cross validation are shown in Table 4.

As the dataset is much easier for recognition than the FERET (compare Table 3), and most of the photos of each person are very similar, the error rate is identical for all tested dissimilarity measures (12). Hence, we decided to conduct an additional experiment. We perform single-linkage clustering for each class (person). The distance between two reference images inside one cluster does not exceed $\rho_0$ (3). The training set is filled with *R=881* medoids of each cluster. The error rates for the clustered dataset are also shown in Table 4.

As one can notice, SVM is again one of the worst classifiers, because it cannot provide the highest accuracy, and its performance is lower, when compared to the NN rule (12) with the Euclidean distance for the clustered training set. However, the recognition time of SVM is lower for the initial dataset, which contains a large number of instances for each person. The error rate of the NN rule (12) is slightly lower, when the HOGs alignment ($\Delta$=1) is used. However, this alignment dramatically increases the recognition complexity. Hence, we do not explore the case of $\Delta$=1 in the next comparison of our P-ML-ANN with other ANN methods. An average recognition time for the HT-PNN (2), (12) for the original and clustered training sets is presented in Fig. 10 and Fig. 11, respectively.

Fig. 10. Average face recognition time (ms), HT-PNN (2) distance, Essex dataset, $\Delta$=0.

Fig. 11. Average face recognition time (ms), HT-PNN (2) distance, Essex (clustered) dataset, $\Delta$=0.

The results for the original training set (Fig. 10) significantly differ from the previously reported results for the FERET dataset. First, the performance of the randomized kd-trees in the nonparallel implementation is excellent: the average recognition time is 2.5 and 1.3 times lower in comparison with the perm-sort and the proposed ML-ANN, respectively. It can be explained by either a larger training set (in the first case, Fig. 11) or by the well-expressed clustered structure of the dataset. However, the proposed P-ML-ANN is the best choice even in this case. It is 30 times more effective than the brute force. It is remarkable that the performance of the P-ML-ANN is slightly worse than the ML-ANN in the parallel implementation. In fact, it seems that the decision (3) can be made in the early stages with low $E_{\max}$ in this particular case. We demonstrated (Fig. 2) that in this case the ML-ANN is more effecient due to the low complexity of its initialization procedure.

The results (Fig. 11) with clustering are quite similar. The randomized kd-tree is 71% better than the perm-sort in the nonparallel case and a bit worse in the parallel one. However, the proposed ML-ANN is 30% more effective than the randomized kd-tree even in the nonparallel case. The recognition time of the ML-ANN is also 7-17% lower than the average time of the original DEM. The P-ML-ANN is again the best choice in both the parallel and nonparallel implementations. It is 9-18-times more effective than the brute force and 75-98% better then the second best variant, namely, the ML-ANN. It is also interesting to note that although the size of the training set was practically 6 times lower after clustering, the average recognition speed of the P-ML-ANN is only 3.8 times lower (compare Fig. 10 and 11). It seems, that the proposed ML-ANN and its modification P-ML-ANN makes it possible to partially extract the "clustered" structure of the data [18,52] as it is done in the randomized kd-tree.

In the last experiment we examine the potential application of the proposed approach in unconstrained face recognition [51] for the complex LFW dataset (http://vis-www.cs.umass.edu/lfw/) [53]. We took *C*=1680 persons from this database, which have two or more photos. The resulted dataset with 9034 facial photos was divided into the training and the test sets with $R$=4571 and 4463 instances, respectively. We used the DNN model of the Oxford's Visual Geometry Group [20] preliminarily trained with the Caffe framework [54]. The 4096 non-negative features extracted with this DNN were normalized in order to treat them as probability distributions. The average error rate of the NN rule with the Euclidean metric was equal to 14.8%. This error rate is even lower, than the known results from existing literature [7,55]. The HT-PNN classifier makes it possible to increase the accuracy in 0.3-0.5%, though its performance is 3-4 times worse. Moreover, the number of features (4096) and the database size (4571 instances) are rather high, so the performance of the brute force can be insufficient even for the state-of-the-art Euclidean distance. Hence, we report the results (Fig. 12) for this practically important case.

Fig. 12. Average face recognition time (ms), Euclidean distance, DNN features, LFW dataset.

Here the proposed approach is 3.5-5.5 times faster, than the brute force. The recognition speed of the P-ML-ANN is twice higher, than the speed of the ordered permutations. The proposed modification (P-ML-ANN) is preferable, than the original ML-ANN, especially, in the non-parallel case ($T$=1), when the P-ML-ANN is 1.5-times faster. The parallel computing with $T$=8 threads does not provide the significant gain in classification time for most of ANN methods, because the Euclidean dissimilarity measure is computed very efficiently. Thus, these results support the statement, that the proposed ML-ANN and its modification (P-ML-ANN) are the most efficient ANN methods even for the feature vectors, which do not represent the histogram of any primitive image feature.

## 6. Conclusion

In this paper, we introduced the novel ANN method with the maximal likelihood search of the next reference instance (7), (9) in order to improve the performance of the NN-based image recognition methods. We proposed its modification P-ML-ANN to make our approach suitable for practical applications. Several pivots are chosen at the preprocessing stage of the P-ML-ANN method. Hence, the main cycle of the search procedure (Table 1) becomes as fast as the brute force. Although this approach does not achieve the lowest average number of computed distances as the ML-ANN, it is more computationally efficient in most cases. Our modification requires only linear memory space to store the distances between pivots and all reference images. An obvious drawback of this procedure is the need for a proper choice of the number of pivots. We experimentally demonstrated that this choice could significantly improve the recognition performance. It is especially important that our approach is very efficient even in the case of the DNN-based features and the Euclidean distance (Fig. 12).

It is known [29] that the pivot-based techniques have several limitations in the processing of high-dimensional data and very large datasets. Even in our experiments, the performance of the pivot-based ordering permutations decreases, when the large Essex dataset is applied. Thus, the main direction for further research of the P-ML-ANN is exploring the bounds of its usage. It is evident that the quadratic memory requirement of the proposed ML-ANN limits its application to medium databases. However, it is not so obvious for its modification (Table 1). Based on the results of the paper [29], we expect that our approach is not as effective as the previously suggested ANN methods for very large databases, especially if the simple dissimilarity measures (e.g., Euclidean) are used. At the same time, it is difficult to estimate in advance what is the best approach to be used in a particular case. Thus, it is necessary to explore the possibility of automatic algorithm configuration [39] to fuse the P-ML-ANN with any appropriate ANN algorithm. Moreover, the complexity of the training database is not only dependent on its size and the number of images per class, it also depends on the confusability of the recognized classes. It is known [18,56] that an expected recognition accuracy for particular database can be estimated in statistical pattern recognition (including the HT-PNN). Hence, it is possible to compute an expected number of distance computations $K_{checks}$ for our search procedure (7), (9) and examine its dependence on the database confusability similarly to the studies in speech recognition [57].

Another possible direction of research is the termination condition (3). The ML-ANN and P-ML-ANN are used for the classification task only and the threshold $\rho_0$ is estimated by using the inter-class distances. In fact, our approach differs from the known ANN methods used to look for the NN without the need for any information about class labels. Hence, it is not easy to apply our methods to such tasks as a quick search for 2-NN SIFT descriptors [35,39]. It is necessary to test other termination conditions that do not require class labels. Finally, it is possible to implement the proposed ANN method for each image resolution in sequential processing [32] of the pyramid HOGs [5].

**Acknowledgement**

The work is partially supported by Laboratory of Algorithms and Technologies for Network Analysis, National Research University Higher School of Economics.

**References**

[1] S. Prince, Computer Vision: Models, Learning, and Inference, Cambridge University Press, New York, 2012.

[2] D.G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision. 60 (2004) 91–110. doi:10.1023/B:VISI.0000029664.99615.94.

[3] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-Up Robust Features (SURF), Computer Vision and Image Understanding. 110 (2008) 346–359. doi:10.1016/j.cviu.2007.09.014.

[4] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, 2005: pp. 886–893.

[5] A. Bosch, A. Zisserman, X. Munoz, Representing Shape with a Spatial Pyramid Kernel, in: Proceedings of the 6th ACM International Conference on Image and Video Retrieval, ACM, New York, NY, USA, 2007: pp. 401–408.

[6] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, DeepFace: Closing the Gap to Human-Level Performance in Face Verification, in: Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, 2014: pp. 1701–1708. doi:10.1109/CVPR.2014.220.

[7] E.G. Ortiz, B.C. Becker, Face recognition for web-scale datasets, Computer Vision and Image Understanding. 118 (2014) 153–170. doi:10.1016/j.cviu.2013.09.004.

[8] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature. 521 (2015) 436–444. doi:10.1038/nature14539.

[9] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition,

Proceedings of the IEEE. 86 (1998) 2278–2324. doi:10.1109/5.726791.

[10] P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: An evaluation of the state of the art, IEEE Transactions on Pattern Analysis and Machine Intelligence. 34 (2012) 743–761.

[11] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks. 61 (2015) 85–117. doi:10.1016/j.neunet.2014.09.003.

[12] S.J. Raudys, A.K. Jain, Small Sample Size Effects in Statistical Pattern Recognition: Recommendations for Practitioners, IEEE Transactions on Pattern Analysis and Machine Intelligence. 13 (1991) 252–264. doi:10.1109/34.75512.

[13] X. Tan, S. Chen, Z.-H. Zhou, F. Zhang, Face recognition from a single image per person: A survey, Pattern Recognition. 39 (2006) 1725–1745.

[14] Y. Zhao, Y. Liu, Y. Liu, S. Zhong, K.A. Hua, Face recognition from a single registered image for conference socializing, Expert Systems with Applications. 42 (2015) 973–979. doi:10.1016/j.eswa.2014.08.016.

[15] S. Theodoridis, K. Koutroumbas, Pattern Recognition, 4th ed., Academic Press, Burlington, MA; London, 2008.

[16] G. Zhang, X. Huang, S.Z. Li, Y. Wang, X. Wu, Boosting Local Binary Pattern (LBP)-Based Face Recognition, in: S.Z. Li, J. Lai, T. Tan, G. Feng, Y. Wang (Eds.), Advances in Biometric Person Authentication, Springer Berlin Heidelberg, 2005: pp. 179–186.

[17] E. Zhou, Z. Cao, Q. Yin, Naive-Deep Face Recognition: Touching the Limit of LFW Benchmark or Not?, CoRR. abs/1501.04690 (2015). http://arxiv.org/abs/1501.04690 (accessed February 26, 2015).

[18] A.V. Savchenko, Search Techniques in Intelligent Classification Systems, Springer International Publishing, New York, NY, 2016.

[19] Z. Zhang, C. Wang, Y. Wang, Video-Based Face Recognition: State of the Art, in: Z. Sun, J. Lai, X. Chen, T. Tan (Eds.), Biometric Recognition, Springer Berlin Heidelberg, 2011: pp. 1–9.

[20] O.M. Parkhi, A. Vedaldi, A. Zisserman, Deep face recognition, Proceedings of the British Machine Vision. 1 (2015) 6.

[21] A.V. Savchenko, Probabilistic neural network with homogeneity testing in recognition of discrete patterns set, Neural Networks. 46 (2013) 227–241.

[22] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, A.Y. Wu, An Optimal Algorithm for Approximate Nearest Neighbor Searching Fixed Dimensions, Journal of the ACM. 45 (1998) 891–923. doi:10.1145/293347.293348.

[23] D. Vanderkam, R. Schonberger, H. Rowley, S. Kumar, Nearest Neighbor Search in Google Correlate, Google, 2013. http://www.google.com/trends/correlate/nnsearch.pdf.

[24] A.V. Savchenko, An Optimal Greedy Approximate Nearest Neighbor Method in Statistical Pattern Recognition, in: M. Kryszkiewicz, S. Bandyopadhyay, H. Rybinski, S.K. Pal (Eds.), Pattern Recognition and Machine Intelligence, Springer International Publishing, 2015: pp. 236–245.

[25] S. Kullback, Information Theory and Statistics, Dover Publications, Mineola, N.Y, 1997.

[26] B. Bustos, G. Navarro, E. Chávez, Pivot selection techniques for proximity searching in metric spaces, Pattern Recognition Letters. 24 (2003) 2357–2366. doi:10.1016/S0167-8655(03)00065-5.

[27] A.V. Savchenko, Directed enumeration method in image recognition, Pattern Recognition. 45 (2012) 2952–2961.

[28] J.L. Bentley, Multidimensional Binary Search Trees Used for Associative Searching, Communications of the ACM. 18 (1975) 509–517. doi:10.1145/361002.361007.

[29] I. Volnyansky, V. Pestov, Curse of Dimensionality in Pivot Based Indexes, in: Second International Workshop on Similarity Search and Applications, 2009: pp. 39–46. doi:10.1109/SISAP.2009.9.

[30] L. Boytsov, B. Naidan, Engineering Efficient and Effective Non-metric Space Library, in: N. Brisaboa, O. Pedreira, P. Zezula (Eds.), Similarity Search and Applications, Springer Berlin Heidelberg, 2013: pp. 280–293.

[31] P. Dollár, R. Appel, S. Belongie, P. Perona, Fast feature pyramids for object detection, IEEE Transactions on Pattern Analysis and Machine Intelligence. 36 (2014) 1532–1545.

[32] A.V. Savchenko, Fast multi-class recognition of piecewise regular objects based on sequential three-way decisions and granular computing, Knowledge-Based Systems. 91 (2016) 252–262. doi:10.1016/j.knosys.2015.09.021.

[33] E.R. Vidal, An algorithm for finding nearest neighbours in (approximately) constant average time, Pattern Recognition Letters. 4 (1986) 145–157. doi:10.1016/0167-8655(86)90013-9.

[34] M.L. Micó, J. Oncina, E. Vidal, A New Version of the Nearest-neighbour Approximating and Eliminating Search Algorithm (AESA) with Linear Preprocessing Time and Memory Requirements, Pattern Recognition Letters. 15 (1994) 9–17. doi:10.1016/0167-8655(94)90095-7.

[35] J.S. Beis, D.G. Lowe, Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces, in: Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, 1997: pp. 1000–1006.

[36] P. Zezula, P. Savino, G. Amato, F. Rabitti, Approximate Similarity Retrieval with M-trees, The VLDB Journal. 7 (1998) 275–293. doi:10.1007/s007780050069.

[37] P. Indyk, R. Motwani, Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality, in: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, ACM, New York, NY, USA, 1998: pp. 604–613. doi:10.1145/276698.276876.

[38] D. Novak, M. Kyselak, P. Zezula, On Locality-sensitive Indexing in Generic Metric Spaces, in: Proceedings of the Third International Conference on SImilarity Search and APplications, ACM, New York, NY, USA, 2010: pp. 59–66. doi:10.1145/1862344.1862354.

[39] M. Muja, D.G. Lowe, Fast approximate nearest neighbors with automatic algorithm configuration, in: VISAPP International Conference on Computer Vision Theory and Applications, 2009: pp. 331–340.

[40] E.C. Gonzalez, K. Figueroa, G. Navarro, Effective Proximity Retrieval by Ordering Permutations, IEEE Transactions on Pattern Analysis and Machine Intelligence. 30 (2008) 1647–1658. doi:10.1109/TPAMI.2007.70815.

[41] E. Chávez, G. Navarro, Probabilistic proximity search: Fighting the curse of dimensionality in metric spaces, Information Processing Letters. 85 (2003) 39–46. doi:10.1016/S0020-0190(02)00344-7.

[42] L. Rutkowski, Adaptive probabilistic neural networks for pattern classification in time-varying environment, IEEE Transactions on Neural Networks. 15 (2004) 811–827. doi:10.1109/TNN.2004.828757.

[43] Q.D. Katatbeh, J. Martínez-Aroza, J.F. Gómez-Lopera, D. Blanco-Navarro, An Optimal Segmentation Method Using Jensen–Shannon Divergence via a Multi-Size Sliding Window Technique, Entropy. 17 (2015) 7996–8006.

[44] P.J. Curran, K.A. Bollen, P. Paxton, J. Kirby, F. Chen, The noncentral chi-square distribution in misspecified structural equation models: Finite sample results from a Monte Carlo simulation, Multivariate Behavioral Research. 37 (2002) 1–36.

[45] S.Y. Chun, A. Shapiro, Normal versus noncentral chi-square asymptotics of misspecified models, Multivariate Behavioral Research. 44 (2009) 803–827.

[46] J.K. Patel, C.B. Read, Handbook of the Normal Distribution, 2 edition, CRC Press, New York, 1996.

[47] E. P’kalska, R.P. Duin, Classifiers for dissimilarity-based pattern recognition, in: Proceedings of the 15th

International Conference on Pattern Recognition, IEEE Computer Society, 2000: pp. 12–16.

[48] G. Burghouts, A. Smeulders, J.-M. Geusebroek, The distribution family of similarity distances, in: Advances in Neural Information Processing Systems, 2008: pp. 201–208. http://papers.nips.cc/paper/3367-the-distribution-family-of-similarity-distances (accessed January 25, 2016).

[49] T.F. Gonzalez, Clustering to minimize the maximum intercluster distance, Theoretical Computer Science. 38 (1985) 293–306. doi:10.1016/0304-3975(85)90224-5.

[50] C. Silpa-Anan, R. Hartley, Optimised KD-trees for fast image descriptor matching, in: Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, 2008: pp. 1–8. doi:10.1109/CVPR.2008.4587638.

[51] M. De Marsico, M. Nappi, Face Recognition in Adverse Conditions: A Look at Achieved Advancements, Face Recognition in Adverse Conditions. (2014) 388.

[52] A.V. Savchenko, Clustering and maximum likelihood search for efficient statistical classification with medium-sized databases, Optimization Letters. (2015). doi:10.1007/s11590-015-0948-6.

[53] G.B. Huang, M. Mattar, T. Berg, E. Learned-Miller, Labeled faces in the wild: A database forstudying face recognition in unconstrained environments, in: Workshop on Faces in’Real-Life’Images: Detection, Alignment, and Recognition, 2008.

[54] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, et al., Caffe: Convolutional architecture for fast feature embedding, in: Proceedings of the ACM International Conference on Multimedia, ACM, 2014: pp. 675–678.

[55] L. Best-Rowden, H. Han, C. Otto, B.F. Klare, A.K. Jain, Unconstrained face recognition: Identifying a person of interest from a media collection, IEEE Transactions on Information Forensics and Security. 9 (2014) 2144–2157.

[56] A.V. Savchenko, L.V. Savchenko, Towards the creation of reliable voice control system based on a fuzzy approach, Pattern Recognition Letters. 65 (2015) 145–151. doi:10.1016/j.patrec.2015.07.013.

[57] T.M. Bailey, U. Hahn, Phoneme similarity and confusability, Journal of Memory and Language. 52 (2005) 339–362.

**Andrey Savchenko** received PhD in Mathematical Modeling (2010) in Higher School of Economics, Moscow, Russian Federation, and M.S. degree in Applied Mathematics and Informatics (2008) in Nizhniy Novgorod State Technical University, Russian Federation. Currently, he is working in National Research University Higher School of Economics, Nizhny Novgorod.

Table 1. Pivot-based Maximum Likelihood Approximate Nearest Neighbor algorithm

| **Data**: an observed image *X*, reference database $\{X_r\}$ |
|---|

**Output**: reference image $X^*$, which is either the nearest neighbor of $X$, or $\rho\left(X, X^*\right) < \rho_0$, or the number of distance computations exceeds $E_{\max}$

**Preliminarily step**: the distance matrix $\mathrm{P} = [\rho_{i,j}]$ and threshold $\rho_0$ are computed

```
1. Pivots $\left\{X_{r_1},...,X_{r_p}\right\}$ are chosen (10)
2. Check condition (3) for all pivots
3.Assign $\mu := r_1$, $k := 1$
4.Calculate the distances to all pivots - for each $i \in \{2,...,p\}$ repeat
    4.1. If $\rho(X, X_{r_i}) < \rho(X, X_\mu)$, then
        4.1.1. Assign $\mu := r_i$.
    4.2. Assign $k := k+1$
5. For each $\mu \in \{1,...,R\} - \{r_1,...,r_p\}$ repeat
    5.1. For each $i \in \{1,...,p\}$ repeat
        5.1.1. Initialize an array of accumulated log-likelihoods (11): assign $\varphi_{\Sigma;\mu} := \varphi_{\Sigma;\mu} + \varphi_\mu(r_i)$
6.Sort the array $\{\varphi_{\Sigma;\mu}\}$ in ascending order. The reference images corresponding to the ordered log-likelihoods are obtained $\{X_{r_i}\}, i \in \{p+1,...,R\}$
7.While $\rho(X, X_\mu) \geq \rho_0$ AND $k < E_{\max}$, repeat
    7.1. If $\rho(X, X_{r_l}) < \rho(X, X_\mu)$, then
        7.1.1. Assign $\mu := r_k$.
    7.2. Assign $k := k+1$
8.Return $X_\mu$ as a result $X^*$
```

Table 2. Time and space complexities of the P-ML-ANN algorithm

| | **Time complexity** | **Space complexity** |
|---|---|---|
| NN method (exhaustive search) | $O(R \cdot N)$ | $O(R \cdot N)$ |
| Ordering permutations | $O(E_{\max} \cdot (N+p) + p \log p)$ | $O(R \cdot (N+p))$ |
| DEM | $O(E_{\max} \cdot (N + M \log(E_{\max} \cdot M))$ | $O(R \cdot (N+R))$ |

| | | |
|---|---|---|
| ML-ANN | $O\left(E_{\max}\cdot(N+R)\right)$ | $O\left(R\cdot(N+R)\right)$ |
| P-ML-ANN | $O\left(E_{\max}\cdot N+p\log p\right)$ | $O\left(R\cdot(N+p)\right)$ |

Table 3. Face recognition results, FERET dataset

| **Classifier** | **Δ=0** | | **Δ=1** | |
|---|---|---|---|---|
| | **Error rate, %** | **Classification time, ms.** | **Error rate, %** | **Classification time, ms.** |
| SVM | 9.13±1.39 | 14.75±1.05 | | - |
| NN method (18), Euclidean metric | 5.17±1.20 | 1.92±0.35 | 3.25±0.35 | 12.82±0.77 |
| NN method (18), HT-PNN (3), (4) | 3.69±1.15 | 8.59±0.52 | 1.48±0.23 | 59.90±1.38 |

Table 4. Face recognition results, Essex dataset

| **Dataset** | **Classifier** | **Δ=0** | | **Δ=1** | |
|---|---|---|---|---|---|
| | | **Error rate, %** | **Classification time, ms.** | **Error rate, %** | **Classification time, ms.** |
| Essex | SVM | 0.16±0.06 | 5.06±0.89 | | - |
| | NN method (18), Euclidean metric | 0.16±0.06 | 7.77±0.92 | 0.16±0.05 | 39.94±1.04 |
| | NN method (18), HT-PNN (3), (4) | 0.16±0.06 | 31.45±1.15 | 0.16±0.05 | 242.95±1.56 |
| Essex (clustered) | SVM | 0.82±0.11 | 2.61±0.11 | | - |
| | NN method (18), Euclidean metric | 0.57±0.10 | 1.22±0.13 | 0.25±0.08 | 10.86±0.48 |
| | NN method (18), HT-PNN (3), (4) | 0.49±0.10 | 5.19±0.27 | 0.24±0.09 | 37.08±0.67 |

Highlights

- Maximal-likelihood search improves the performance of the nearest neighbor method
- Proposed method is 2-10 faster than brute force, randomized kd-tree and perm-sort
- Unlike the baseline DEM, proposed method does not require quadratic memory space
- The method is applied with similarity measures which do not met metric properties

Figure1

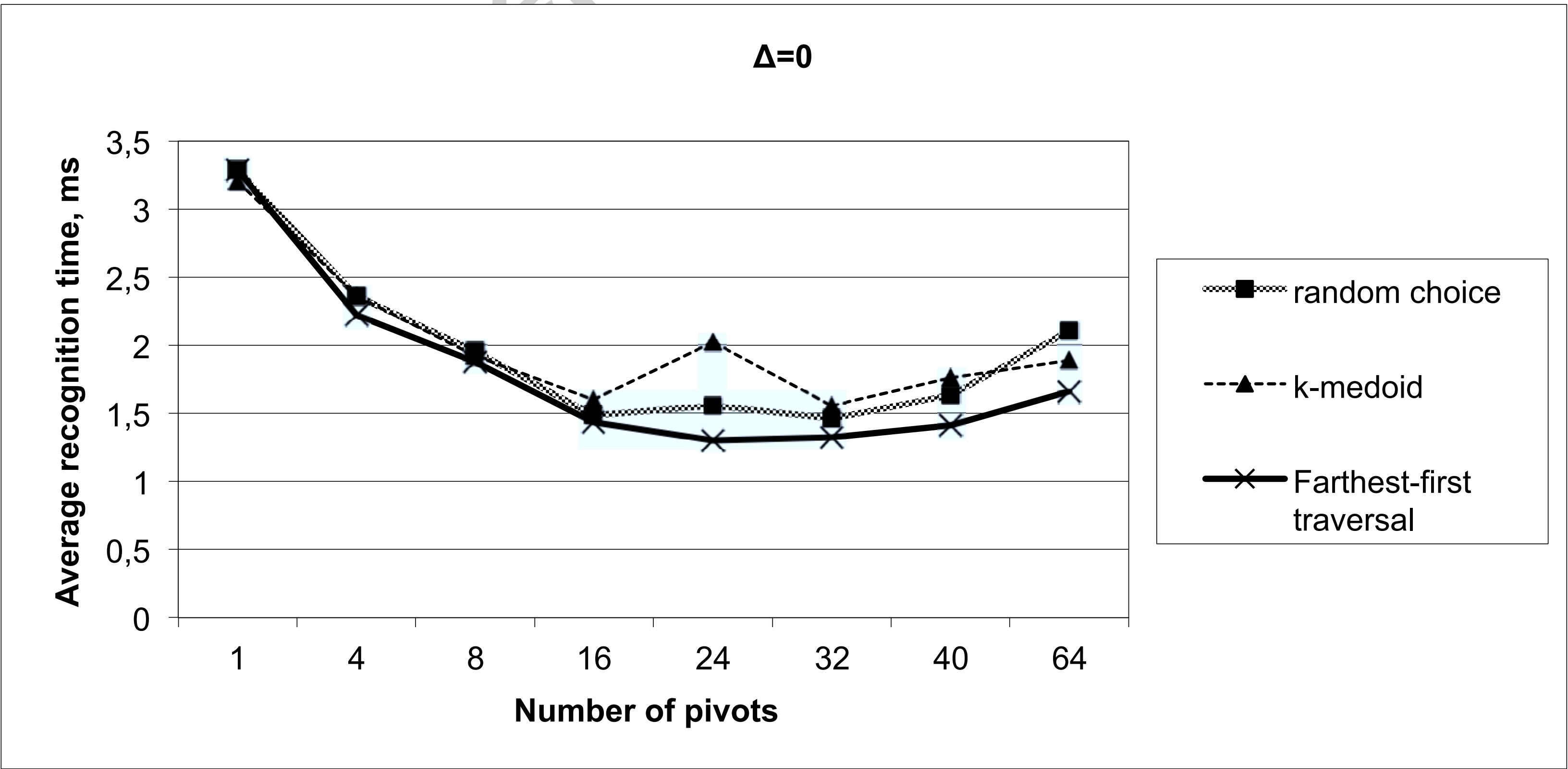

Δ=0
Average recognition time, ms
3,5
3
2,5
2
1,5
1
0,5
0
1
4
8
16
24
32
40
64
Number of pivots
random choice
k-medoid
Farthest-first traversal

Figure2

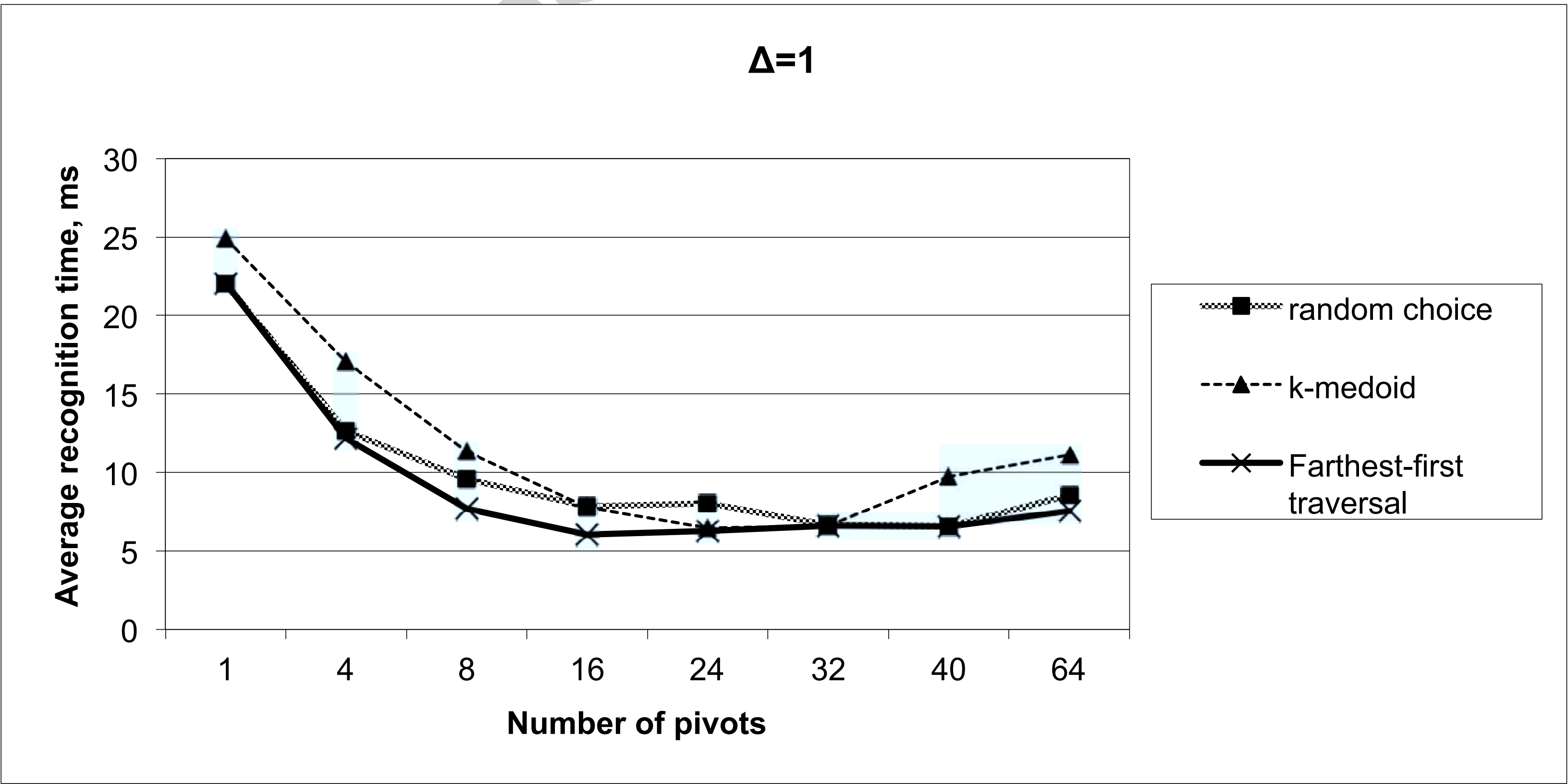
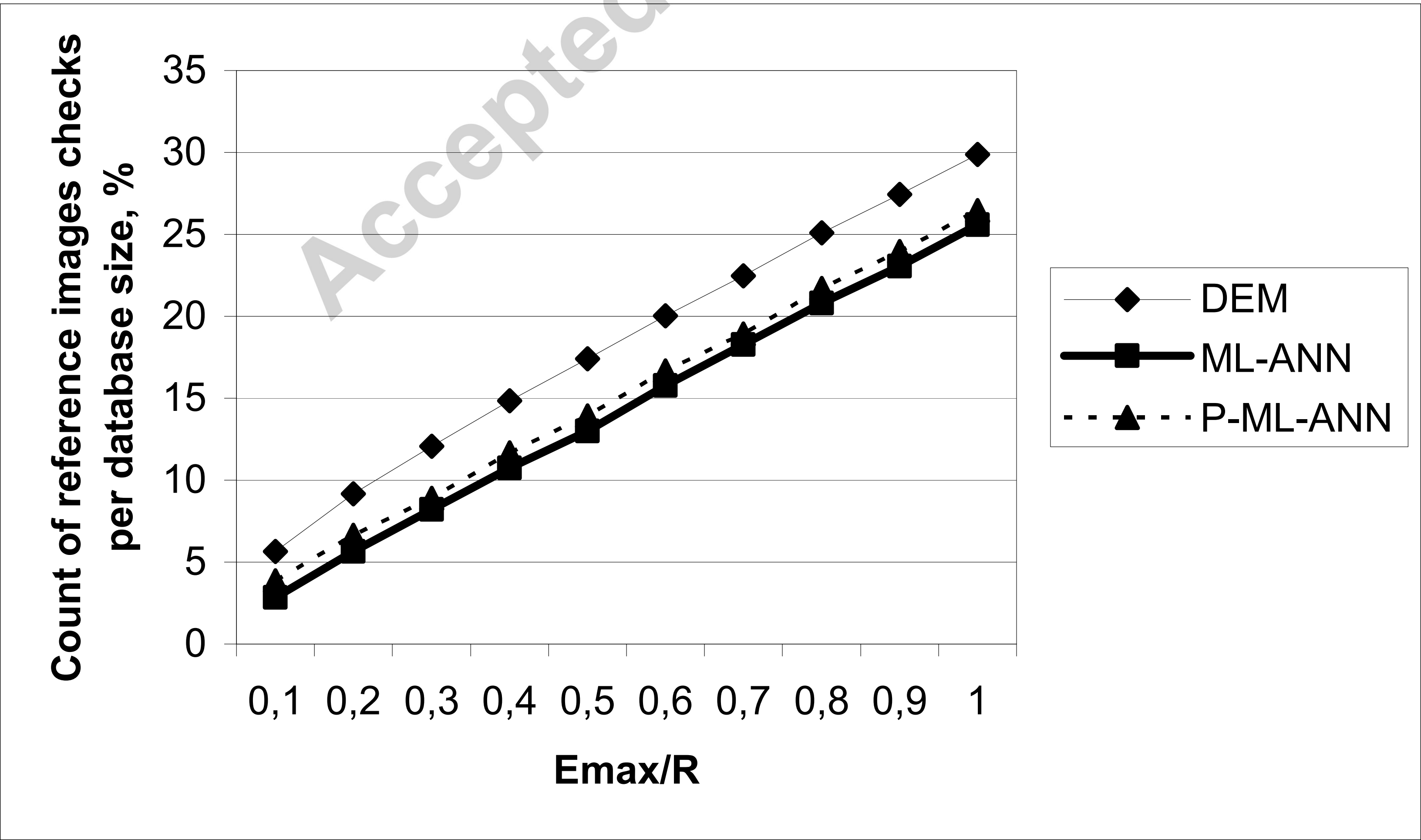

Δ=1
Average recognition time, ms
30
25
20
15
10
5
0
1
4
8
16
24
32
40
64
Number of pivots
random choice
k-medoid
Farthest-first traversal

Figure3


Count of reference images checks per database size, %
35
30
25
20
15
10
5
0
0,1
0,2
0,3
0,4
0,5
0,6
0,7
0,8
0,9
1
Emax/R
DEM
ML-ANN
P-ML-ANN

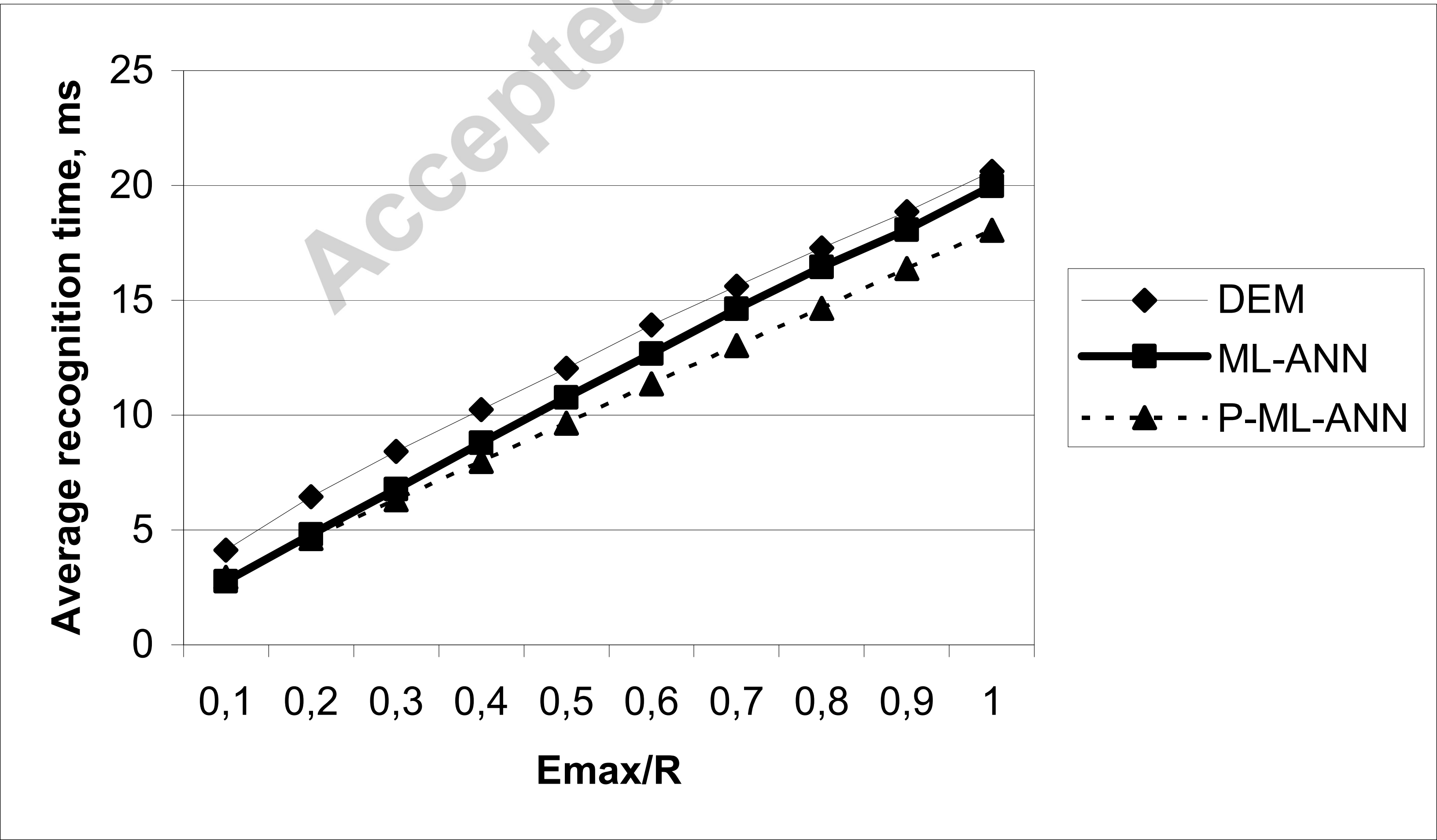

25
20
15
10
5
0
Average recognition time, ms
0,1
0,2
0,3
0,4
0,5
0,6
0,7
0,8
0,9
1
Emax/R
DEM
ML-ANN
P-ML-ANN

Figure5

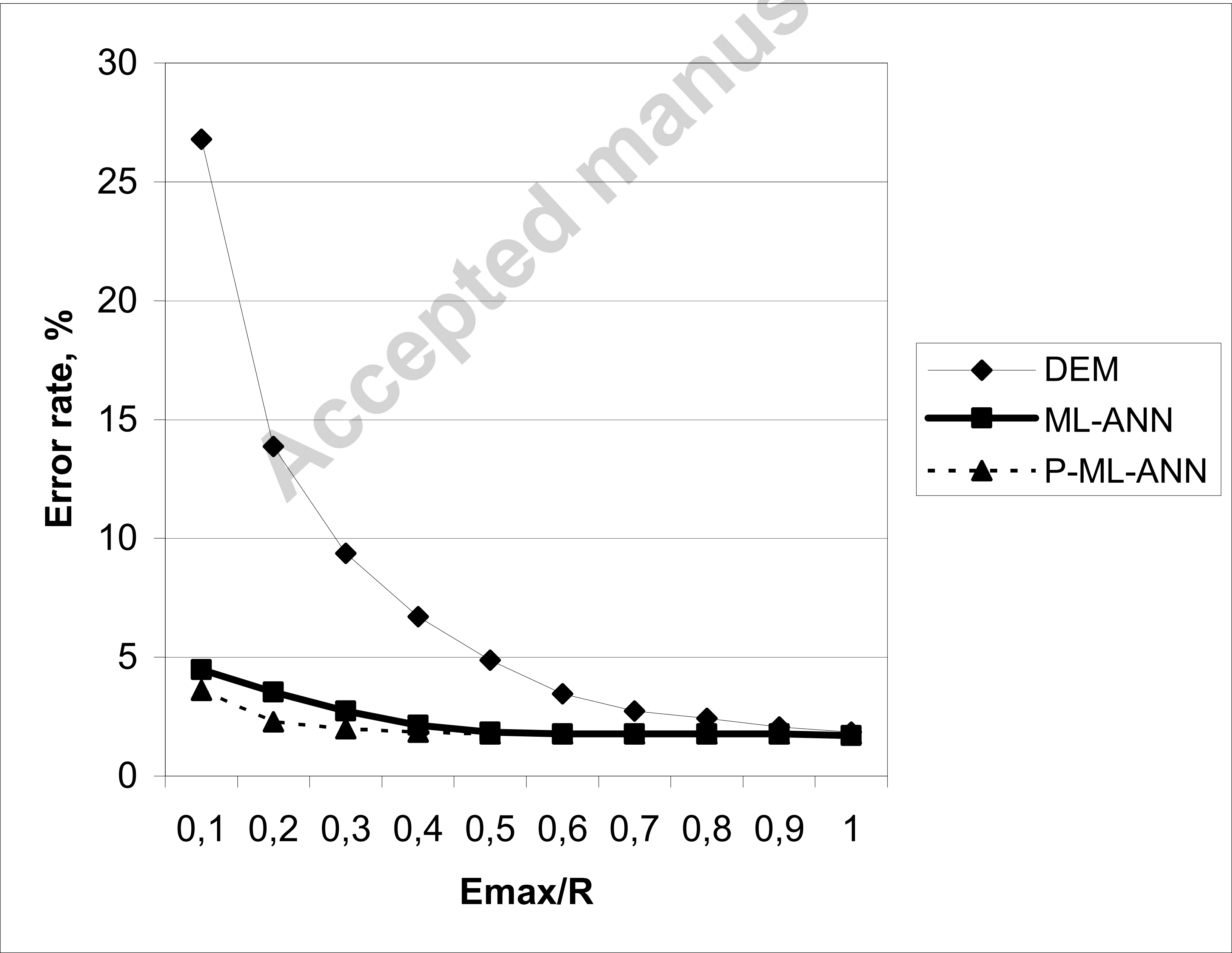

30
25
20
15
10
5
0
Error rate, %
0,1
0,2
0,3
0,4
0,5
0,6
0,7
0,8
0,9
1
Emax/R
DEM
ML-ANN
P-ML-ANN

Figure6

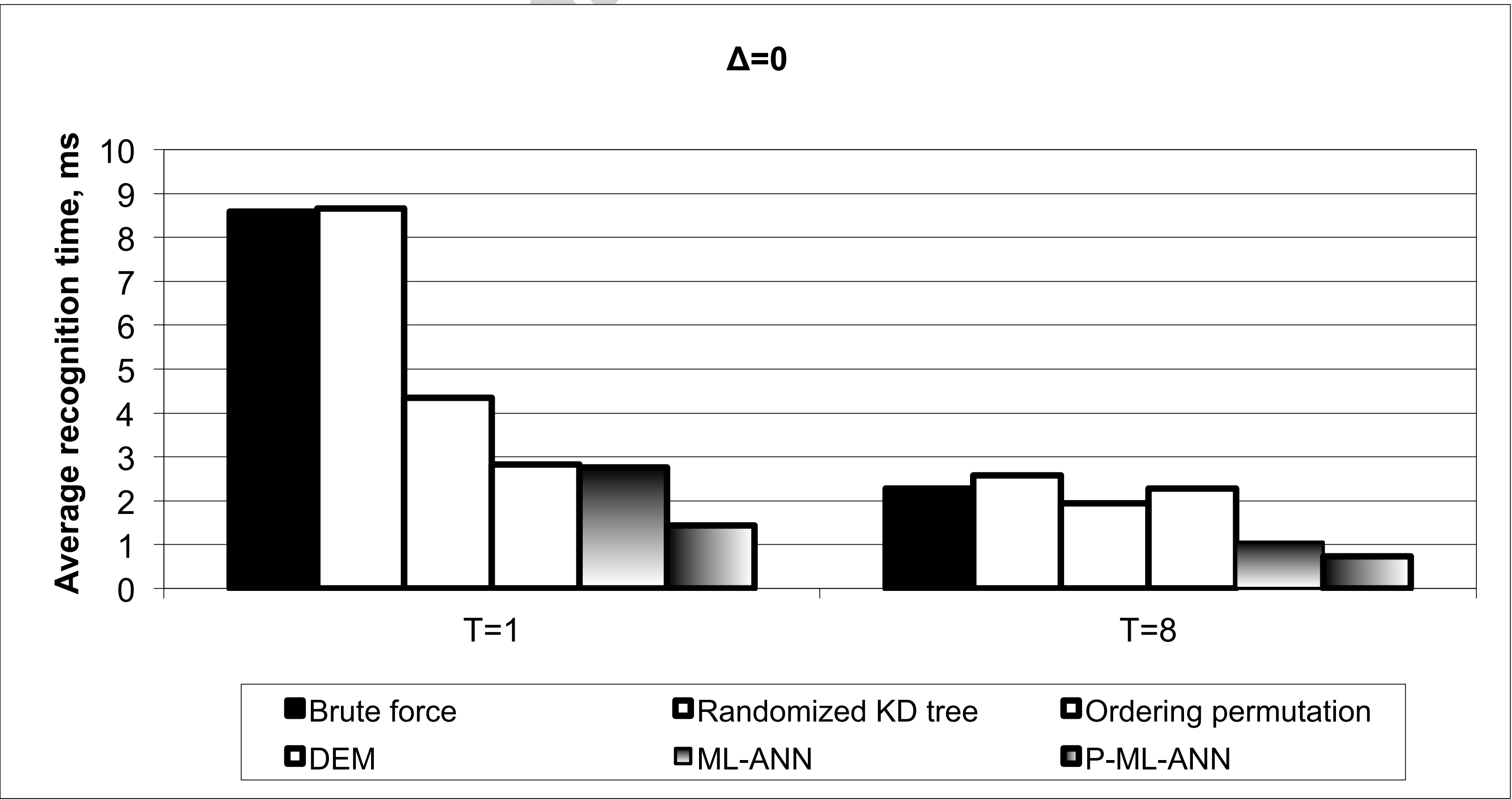

Δ=0
Average recognition time, ms
0
1
2
3
4
5
6
7
8
9
10
T=1
T=8
Brute force
Randomized KD tree
Ordering permutation
DEM
ML-ANN
P-ML-ANN

Figure7

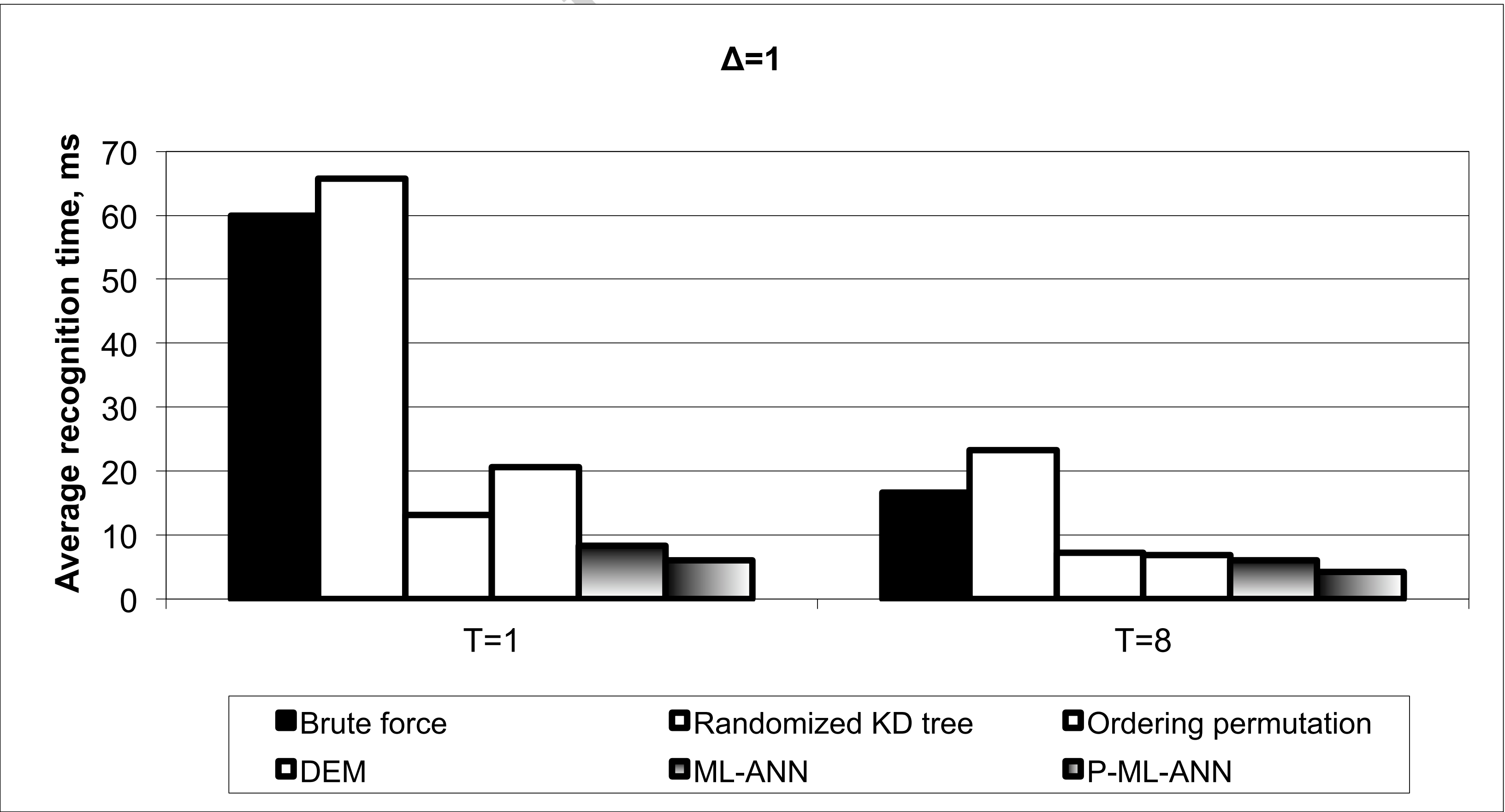

Δ=1
Average recognition time, ms
70
60
50
40
30
20
10
0
T=1
T=8
Brute force
Randomized KD tree
Ordering permutation
DEM
ML-ANN
P-ML-ANN

Figure8

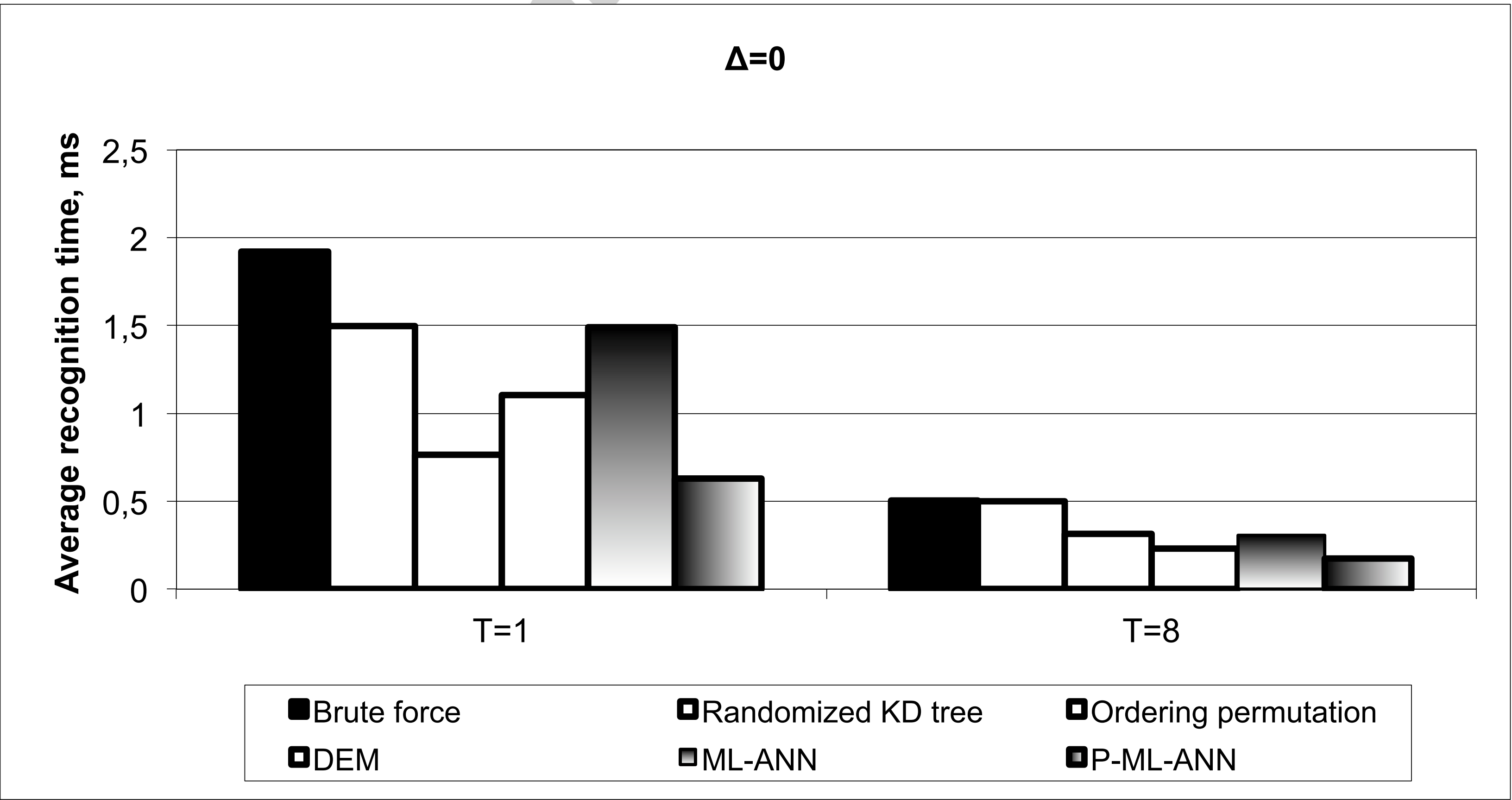

Δ=0
Average recognition time, ms
2,5
2
1,5
1
0,5
0
T=1
T=8
Brute force
Randomized KD tree
Ordering permutation
DEM
ML-ANN
P-ML-ANN

Figure9

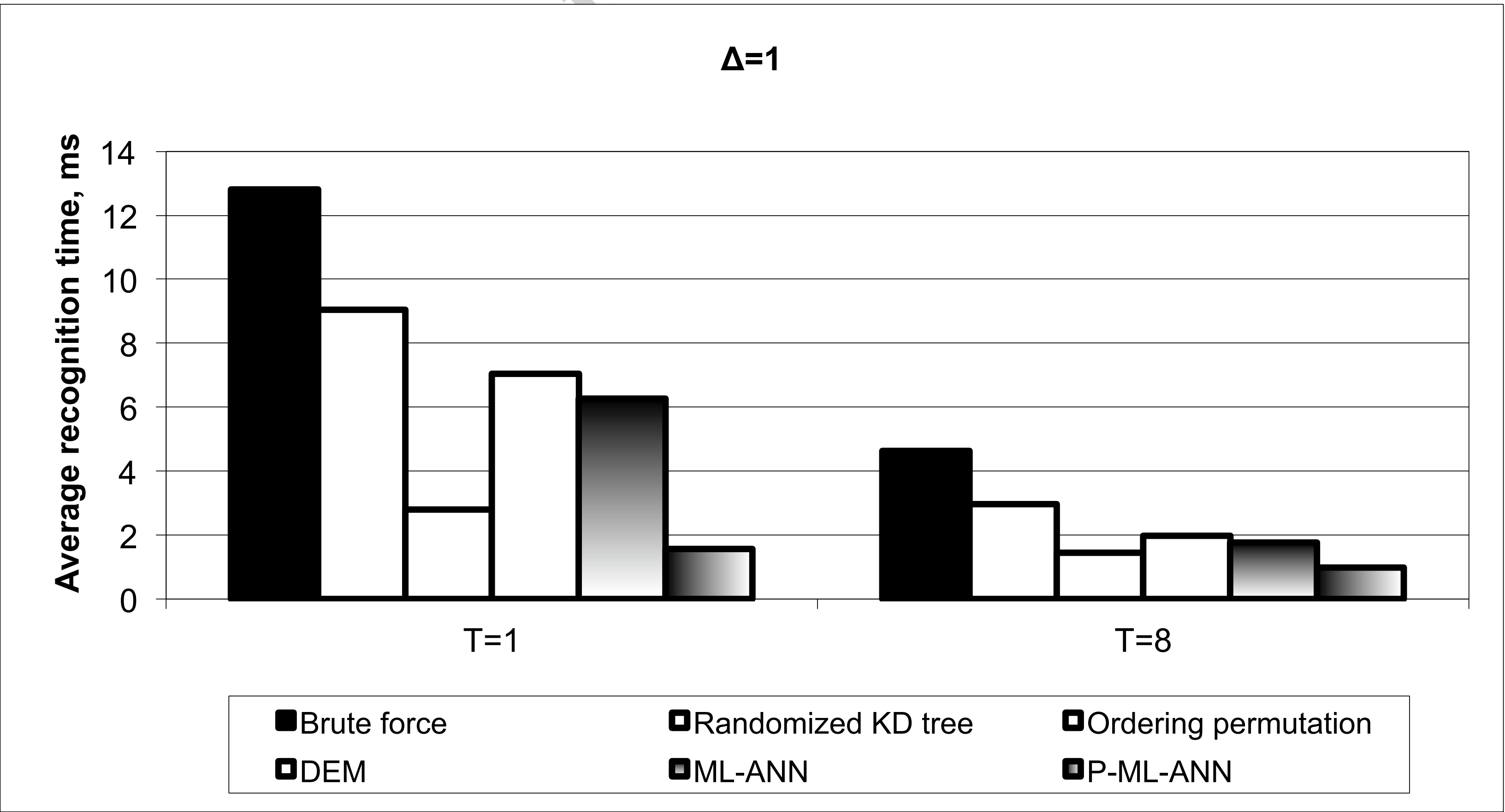
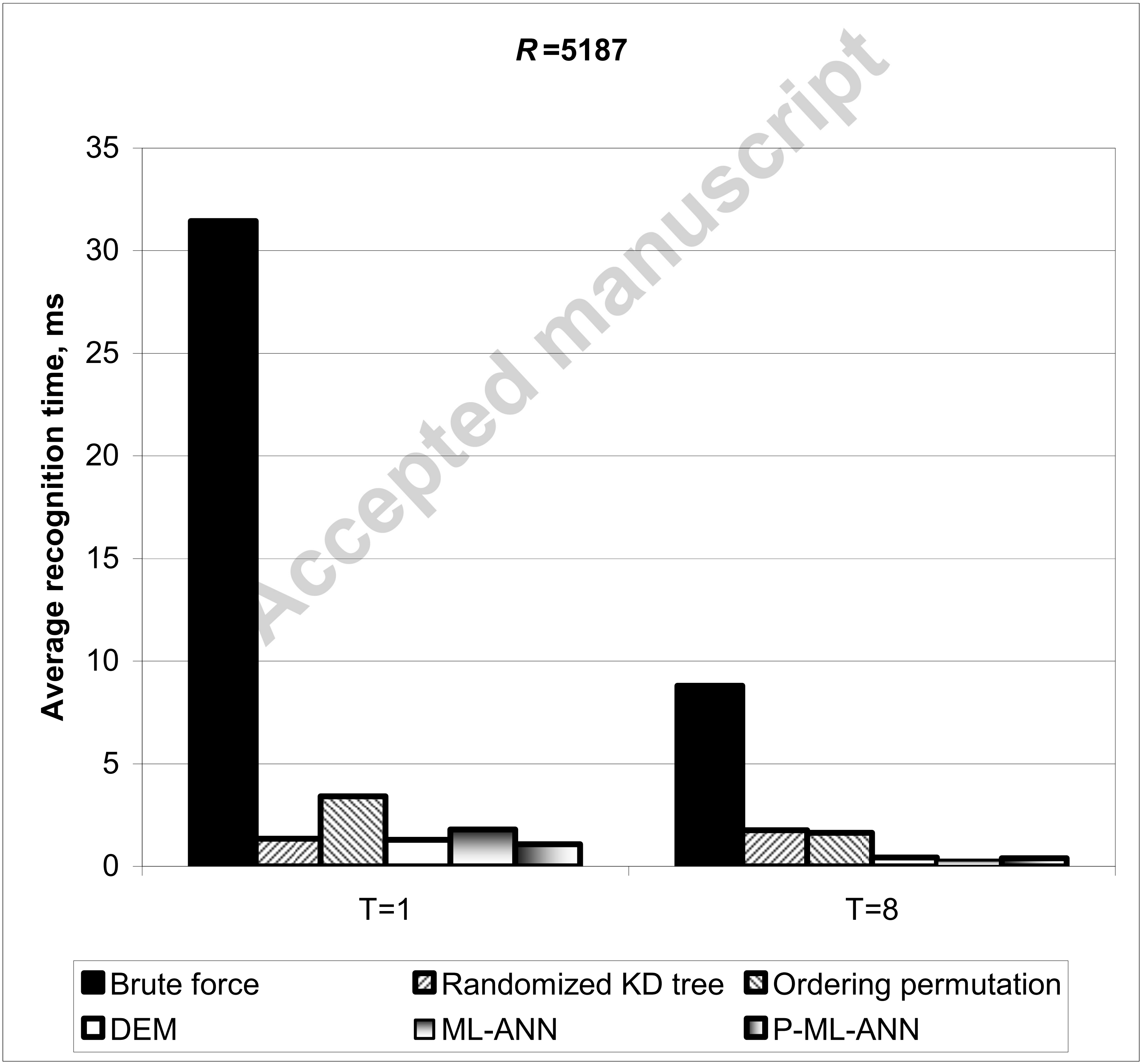

Δ=1
Average recognition time, ms
0
2
4
6
8
10
12
14
T=1
T=8
Brute force
Randomized KD tree
Ordering permutation
DEM
ML-ANN
P-ML-ANN

Figure10


*R* =5187
Average recognition time, ms
35
30
25
20
15
10
5
0
T=1
T=8
Brute force
Randomized KD tree
Ordering permutation
DEM
ML-ANN
P-ML-ANN

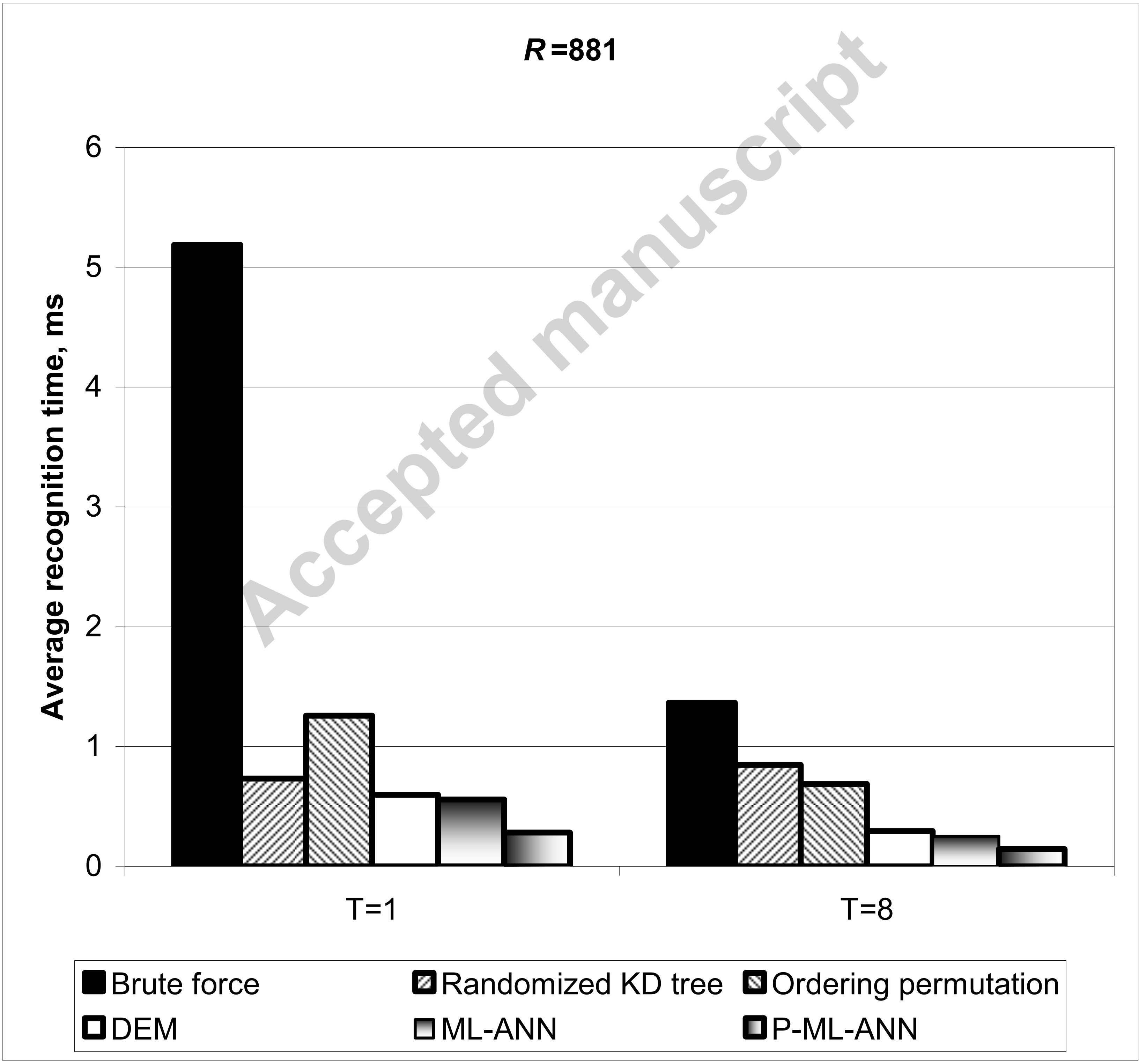
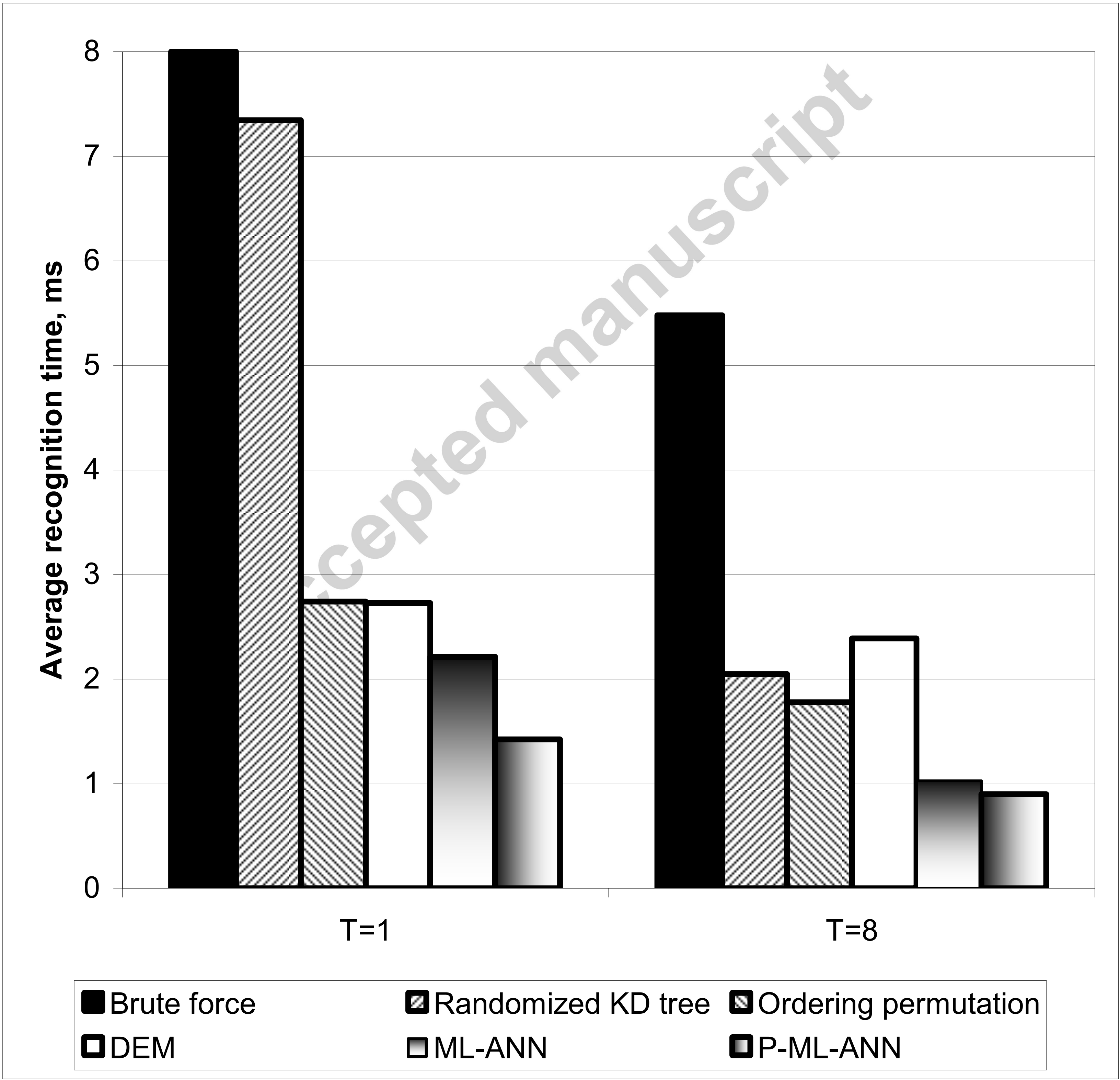

*R* =881
Average recognition time, ms
6
5
4
3
2
1
0
T=1
T=8
Brute force
Randomized KD tree
Ordering permutation
DEM
ML-ANN
P-ML-ANN

Figure12


Average recognition time, ms
0
1
2
3
4
5
6
7
8
T=1
T=8
Brute force
Randomized KD tree
Ordering permutation
DEM
ML-ANN
P-ML-ANN