On Some Fuzzy Classification Algorithms and the AEC Model

ANDREY OSIPOV
Federal State Institution "Scientific-Research Institute for System Analysis
of the Russian Academy of Sciences"
Nakhimovskii pr. 36-1, Moscow, 117218
RUSSIA

Abstract: - In the paper some fuzzy classification algorithms based upon a nearest neighbor decision rule are considered in terms of the pattern recognition algorithms which are based on the computation of estimates (the so-called AEC model). It is shown that the fuzzy K nearest neighbor algorithm can be assigned to the AEC class. In turn, it is found that some standard AEC algorithms, which depend on a number of numerical parameters, can be used as fuzzy classification algorithms. Yet among them there exist algorithms extremal with respect to these parameters. Such algorithms provide maximum values of the associated performance measures.

Key-Words: - Pattern recognition, performance measures, heuristic algorithms, fuzzy set theory

Received: May 13, 2020. Revised: July 2, 2020. Re-revised: July 24, 2020. Accepted: July 31, 2020. Published: August 23, 2020.

1 Introduction

In recent decades, a growing number of pattern recognition methods, which rely on the fuzzy set theory, have emerged [1]. For solving some computer vision tasks, e. g. for the processing of blurred images, the use of such methods is well justified. Also, in many pattern recognition methods the reference classification is ambiguous, so the fuzzy approach can be applied here as well.

The appearance of fuzzy algorithms has raised some natural tasks, e. g. to study their scope, to find an optimal fuzzy algorithm for solving the same problem, or to compare the fuzzy methods with their non-fuzzy counterparts. These tasks are partly reflected in our software system named PICASSO, designed for the purpose of empirical supervised evaluation of computer vision algorithms [2]-[4], [10]. The system performs a comparison between a processed image (algorithm's output) against a reference image which is often referred to as a ground-truth, by using some quantitative evaluation criteria – similarity measures. The current version of PICASSO, alongside with some amount of fuzzy ground truth images (as well as the methods for their generation) contains some fuzzy similarity measures, applicable to both fuzzy and non-fuzzy algorithms (see [3]-[4] for details).

In general, our approach to evaluation of the fuzzy algorithms corresponds to the ideas expressed by Yu. M. Zhuravlev in 1970s about the study of pattern recognition algorithms [5]-[7]. For example, as noted in [5], "A great number of various methods and algorithms emerged at the early stage in the evolution of recognition theory and practice and were applied to practical problems without any serious mathematical basis. As is customary in all experimental sciences, the methods were verified by a direct test – success or failure in tackling real problems. Many of them have stood this test and are used despite the lack of mathematical justification. … So, having recognized the existence and practical usefulness of ill-defined procedures of solving poorly formalized problems as reality, we face the task of studying the very set of these procedures using rigorous mathematical methods." It may be established as a fact, that to date the share of practically efficient, though theoretically unfounded algorithms have risen substantially. For instance, nowadays many pattern recognition methods rely on the use of neural network models. Yet, all of these methods inherit the open issues of the neural network theory: completeness of the training samples, avoidance of deadlocks during the network training, etc. Thus, the problem of studying these "non-rigorous algorithms using rigorous mathematical methods" remains to be actual.

As part of a study of heuristic pattern recognition algorithms, the model of the recognition algorithms based on estimate calculations (later referred to as AEC[1] model) was introduced in [6]. This model unified several known by that time principles and procedures of recognition. To date, this model and its generalizations were considered in many papers

[1] ABO in Russian

(see [8] and references thereafter). In particular, it found its place in the Descriptive Image Analysis [9]. Below we describe the main principles of the classical version of this model.

The purpose of this paper is to study connections between the fuzzy classification algorithms, where the grade of membership of an object in some class depends on the number of neighboring training sample objects from the same class, with the AEC model. Such study may be useful for the further development of the fuzzy recognition algorithms and for extending the "fuzzy" component of the theory of heuristic algorithms.

The paper is organized as follows. Section 1 mainly contains a description of a known fuzzy *K*-nearest neighbor algorithm. Brief information on the classical recognition algorithms based on estimate calculations is contained in the Section 2. The sections to follow deal with comparison of the nearest neighbor algorithms with the AEC model, numerical experiments and discussion of results.

2 Fuzzy Nearest Neighbor Algorithms

First we recall some basic notions of the fuzzy sets theory. Let X be a non-empty set. A fuzzy set C in X is a pair $<X, f_C>$, where f_C is a mapping from X into [0, 1]. The value $f_C(x)$ at $x \in X$ is called the grade of membership of x to C, and the function f_C is called the fuzzy set membership function. A fuzzy set is nonempty if for at least one $x \in X$, $f_C(x)>0$. Note that ordinary (crisp) subsets M of X are covered by the fuzzy set approach if we view them as standard characteristic functions $1_M : X \to [0,1]$. It means that, if we have an object $x \in M$, then $f_M(x)=1$ and $f_M(x)=0$ for $x \notin M$.

Definition 1. A fuzzy classification F of X:

$$F := < X, f_{C_1},....,f_{C_N} >, \quad f_{C_m} : X \to [0,1], \quad m = 1,...,N;$$

is called the collection of N fuzzy classes $C_m := < X, f_{C_m} >$, satisfying the condition

$$\sum_{m=1}^{N} f_{C_m}(x) = 1, \quad \forall x \in X, \tag{1}$$

where f_{C_m} is a membership function of the class C_m. One can easily see that an ordinary classification of X (its representation as a union of N disjoint subsets) is also its fuzzy classification (every element of X belongs only to one class, and the membership function is the characteristic function of this class). From now on assume that X is finite.

We now turn to a known *K*- fuzzy nearest neighbour algorithm (Google search gives over 1900 matches for references to the original paper [10] where it was introduced; we mention here its recent applications to big data analysis [11]-[12]). The essence of the algorithm is as follows. Suppose we have a set X containing objects from l classes. Each object is identified with its feature vector of fixed length. Let $M_R \subset X$ be the training set containing the objects for which the class membership is known in advance, such that all l classes are represented. On the first stage of the algorithm, the objects of M_R are enumerated, and for each object $x \in M_R$ its membership in each class is assigned according to the equation:

$$\mu_{ij}(x) = \begin{cases} 0.51 + 0.49\left(\dfrac{n_{ij}}{K_1}\right) & \text{for } x \text{ from the class } j, \\ 0.49\left(\dfrac{n_{ij}}{K_1}\right) & \text{otherwise;} \end{cases}$$

$$j = 1,...,l; \tag{2}$$

where i is the number of x in M_R, j is the class number, K_1 is a certain fixed number of the nearest neighbors (the neighborship function is defined according to a certain (e. g. Euclidian) metric applied to the feature vectors of the objects. As we see, the more neighbors of x are of the same class, the higher is its grade of membership in this class (it equals to 1 for K_1 nearest neighbors).

Then, suppose we have a test set $M_K \subset X$. For each $y \in M_K$, we find its K nearest neighbors (K is not necessarily equal to K_1) and assign its membership in each of the l classes by the formula:

$$u_j(y) = \frac{\sum_{k=1}^{K} \mu_{jk}(x_k) \left(\dfrac{1}{\| y - x_k \|^{\frac{2}{P-1}}} \right)}{\sum_{k=1}^{K} \left(\dfrac{1}{\| y - x_k \|^{\frac{2}{P-1}}} \right)}, \quad j = 1,...,l; \tag{3}$$

where P is a fixed natural number (e. g. P=2) and $\|.\|$ is the norm induced by the above-mentioned metric. It is assumed that y belongs to the class with the highest membership value. As we see, this algorithm depends on the parameters K, K_1 and P.

The latter parameter determines how heavily the distance is weighted when calculating each neighbor's contribution to the membership value. As P approaches one, the closer neighbors are weighted far more heavily than those farther away; as P increases, the neighboring points are more evenly weighted. We denote this algorithm as F_K.

In spite of a fairly good results obtained in some cases (we will dwell on this below), in [10] were also mentioned some shortages of F_K. For example, assume that an object y is assigned 0.53 membership in class one, 0.45 membership in class two and 0.02 membership in class three. According to F_K, y belongs to class one. However, from the common sense viewpoint, we should be hesitant to assign the object y based on these results. We can only feel confident that it does not belong to class three. It exhibits a high degree of membership in both classes one and two, so a further examination of y is highly desirable.

In particular applications, e. g. in the face recognition, usually only the first stage of this algorithm (the calculation of μ_{ij} by formula (2)) is used. These quantities are used for the calculation of between-class scatter matrix (empirical covariance matrix) of the training set. It is believed (although not strictly proven) that this matrix is more balanced than if calculated without using them. Further on, various versions of the principle component analysis method are applied (see [4], [13], [14] for details).

In addition to the above-considered method, a fuzzy version of the nearest prototype algorithm was also considered in [10]. In this algorithm, first we select a set of l prototypes $(Z_1,...,Z_l)$, $Z_j \subset M_R$, representing all l classes. Then for each $y \in M_K$ we calculate its grade of membership in each class by the formula:

$$u_j(y) = \frac{\dfrac{1}{\| y - Z_k \|^{\frac{2}{P-1}}}}{\displaystyle\sum_{k=1}^{l} \left(\frac{1}{\| y - Z_k \|^{\frac{2}{P-1}}} \right)}, \qquad (4)$$

$$j = 1,...,l;$$

As we can see, the closer is the object y to the class prototype, the higher is its membership value in this class. The difference between (4) and (3) is that membership in each class is assigned based only on the distances from the prototypes of the classes. This is because the prototypes should naturally be assigned complete memberships in their own class.

Note that both of the above algorithms follow the same scheme: first, from the object's feature vector we calculate its membership values in each class. Then, based on these values, we decide to which class the object belongs. In other words, we apply a decision rule based on the highest membership value that the object belongs to a certain class.

3 On the Algorithms of Estimate Calculation

Considering the AEC algorithms, we first note that a principle of partial precedence underlines their working. The proximity between the parts of the feature sets of the objects already classified and the object presented for recognition is analyzed. The proximity is a partial precedent estimated numerically according to some predefined rule. A set of proximity measures yields a general quantitative estimate of an object to be recognized for a class (or, in other words, we get a membership value in this class).

Now consider the classical AEC model introduced in [5]. Denote an algorithm of this model as A. Assume X is a given finite set of objects which belong to l classes; we denote the latter as $K_1,\ldots,K_l$. Let $M_R \subset X$ and $M_K \subset X$ be the training (or reference) and the test sets respectively. Assume that the set M_R of size m contains $m1$ objects of class K_1, $m2$ objects of class $K_2,\ldots$, ml objects of class K_l, so all l classes are represented in the training set. Each element of X is identified with its vector feature set of length n. The aim is to classify each element of M_K as one of l classes.

For a fixed $k \subset \{1,\ldots,n\}$, denote as $\{\Omega_k\}$ the set of all subsets of $\{1,2,\ldots,n\}$ of cardinality k. The elements $\Omega_k \subset \{\Omega_k\}$, $\Omega_k = \{r1,_{\ldots,} rk\}$ are called the reference (or support) subsets of AEC model. Also, let $\rho(.,.)$ be a certain metric, and $\varepsilon_1,\ldots,\varepsilon_n$ − some positive numbers (threshold values).

On the initial stage of the algorithm A, a table containing representatives of all the classes is formed from the elements of M_R (their feature vectors). Namely, its first $m1$ rows contain the arbitrary enumerated objects $S_1,\ldots,S_{m1}$ from class K_1, then follow $m2$ rows containing the objects $S_{m1+1},\ldots,S_{m2}$ from class K_2, etc. It is denoted as $T_{n,m,l}$.

After that, for an arbitrary $\Omega_k \subset \{\Omega_k\}$, a reduced table $T_{n,m,l}(\Omega_k)$ is constructed from $T_{n,m,l}$ by keeping the columns number $r1,_{\ldots,} rk$ and removing the other ones. Denote its rows as $S_i(\Omega_k), i=1,\ldots m$. Similarly, for an object $S \subset M_K$ and its feature set $(\beta_1,\ldots,\beta_k)$, a reduced feature set $S(\Omega_k) = (\beta_{r1},\ldots,\beta_{rk})$ is built.

Then, for each i the distances in terms of the metric $\rho(.,.)$, between the coordinates of $S_i(\Omega_k)=(\alpha_{r1},\ldots,\alpha_{rk})$ and $S(\Omega_k)$ are calculated. Let fn

be the number of unfulfilled inequalities of the system $\rho(\alpha_{r1}, \beta_{r1}) \le \varepsilon_{r1}, \ldots, \rho(\alpha_{rk}, \beta_{rk}) \le \varepsilon_{rk}$. The objects S and S_i are called close in Ω_k, if for a fixed $\varepsilon \subset \{0,1,\ldots,k\}$, $fn \le \varepsilon$. Denote as $\{S_i\}^j$ the set of rows of $T_{n,m,l}$ close to S in Ω_k which belong to the class K_j, $j=1,\ldots,l$.

Further on, the quantities $\gamma(S_1), \ldots, \gamma(S_m)$ are introduced into consideration (typically they are identical for the objects of the same class). Set

$$\Gamma^j_{\Omega_k}(S) = \sum_{S_i \subset \{S_i\}^j} \gamma(S_i),$$

and

$$\Gamma_j(S) = \sum_{\Omega_k \subset \{\Omega_k\}} \Gamma^j_{\Omega_k}(S), \ j = 1,\ldots,l. \quad (5)$$

The values $\Gamma_j(S)$ are called the estimates of the classes K_j. They indicate the proximity of S to the elements of K_j (in other words, $\Gamma_j(S)$ is a vote in favor of K_j). Given $S=(\alpha_1,\ldots,\alpha_n)$ and $S_i=(\beta_1,\ldots,\beta_n)$, let $r(S,S_i)$ be the number of fulfilled inequalities of the system $\rho(\alpha_1, \beta_1) \le \varepsilon_1, \ldots, \rho(\alpha_n, \beta_n) \le \varepsilon_n$; (obviously, $0 \le r(S,S_i) \le n$). There exists a formula for calculation of $\Gamma_j(S)$, deductible from combinatorial arguments (see [7], Theorem 1):

$$\Gamma_j(S) = \sum_{i=m_{j-1}+1}^{m_j} \gamma_i \sum_{t=0}^{\varepsilon} C^{k-t}_{r(S,S_i)} C^{t}_{n-r(S,S_i)}, \quad (6)$$

where $m_0=0$, $\gamma_i = \gamma(S_i)$; $C^P_N = 0$ for $P>N$. As a result of above actions, the feature set $(\alpha_1,\ldots,\alpha_n)$ of S transforms into the set $(\Gamma_1(S), \Gamma_2(S),\ldots, \Gamma_l(S))$. Assume that the test set M_K contains q elements $S_1,\ldots,S_q$. Then, the above-considered operations can be interpreted as transformation of the matrix I of size $q \times l$ which rows are the feature sets of $S_1,\ldots,S_q$ respectively:

$$I = I(\mathrm{S}_1,\ldots,\mathrm{S}_q) = \begin{pmatrix} \alpha_{11} & \ldots & \alpha_{1n} \\ \vdots & \ldots & \vdots \\ \alpha_{q1} & \ldots & \alpha_{qn} \end{pmatrix},$$

into the matrix of estimates

$$\Gamma = \Gamma(\mathrm{S}_1,\ldots,\mathrm{S}_q) = \begin{pmatrix} \Gamma_{11} & \ldots & \Gamma_{1l} \\ \vdots & \ldots & \vdots \\ \Gamma_{q1} & \ldots & \Gamma_{qn} \end{pmatrix}, \quad (7)$$

where $\Gamma_{pj} = \Gamma_j(S_p)$, $p=1,\ldots,q$ of size $q \times l$. An operator R_A transforming I into Γ is called a recognition operator.

After the calculation of matrix Γ, the next step of the algorithm A is the classification itself, performed by using a fixed decision rule r_A. Here is its typical example (see [5]-[8] for more details): let Δ_1 and Δ_2 be given positive numbers, $0 < \Delta_2 \le 1$. Then the object S is assigned to the j-th class if

1) $\Gamma_j(S) - \Gamma_i(S) \ge \Delta_1, \quad j \ne i;$

2) $\dfrac{\Gamma_j(S)}{\sum_{i=1}^{l} \Gamma_i(S)} \ge \Delta_2, \quad j = 1\ldots l.$

If 1) and 2) are not fulfilled simultaneously for all j (in particular, when $\Gamma_j(S)=0$ for all j) then the algorithm refuses to classify S. It can be said that r_A is an operator which transforms the matrix Γ (6) into the classification matrix Cl_A with elements from the set $\{0,1,Na\}$ where 1 means that the object of the test set is assigned to the corresponding class, 0 – if not assigned, Na means refusal to classify the object. From the condition 1) follows that in the case of such r_A, this matrix contains at most one 1s in each row. Thus, each algorithm A of AEC model is a composition of a recognition operator and a decision rule:

$$A = r_A \circ R_A. \quad (8)$$

The decision rule defined by 1)-2) is relatively simple and corresponds to the classical version of this model. The combinatorial formula (6) for calculation of $\Gamma_j(S)$ is also typical for the classical model; in more complicated cases, e. g. for the multilevel object recognition performed by AEC algorithms, see [8].

As we see, the algorithms A have two characteristic features. First, for a fixed k, $1 \le k \le n$ all the feature subsets with size k are taken into consideration while calculating $\Gamma_j(S)$ by (5)-(6). Second, the importance of each class (and even of each object of the reference set) for the recognition is taken into account via the weights γ_i.

Also, in [5]-[7] the efficiency of recognition algorithms expressed in terms of some quality functionals (measures), was studied. Namely, assume that for each object of M_K it is known in advance, to which of the classes $K_1,K_2\ldots,K_l$ it belongs. Set $M_K \cap K_j = K_j'$, $j=1,\ldots,l$. Given algorithm A, each K_j' is matched to its partition K_{jp}', $p=0,\ldots,l$ into disjoint sets. For $p>0$, the set K_{jp}' consists of the objects of K_j assigned to the class K_p by the algorithm A, and K_{j0}' consists of non-classified objects. Then, a typical quality measure of A is defined as

$$\psi_A^0 = \frac{1}{\mu(M_K)} \sum_{j=1}^{l} \mu(K_{jj}'), \qquad (9)$$

where $\mu(.)$ is a certain measure on a set, e.g. its number of elements. In the latter case ψ_A^0 is called the overall sensitivity of A. This is a standard statistical performance measure of classification algorithms which evaluates the amount of correctly classified objects (see [2], [4], [15]). Given the weight coefficients $\kappa_1,\ldots,\kappa_l$; $0 \le \kappa_j \le 1$, characterizing the "importance" of each class, the above formula can be modified:

$$\psi_A^1 = \frac{1}{\mu(M_K)} \sum_{j=1}^{l} \kappa_j \mu(K_{jj}'). \qquad (10)$$

Also, assume that the quantities ν_{ij}, $0 \le \nu_{ij} \le 1$; $i,j=1,\ldots,l$ are given, such that for $i \ne j$, ν_{ij} is a penalty for assignment to the class K_i of object from the class K_j, and ν_{jj} is a reward for correct recognition of the latter. Let also φ_j, $0 \le \varphi_j \le 1$, be the penalties for refusal to classify an object from the class K_j. Then, the following quality functional of A was offered in [6]:

$$\psi_A^2 = max\{\tilde{\psi}_A^2, 0\}, \qquad (11)$$

where

$$\tilde{\psi}_A^2 = \frac{1}{\mu(M_K)} \sum_j \nu_{jj} \mu(K_{jj}') - \left(\sum_{\substack{i,j \\ i \ne j}} \nu_{ij} \mu(K_{ij}') + \sum_j \varphi_j \mu(K_{j0}') \right).$$

Let $\{A\}$ be the above-considered set of recognition algorithms. Each algorithm A can be encoded by the set of $n+m+4$ parameters k, $\varepsilon_1,\ldots,\varepsilon_n$, ε, $\gamma_1,\ldots,\gamma_m$, Δ_1, Δ_2 (as we have just seen, the algorithm is completely characterized by such set). In case of the decision rule r_A other than 1)-2), the latter two parameters can be replaced by the ones corresponding to this r_A. If the quantities $\gamma_i = \gamma(S_i)$ are the same for the objects S_i of the same class, then A can be associated with the smaller set k, $\varepsilon_1,\ldots,\varepsilon_n$, ε, $\gamma_1,\ldots,\gamma_l$, Δ_1, Δ_2 of $n+l+4$ parameters. Thus, the above measures (9)-(11) can be considered as multivariable functions of these parameters:

$$\psi_A^n = \psi_A^n(k, \varepsilon_1,\ldots,\varepsilon_n, \varepsilon, \gamma_1,\ldots,\gamma_m, \Delta_1, \Delta_2), \quad n = 0,1,2.$$

The domain T for the latter parameters can be naturally defined as follows: k, ε are nonnegative integers, $\varepsilon \le k$; $1 \le k \le n$; $0 \le \varepsilon_p \le \varepsilon^{max}$ $p=1,\ldots,n$; $0 \le \gamma_i \le 1$, $i=1,\ldots,m$; $0 < \Delta_1 \le \Delta_1^{max}$, $0 < \Delta_2 \le 1$.

From the condition 1) and (6) one can derive the following formula for Δ_1^{max} (see [6]):

$$\Delta_1^{max} = C_n^k \sum_{i=1}^{m} \gamma_i.$$

Each algorithm from $\{A\}$ assigns an object from the finite set M_K to one of l classes or rejects it as been unrecognizable. Thus, the measures (9)-(11), considered as functions on the set T, take a finite number of values. Therefore, the following statement holds ([6], Theorem 3):

Statement 1. *The functions (9)-(11) achieve an absolute maximum:* $max_T \psi_A^p$, $p = 0,1,2$ *in the set T.*

The algorithm, corresponding to the point of T where the extremum of ψ_A^p is achieved, is called the extremal algorithm. Thus, the problem of optimal algorithms design can be reduced to finding the extrema of certain multivariable functions. To solve the latter task, various methods (e.g. gradient type methods or relaxation methods) can be applied, see [7], [8] and references thereafter.

4 Fuzzy Classification as the Estimates Calculation

First note that the above quality functionals (or algorithm performance measures) take their values from 0 to 1. This allows us to define on $\{A\}$ a fuzzy set of "effective algorithms"

$$Ef_A := \langle \{A\}, \psi_A \rangle,$$

where ψ_A is one of the measures (9)-(11).

Then, assume that the marix Γ (6) contains no zero rows (as follows from the previous section, an operator r_A transforms the zero row to the row $(Na,\ldots,Na)$ of the matrix Cl_A, and no classification happens). Consider the matrix

$$M = M(\Gamma) = (\mu_{pj}(\Gamma))_{p=1,\ldots,q}^{j=1,\ldots,l};$$

$$\mu_{pj}(\Gamma) = \frac{\Gamma_{pj}}{\sum_{k=1}^{l} \Gamma_{pk}} = \frac{\Gamma_j(S_p)}{\sum_{k=1}^{l} \Gamma_k(S_p)};$$

i.e. this matrix is obtained from Γ by normalization its rows. As follows from Definition 1, the matrix M

represents a fuzzy classification of the set M_K, where $\mu_{pj}(\Gamma)$ is a grade of membership of $S_p \subset M_K$ in the class Kj.

Definition 2. The matrix M is called a fuzzy matrix of estimates.

The above decision rule for the algorithm A can be rewritten as

$$\text{i.} \quad \mu_j(S) - \mu_i(S) \geq \Delta_1', \quad j \neq i; \qquad (12)$$
$$\text{ii.} \quad \mu_j(S) \geq \Delta_2', \quad j = 1...l,$$

for some fixed Δ_1' и Δ_2'; : $0 < \Delta_1', \Delta_2' \leq 1$. The remaining parameters of A are related to the first (recognition) stage of the algorithm and we keep them unchanged.

Definition 3. An algorithm *F(A) with the following scheme*

$$F(A): \quad I \xrightarrow{R_{F(A)}} \mathrm{M} \xrightarrow{r_{F(A)}} Inf_{F(A)}$$

is called a fuzzy representation of the algorithm A of AEC model.

Here $R_{F(A)}$ and $r_{F(A)}$ are, respectively, the recognition operator and the decision rule (12) for *F(A)*.

Similarly to the algorithm A, *F(A)* can be encoded with the set of parameters (k, $\varepsilon_1,\ldots, \varepsilon_n, \varepsilon, \gamma_1, \ldots, \gamma_m, \Delta_1', \Delta_2'$), or with the smaller set (k, $\varepsilon_1,\ldots, \varepsilon_n, \varepsilon, \gamma_1, \ldots, \gamma_l, \Delta_1', \Delta_2'$) .Thus, if we replace in the set T the domains for parameters Δ_1 and Δ_2 by two intervals [0,1] (the domains for Δ_1' and Δ_2'), then the Statement 1 can be applied to establish the existence of extremal fuzzy algorithm on *F(T)*.

Thus, the set {*F(A)*}, generated from {A}, as the set of fuzzy classification algorithms has two advantages as compared to its counterparts: variability in constructing fuzzy classifications and existence of extremal algorithms.

Note that a connection between the algorithms based on computations of estimates and the fuzzy classification algorithms were indicated in [7] (in particular, on the stage related to the use of operators R_A), but the detailed information is missing.

As to the fuzzy classification algorithms considered in Section 2, they can be interpreted as algorithms of AEC model. To see this, it can be noted first that these fuzzy algorithms comprise two stages: the recognition (calculation of class memberships) and the classification itself (application of decision rule based on the highest membership value).

Further on, take the above algorithm F_K. Let M_R be the training set containing the objects $x_1,\ldots,x_m$ such that each object is identified with its vector feature set of length n, and y is an object from M_K identified with its feature vector of the same length. Then, we apply the recognition procedure similar to that of for the above-considered algorithms A except for some changes. Namely, instead of $\gamma(S_1), \gamma(S_2),\ldots, \gamma(S_m)$ we define the weights $\gamma_j(x_i,y)$, $i=1,\ldots,m$; $j=1,\ldots,l$, characterizing the importance of each x_i , considered as an element of class j, for the recognition of y, by the formula:

$$\gamma_j(x_i, y) = \mu_{ij}(x_i)\left(\frac{1}{\| y - x_i \|^{\frac{2}{P-1}}}\right),$$

where μ_{ij} are calculated according to (2) and P is the same as in (3)-(4). Next, from the elements (feature sets) of M_R, we form a table $T_{n,l*m,l}$ which contains l copies of the above table $T_{n,m,l}$. To each row of $T_{n,l*m,l}$ we consequently assign the weights $\gamma_1(x_1,y),\ldots \gamma_l(x_m,y)$. Assume that the Eucledian metric is selected as our distance function and $\delta = \sqrt{\delta_1^2 + \ldots + \delta_n^2}$ is the radius of the disc centered at y which contains exactly K neighbours of y from M_R. Then, if in the recognition procedure for the algorithm A we set $k=n$, $\varepsilon=0$ (every feature is taken into account for the distance measurement), $\varepsilon_1=\delta_1,\ldots, \varepsilon_n=\delta_n$, we find the following analog of (5)-(6) for our case

$$\Gamma_j(y) = \sum_{k=1}^{K} \gamma_j(x_k, y) := \sum_{k=1}^{K} \gamma_{jk}; \qquad (13)$$

where x_k are the nearest neighbors of y. The right-hand side of this equation is exactly the numerator in the right-hand side of (3). The decision rule for the considered algorithm (denote it as A_K) is also standard one for the AEC algorithms. Namely, the object y is assigned to the j-th class if the following condition holds:

$$\Gamma_j(y) - \Gamma_i(y) > 0, \quad j \neq i; \quad i = 1,\ldots,l.$$

As follows from (2) and (13), $\Gamma_j(y) > 0$ for all j; so each $y \in M_K$ will be classified.

Suppose that the values K_1 and P are fixed in F_K. Then the latter can be encoded with the set of parameters (n, $\varepsilon_1,\ldots, \varepsilon_n$, 0, $\gamma_{11}, \ldots, \gamma_{1m},\ldots, \gamma_{l1},\ldots, \gamma_{lm}$). The n epsilons characterize the dependence on K in the algorithm F_K. Assume that the domains for each of $n+lm$ parameters are specified. Denote a direct product of these domains as T_K, and as $\{A_K\}$ -the set of above AEC algorithms with parameters from T_K. Applying the similar arguments to the ones considered in the proof of Statement 1, we get the following results.

Statement 2. *For any K-fuzzy nearest neighbor algorithm there exists a corresponding AEC algorithm A_K with similar output. For the family $\{A_K\}$ there exist extremal (in terms of functions (9)-(11)) algorithms in the set T_K.*

For the denominator in the right-hand side of (3) we have, using (13) и (1):

$$\sum_{k=1}^{K}\frac{1}{\|y-x_k\|^{\frac{2}{m-1}}}=\sum_{k=1}^{K}\sum_{j=1}^{l}\mu_{jk}\frac{1}{\|y-x_k\|^{\frac{2}{m-1}}}=$$

$$=\sum_{j=1}^{l}\sum_{k=1}^{K}\mu_{jk}\frac{1}{\|y-x_k\|^{\frac{2}{m-1}}}=\sum_{j=1}^{l}\Gamma_j(y).$$

According to Definitions 2-3, the latter relation implies the following:

Statement 3. *The algorithm F_K is a fuzzy representation of the corresponding AEC algorithm A_K: $F(A_K)=F_K$.*

At the end of this section note that the above constructions can be applied to get the results similar to the Statements 2-3 for the above-considered nearest prototype algorithm.

5 Numerical Experiments

Here we empirically compare the performance of algorithms from the set $\{F(A)\}$ with that of the classifiers considered in Section 1. As a test set, we chose the recognition of irises from the known Fisher's Iris data set [11] . This data set has been widely used in statistical and machine learning studies. In particular, it was used in [10] for testing the fuzzy nearest neighbour algorithms. This data set represents three subspecies of irises: Iris setosa, Iris virginica and Iris versicolor with the four feature measurements being sepal length, sepal width, petal length and petal width. There are fifty vectors per class in this data set (150 vectors in total). The first class is linearly separable from the other two. Thus to each sample of the set corresponds its feature vector of length four. The classifications in [10] were obtained using the leave-one-out technique. The procedure is to leave one sample out of the sample set and classify it using the remaining (149 for the Fisher's set) samples as the training set. This technique is repeated until all samples in the data set have been classified. Alongside with the original Iris data set, the set Iris23 containing only the elements of the (non-separable) classes two and three (100 in total), was also considered in [10]. Also, in the fuzzy K- nearest neighbour algorithm the parameter P was set to be 2, and the parameters varied were K and K_1. The recognition results are presented in Table 1 containing the numbers of misclassified vectors of the Iris set for different values of K and K_1.

K \ K_1	1	3	5	7	9
1	6	6	6	6	6
2	6	6	6	6	6
3	5	5	5	5	6
4	5	5	5	5	5
5	4	4	5	5	5
6	4	4	4	4	4
7	4	4	4	4	4
8	4	4	4	4	4
9	4	4	4	4	4
Average	4.7	4.7	4.8	4.8	4.9

Table 1. Results of fuzzy K-nearest neighbor classification. Number of misclassified elements of Iris set.

The leave-one-out technique was also used in [10] for performance testing of the fuzzy nearest prototype algorithm. On each step, three nearest test sample's neighbors from different classes were selected as prototypes for each class. This algorithm is much faster compared to the K-nearest neighbor classifier. In [10] both the original feature vectors and the reduced vectors containing only two features (petal length and petal width) were used for the classification. The results (which are taken from the Table 4 of [10]) in the form of confusion matrices are shown in Tables 2 and 3. Their main diagonals contain the numbers of correctly classified samples from each class and off-diagonal cells – the numbers of misclassified ones.

Classes	1	2	3
1	50	0	0
2	0	45	5
3	0	7	43

Table 2. Confusion matrix of the nearest prototype classifier applied to Iris data (four features).

Classes	1	2	3
1	50	0	0
2	0	48	2
3	0	4	46

Table 3. Confusion matrix of the nearest prototype classifier applied to Iris data (two features).

As we see, in the latter case of incomplete feature vector the classification results are better. The performance of both the algorithms on the Iris23 was on average 20 percent worse than on the original data set. This can be explained by inseparability between the two classes of Iris23.

For the algorithms from $\{F(A)\}$ the parameters determining the set $T_{F(A)}$ were selected as follows: $\varepsilon=0$; $k \subset \{2,3,4\}$; $\gamma_1 = \ldots = \gamma_4 = 1$ (all four classes are equally important for classification), $\Delta_1' = 0.1$, $\Delta_2' = 0.6$. The average distances between the vectors from Iris set are: $\varepsilon_{1m}=1.8$, $\varepsilon_{2m}=1.2$, $\varepsilon_{3m}=2.95$, $\varepsilon_{4m}=1.2$. Accordingly, for each $i=1,\ldots,4$ we assumed that two ε_i-s are uniformly placed on the intervals $[0, \varepsilon_{im}]$. Thus, $48=3*2^4$ fuzzy classification algorithms associated with these parameters were built up. They were applied to both Iris and Iris23 data sets. As in the above cases of nearest neighbor classifiers, the training and the test sets were constructed on the leave-one-out basis. Below we give the results for the algorithms *F1* and *F2* associated with the parameters (4, 0.6, 0.4, 1, 0.4, 0, 1,1,1,1, 0.1, 0.6) and (2, 1.2, 0.8, 2, 0.8, 0, 1, 1, 1, 1, 0.1, 0.6) respectively. In particular, the confusion matrices for *F1* and *F2* applied to the Iris data set are contained in Tables 4-5.

Classes	1	2	3
1	49	0	1
2	0	46	4
3	0	1	49

Table 4. Confusion matrix for *F1* applied to Iris.

Classes	1	2	3
1	50	0	0
2	0	48	2
3	0	2	48

Table 5. Confusion matrix for *F2* applied to Iris.

As we see, the performance of *F1* and *F2* is comparable to that of the fuzzy *K*-nearest classifier and exceeds the performance of the fuzzy nearest prototype algorithm. The quality measure ψ_A^0 defined by (9) (the overall sensitivity) takes the values 0.96 and 0.97 on *F1* and *F2* respectively, whereas its values on the nearest prototype classifier are 0.92 (four feature vectors) and 0.96 (two feature vectors). The performance on Iris23 of the algorithms from $\{F(A)\}$ was also worse than on the original Iris set; however their performance is comparable to that of the fuzzy nearest neighbour classifiers. As an example, in Table 6 we give the confusion matrices for *F1* and *F2*.

	F1		*F2*	
Classes	2	3	2	3
2	46	4	47	3
3	11	39	7	43

Table 6. Confusion matrices for *F1* and *F2* applied to Iris23.

The performance of *F(A)* – algorithms with respect to *k* (the number of objects' features) was studied in the following manner. For example, for *F1* and *F2* we fixed all parameters except *k* and formed the families $\{F1\}$ and $\{F2\}$ containing each three algorithms with $k=2,3,4$. Then we calculated the overall sensitivity of the algorithms of each family. For Iris23 the results are given in Table 7.

k / ψ_A^0	2	3	4
{F1}	0.84	0.87	0.85
{F2}	0.9	0.87	0.87

Table 7. Overall sensitivity of algorithms of $\{F1\}$ and $\{F2\}$ families applied to Iris23.

In all the above tests, the training set (which contained all but one elements of Iris or Iris23) varied for each classification. In other words, we applied the leave-one-out cross-validation to evaluate performance of the algorithms. Also, some of our tests were performed with a fixed training set. Namely, in the Fisher data set, 60 samples (20 per each class) were randomly selected to form the set M_R, and the remaining 90 samples formed the test set M_K. The parameters of the algorithms from $\{F(A)\}$ were the same as above except for the ε_i-s; for $i=1,2$ four values of the latter were selected uniformly on the intervals $[0, \varepsilon_{im}]$, and for $i=3,4$ nine ε_i-s were selected similarly. Thus $3888=3*4^2*9^2$ algorithms were generated. In these tests the best results were shown by the algorithms *F3* and *F4* associated with the parameters (4, 1.08, 0.24, 0.6, 0.12, 0, 1, 1, 1, 1, 0.1, 0.6) and (2, 0.36, 0.24, 0.3, 0.12, 0, 1, 1, 1, 1, 0.1, 0.6) respectively. In both cases the value of (9) was equal to 0.98. A direct check showed that in the first case one object of class 2 was misclassified into class 3 and, conversely, one object of class 3 was assigned to class 2. In the second case two objects of class 2 were misclassified into class 3. To rank these algorithms, the measures similar to (10)-(11) which take into consideration the weights of classes can be applied.

k / ψ_A^0	2	3	4

{F3}	0.96	0.96	0.98
{F4}	0.98	0.96	0.97

Table 8. Overall sensitivity of algorithms of {*F3*} and {*F4*} families.

In the same manner as above, we studied the performance of the algorithms on a fixed training set with respect to *k*. For the families {*F3*} and {*F4*} the corresponding results are contained in Table 8.

6 Conclusion

As we have seen in the Section 4, the fuzzy nearest neighbor classifiers considered in the Section 3 can be embedded into the framework of AEC model, so all results on the latter can be applied to study them.

The above results have shown that the fuzzy representations of AEC algorithms can be practically used as classification algorithms. They do not require calculation of fuzzy class memberships for the elements of training set, and they are variable in calculation of fuzzy class memberships (or class estimates in terms of the AEC model) of objects to classify. In our tests their performance on the Iris data set was comparable with that of the nearest neighbor classifiers, while only few input parameters (*k* and ε_i-s) took different values. The existence of AEC algorithms, extremal in the above sense, is their other advantage. At the same time, in the classical AEC model the calculation of class estimates according to (5)-(6) involves all C_n^k reference subsets Ω_k, which may be time consuming for large *n* (size of the objects' feature vectors). In the latter case, a preliminary dimension reduction procedure is desirable. As their other disadvantage, we can point out a difficulty in establishing ranges for distances between the elements of feature vectors; however, it is natural to assume that the object to be classified is close to the training set. In such case, for finding these ranges, the elements of the training set can be used (as it was done in the above examples). As to the extremal algorithms (in our tests, *F3* and *F4* were found by exhaustive search) determining of their parameters (the extremums in $T_{F(A)}$) by analytical methods is a difficult task even when the amount of parameters is small, see [7] and references thereafter. The random search methods appear to be more practical (especially if several quality functionals are used for evaluation of algorithms), and their application for this purpose can be considered as an interesting open issue.

Also, one may enhance the performance of AEC algorithms by taking an arbitrary set of them and performing algebraic or logical correction of its elements (see [8] for the details). In particular, such correction can be applied to the fuzzy AEC algorithms.

As already mentioned in Section 1, in many practical applications of the fuzzy classifiers, only the calculations of fuzzy class memberships of objects to classify, are used. As a decision rule, quite complicated methods are often applied. In this connection, it is useful to evaluate how different algorithms perform these calculations. For this purpose, it seems promising to apply the fuzzy similarity measures introduced in [4] (it will require an additional study of the latter). We plan to address this issue in our future work.

References:

[1] Bezdek, J.C., Keller, J., Krisnapuram, R., Pal, N. *Fuzzy models and algorithms for pattern recognition and image processing. The handbooks of fuzzy set series. vol. 4,* Springer Science & Business Media, 2006.

[2] Gribkov, I.V., Koltsov, P.P., Kotovich, N.V., Kravchenko, A.A., Koutsaev, A.S., Osipov, A.S., & Zakharov, A.V. Empirical evaluation of Image Processing Methods Using PICASSO 2 System. *WSEAS Transactions on Systems,* Vol.4, No.11, 2005, pp. 1923-1930.

[3] A. Osipov, A fuzzy approach to performance evaluation of edge detectors, *Proceedings of the 7th WSEAS International Conference on Signal, Sppeech and Image Processing (SSIP'07),* Beiging, China, September 15-17 2007, pp. 94-99.

[4] A. Osipov, Some fuzzy tools for evaluation of computer vision algorithms, *International Journal of Computer Vision and Image Processing,* Vol.8, Issue 1, 2018, pp. 1-14.

[5] Yu. I. Zhuravlev, V. V. Nikiforov, Recognition algorithms based on computation of estimates, *Cybernetics,* Vol.7, No.3, 1971, pp. 387-400.

[6] Yu. I. Zhuravlev, Extreme problems arising in determination of heuristic procedures. *Probl. Prikl. Mat. I Mech.* (*Problems of Applied Mathematics and Mechanics - Collection of Works*), Moscow, Nauka Press, 1971, pp. 67-74. (in Russian).

[7] Yu. I. Zhuravlev, An algebraic approach to recognition or classifications problems. *Patttern Recognition and Image Analysis,* Vol.8, No.1, 1998, pp. 59-100. (Russian original in *Problemy Kibernetiki*, No 33, 1978, pp. 5-68.)

[8] Ablameyko, S.V., Biryukov, A.S., Dokukin, A.A. et al., Practical algorithms for algebraic and logical correction in precedent-based recognition problems. *Comput. Math. and Math. Phys.*, Vol. 54, 2014, pp. 1915–1928.

[9] I. Gurevich, V. Yashina., Descriptive image analysis. Foundations and descriptive image algebras. *Int. Journal of Pattern Recognition and Artificial Intelligence,* Vol.33, No.11, 1940018, 2019, pp. 1-25.

[10] Keller, J.R., Gray, M.R., & Givern J.A., A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man and Cybernetics,* Vol.15, No.4, 1985, pp. 580-585.

[11] O. Hegazy, S. Safwat, M. E. Bakry, A mapreduce fuzzy techniques of big data classification. *Proceedings of 2016 SAI Computing Conference (SAI)*, London, 13-15 July , 2016, pp. 118-128.

[12] J. Maillo, S. García, J. Luengo, F. Herrera, I. Triguero, Fast and Scalable Approaches to Accelerate the Fuzzy k-Nearest Neighbors Classifier for Big Data. *IEEE Transactions on Fuzzy Systems,* Vol. 28, No. 5, 2020, pp. 874-886.

[13] Kwak, K.C., Pedry, W., Face recognition using a fuzzy fisher classifier. *Pattern Recognition*, Vol. 38, No.10, 2005, pp. 1717–1732.

[14] Xiaodong Li, Face recognition method based on fuzzy 2DPCA, *Journal of Electrical and Computer Engineering,* Vol. 2014, Article ID 919041, 2014, 7 pages.

[15] Koltsov, P.P.,Osipov, A.S., Kutsaev, A.S., Kravchenko, A.A., Kotovich, N.V., Zakharov, A.V. On the quantitative performance evaluation of image analysis algorithms. *Computer Optics,* Vol. 39, No.4, 2015, pp. 542-556.

[16] **Fisher's** Iris data set. Avaliable at: http://archive.ics.uci.edu/ml/datasets/Iris

Creative Commons Attribution License 4.0 (Attribution 4.0 International , CC BY 4.0)