Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2022) 30: 871 – 890

doi:10.3906/elk-2106-28

A new classification method using soft decision-making based on an aggregation operator of fuzzy parameterized fuzzy soft matrices

Samet MEMİŞ[1,*], **Serdar ENGİNOĞLU**[2], **Uğur ERKAN**[3]

[1]Department of Computer Engineering, Faculty of Engineering and Natural Sciences, İstanbul Rumeli University, İstanbul, Turkey

[2]Department of Mathematics, Faculty of Arts and Sciences, Çanakkale Onsekiz Mart University, Çanakkale, Turkey

[3]Department of Computer Engineering, Faculty of Engineering, Karamanoğlu Mehmetbey University, Karaman, Turkey

Received: 04.06.2021 • **Accepted/Published Online:** 06.12.2021 • **Final Version:** 21.03.2022

Abstract: Recently, a precise and stable machine learning algorithm, i.e. eigenvalue classification method (EigenClass), has been developed by using the concept of generalised eigenvalues in contrast to common approaches, such as k-nearest neighbours, support vector machines, and decision trees. In this paper, we offer a new classification algorithm called fuzzy parameterized fuzzy soft aggregation classifier (FPFS-AC) to combine the modelling ability of soft decision-making (SDM) and classification success of generalised eigenvalues. FPFS-AC constructs a decision matrix by employing the similarity measures of fuzzy parameterized fuzzy soft matrices (*fpfs*-matrices) and a generalised eigenvalue-based similarity measure. Then, it applies an SDM method based on the aggregation operator of *fpfs*-matrices to a decision matrix and classifies the given test sample. Afterwards, we perform an experimental study using 15 UCI datasets to manifest the success of our approach and compare FPFS-AC with the well-known and state-of-the-art classifiers (kNN, SVM, fuzzy kNN, EigenClass, and BM-fuzzy kNN) in terms of accuracy, precision, recall, macro F-score, micro F-score, and running time. Moreover, we statistically analyse the experimentally obtained data. Experimental and statistical results show that FPFS-AC outperforms the state-of-the-art classifiers in all the datasets concerning the five performance metrics.

Key words: Fuzzy sets, soft sets, soft decision-making (SDM), *fpfs*-matrices, supervised learning, data classification

1. Introduction

An excess of data and many uncertainties are encountered in a great number of fields, including space sciences, meteorology, defence industry, medicine, psychology, and finance. Therefore, some data-processing technologies, such as machine learning, are needed to handle the data in the aforesaid fields more effectively. Supervised learning is a widely-employed subfield of machine learning for this purpose [1]. One of the most popular supervised learning techniques is classification, in which the main goal of the classifier is to predict the class of unlabelled data (testing set) using the information of the labelled data (training set). To this end, in the literature, various classification algorithms have been introduced. The well-known classification algorithms are k-nearest neighbour (kNN) [2, 3] and support vector machines (SVM) [4]. These two classifiers have been applied to many areas from medical diagnostics to finance and are still in use. To enhance the classification performance of these well-known classifiers, the concept of fuzzy sets [5] has been availed of, listed among the widespread mathematical tools defined to deal with uncertainty. For example, fuzzy k-nearest neighbour (fuzzy

*Correspondence: samettmemis@gmail.com

kNN) [6] utilises a fuzzy membership degree concerning the distance of each neighbour to the test instance to weight each k-nearest neighbour.

Unlike the aforementioned approaches, eigenvalue classification method (EigenClass) [7] based on generalised eigenvalues has been proposed in recent times. EigenClass is a precise and stable classifier thanks to the employed generalised eigenvalue-based quasi-distance. One of the other state-of-the-art classifiers is fuzzy kNN classifier based on the Bonferroni mean (BM-fuzzy kNN) [8]. BM-fuzzy kNN computes the Bonferroni mean vectors of kNNs splitting them into subsamples in terms of their classes. It then calculates the membership degree of the query instance by means of Euclidean distances between the Bonferroni mean vectors and the query instance. Finally, the label of the highest membership degree is assigned to the query instance.

Besides the fuzzy sets successfully applied in machine learning as mentioned above, the concept of soft sets [9] has been propounded by Molodtsov to overcome various uncertainties as a new mathematical tool and applied to assorted fields from algebra to medical diagnostics over the last two decades [10–17]. Soft sets have led to the emergence of new fields, including soft algebra [22–24], soft topology [25–27], soft analysis [28], and soft decision-making (SDM) [29–31], which have given birth to their various applications [18–21]. Moreover, hybrid versions of fuzzy sets and soft sets, such as fuzzy soft sets [32, 33], fuzzy parameterized soft sets [34], and fuzzy parameterized fuzzy soft sets (*fpfs*-sets) [35] have been put forward to model further uncertainties than fuzzy uncertainty and applied to several decision-making problems. Afterwards, soft matrices [36], fuzzy soft matrices [37], and fuzzy parameterized fuzzy soft matrices (*fpfs*-matrices) [38] have been propounded to process a large number of data faster and more effectively. However, most of the applications therein have been carried out by using fictitious problems and data [39]. Only a few have been applied to real-world problems, e.g., classification problem in machine learning [40–43] and performance-based value assignment problem in image denoising [44–46]. Although the proposed classifiers employing fuzzy soft sets in the studies above are real-world applications, they have exhibited limited classification performance due to their working principles' reliance on by-class averaging of the training data and they fail to consider parameters' effects on classification. To deal with these drawbacks, a number of studies [48–50] have introduced similarity and distance measures of *fpfs*-matrices, which can model problems containing fuzzy parameters/objects and consider parameters' impacts on the classification.

In this study, we propose a new classification algorithm, i.e. fuzzy parameterized fuzzy soft aggregation classifier (FPFS-AC), via the SDM method CCE10 [10, 35] based on an aggregation operator of *fpfs*-matrices to utilise multisimilarity measures of *fpfs*-matrices and generalised eigenvalue-based similarity measure. Our main goal herein is to avail of modelling skills of each similarity measure of *fpfs*-matrices and classification ability of generalised eigenvalues to offer a more precise and stable classification method than EigenClass in supervised learning. In general, various similarity measures have different classification abilities. It is not straightforward to figure out which one is more convenient than the others for any classification task. Even if a proper similarity measure is determined, repeating the determination process may be required for each dataset. In this paper, the idea of simultaneously employing several similarity measures in the same classification task is considered to overcome these drawbacks. Moreover, the pseudo-similarities of *fpfs*-matrices, whose modelling abilities are manifested in the recent literature [48–50], and an eigenvalue-based quasi-similarity, defined herein by using the eigenvalue-based quasi-metric [7], are utilised for the aforesaid purpose to achieve high classification performance. Besides, an SDM method, i.e. CCE10, based on an aggregation operator of *fpfs*-matrices is applied to the decision-making problem related to the prediction of the test sample's class label under the aforesaid similarity measures. The main reason for choosing CCE10, it has efficacious modelling skills for multicriteria

decision-making problems. The major contributions of the present study can be summed up as follows:

- Similarity measures of *fpfs*-matrices were applied to supervised learning.
- An SDM method constructed via *fpfs*-matrices was applied to supervised learning.
- A generalised eigenvalue-based similarity measure was offered.
- Multi-similarity measures of *fpfs*-matrices and the generalised eigenvalue-based similarity measure were simultaneously employed for data classification.
- A new classification algorithm referred to as FPFS-AC was developed.

Section 2 of the present study provides the definitions of *fpfs*-sets, *fpfs*-matrices, similarity measures of *fpfs*-matrices, and pseudocode of CCE10 required in the next sections. Section 3 presents some basic notations needed for the FPFS-AC algorithm, a generalised eigenvalue-based similarity measure, and FPFS-AC. Section 4 firstly provides the properties of the University of California-Irvine (UCI) datasets used herein and mathematical notations of the performance metrics accuracy, precision, recall, macro F-score, and micro F-score. Secondly, the section performs an experimental study employing the 15 UCI datasets. It then compares FPFS-AC with the well-known and state-of-the-art classifiers, namely kNN, SVM, fuzzy kNN, EigenClass, and BM-fuzzy kNN, in terms of the aforesaid performance metrics and running time. Thirdly, it analyses the comparison results and presents the Nemenyi diagrams for each performance metric. The final section makes some suggestions and provides some conclusive remarks for further research. This study was derived from the first author's PhD dissertation.

2. Preliminaries

In this section, we first present some of the basic definitions needed for the following sections. Throughout this paper, let E be a parameter set, $F(E)$ be the set of all fuzzy sets over E, and $\mu \in F(E)$. Here, $\mu := \{^{\mu(x)}x : x \in E\}$.

Definition 1 [35] *Let U be a universal set, $\mu \in F(E)$, and α be a function from μ to $F(U)$. Then, the set $\left\{(^{\mu(x)}x, \alpha(^{\mu(x)}x)) : x \in E\right\}$, the graphic of α, is called a fuzzy parameterized fuzzy soft set (fpfs-set) parameterized via E over U (or briefly over U).*

Throughout the study, the set of all *fpfs*-sets over U is denoted by $FPFS_E(U)$. In $FPFS_E(U)$, since the graph(α) and α generate each other uniquely, the notations are interchangeable. Therefore, as long as it causes no confusion, we denote an *fpfs*-set graph(α) by α.

Example 1 *Let $E = \{x_1, x_2, x_3\}$ and $U = \{u_1, u_2, u_3\}$. Then,*

$$\alpha = \left\{(^1x_1, \{^{0.5}u_1, ^{0.2}u_2, ^{0.4}u_3\}), (^{0.2}x_2, \{^{0.1}u_1, ^{0.1}u_2, ^{0.8}u_3\}), (^{0.4}x_3, \{^1u_1, ^{0.5}u_2, ^1u_3\})\right\}$$

is an fpfs-sets over U.

Definition 2 [38] *Let $\alpha \in FPFS_E(U)$. Then, $[a_{ij}]$ is called the fpfs-matrix of α and is defined by*

$$[a_{ij}] := \begin{bmatrix} a_{01} & a_{02} & a_{03} & \dots & a_{0n} & \dots \\ a_{11} & a_{12} & a_{13} & \dots & a_{1n} & \dots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} & \dots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots \end{bmatrix}$$

such that for $i \in \{0, 1, 2, \cdots\}$ and $j \in \{1, 2, \cdots\}$,

$$a_{ij} := \begin{cases} \mu(x_j), & i = 0 \\ \alpha^{(\mu(x_j)}x_j)(u_i), & i \neq 0 \end{cases}$$

Here, if $|U| = m - 1$ and $|E| = n$, then $[a_{ij}]$ has order $m \times n$.

Hereinafter, the set of all *fpfs*-matrices parameterized via E over U is denoted by $FPFS_E[U]$ and let $[a_{ij}], [b_{ij}], [c_{ij}] \in FPFS_E[U]$. Moreover, Let I_m denote the set of all unsigned integer numbers from 1 to m, i.e. $I_m := \{1, 2, \dots, m\}$. Similarly, let $I_m^* := \{0, 1, 2, \dots, m\}$.

Example 2 *Let us consider the α provided in Example 1. From the Definition 2, all the entries of the fpfs-matrices of α are obtained as $a_{01} = \mu(x_1) = 1$, $a_{02} = \mu(x_2) = 0.2$, $a_{03} = \mu(x_3) = 0.4$, $a_{11} = \alpha^{(\mu(x_1)}x_1)(u_1) = 0.5$, $a_{12} = \alpha^{(\mu(x_2)}x_2)(u_1) = 0.1$, $a_{13} = \alpha^{(\mu(x_3)}x_3)(u_1) = 1$, $a_{21} = \alpha^{(\mu(x_1)}x_1)(u_2) = 0.2$, $a_{22} = \alpha^{(\mu(x_2)}x_2)(u_2) = 0.1$, $a_{23} = \alpha^{(\mu(x_3)}x_3)(u_2) = 0.5$, $a_{31} = \alpha^{(\mu(x_1)}x_1)(u_3) = 0.4$, $a_{32} = \alpha^{(\mu(x_2)}x_2)(u_3) = 0.8$, and $a_{33} = \alpha^{(\mu(x_3)}x_3)(u_3) = 1$. Then, the fpfs-matrices of α is*

$$[a_{ij}] = \begin{bmatrix} a_{01} & a_{02} & a_{03} \\ a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0.2 & 0.4 \\ 0.5 & 0.1 & 1 \\ 0.2 & 0.1 & 0.5 \\ 0.4 & 0.8 & 1 \end{bmatrix}$$

Definition 3 [38] *Let $[a_{ij}] \in FPFS_E[U]$. For all i and j, if $a_{ij} = \lambda$, then $[a_{ij}]$ is called λ-fpfs-matrix and is denoted by $[\lambda]$. Here, $[0]$ and $[1]$ are called empty fpfs-matrix and universal fpfs-matrix, respectively.*

Definition 4 [38] *Let $[a_{ij}], [b_{ij}] \in FPFS_E[U]$. For all i and j,*

If $a_{ij} = b_{ij}$, then $[a_{ij}]$ and $[b_{ij}]$ are called equal fpfs-matrices and is denoted by $[a_{ij}] = [b_{ij}]$.

If $a_{ij} \leq b_{ij}$, then $[a_{ij}]$ is called a submatrix of $[b_{ij}]$ and is denoted by $[a_{ij}] \tilde{\subseteq} [b_{ij}]$.

If $[a_{ij}] \tilde{\subseteq} [b_{ij}]$ and $[a_{ij}] \neq [b_{ij}]$, then $[a_{ij}]$ is called a proper submatrix of $[b_{ij}]$ and is denoted by $[a_{ij}] \tilde{\subsetneq} [b_{ij}]$.

Definition 5 [48]*Let $s : FPFS_E[U] \times FPFS_E[U] \to \mathbb{R}$ be a mapping. Then, for all $[a_{ij}], [b_{ij}], \in FPFS_E[U]$, s is pseudo-similarity over $FPFS_E[U]$ if and only if s satisfies the following properties:*

i) $s([a_{ij}], [a_{ij}]) = 1$
ii) $s([a_{ij}], [b_{ij}]) = s([b_{ij}], [a_{ij}])$
iii) $0 \leq s([a_{ij}], [b_{ij}]) \leq 1$

Proposition 1 [50] *The mapping* s_H *defined by* $s_H([a_{ij}],[b_{ij}]) := 1 - \frac{1}{(m-1)n}\sum_{i=1}^{m-1}\sum_{j=1}^{n}|a_{0j}a_{ij} - b_{0j}b_{ij}|$ *is a pseudo-similarity over* $FPFS_E[U]$ *and is called Hamming pseudo-similarity.*

Proposition 2 [48] *The mapping* s_E *defined by* $s_E([a_{ij}],[b_{ij}]) := 1-\frac{1}{\sqrt{(m-1)n}}\left(\sum_{i=1}^{m-1}\sum_{j=1}^{n}|a_{0j}a_{ij} - b_{0j}b_{ij}|^2\right)^{\frac{1}{2}}$ *is a pseudo-similarity over* $FPFS_E[U]$ *and is called Euclidean pseudo-similarity.*

Proposition 3 [48] *The mapping* s_{Hs} *defined by* $s_{Hs}([a_{ij}],[b_{ij}]) := 1 - \frac{1}{m-1}\sum_{i=1}^{m-1}\max_{j\in I_n}\{|a_{0j}a_{ij} - b_{0j}b_{ij}|\}$ *is a pseudo-similarity over* $FPFS_E[U]$ *and is called Hausdorff pseudo-similarity.*

Proposition 4 [48] *The mapping* s_M^p *defined by* $s_M^p([a_{ij}],[b_{ij}]) := 1-\frac{1}{\sqrt[p]{(m-1)n}}\left(\sum_{i=1}^{m-1}\sum_{j=1}^{n}|a_{0j}a_{ij} - b_{0j}b_{ij}|^p\right)^{\frac{1}{p}}$ *is a pseudo-similarity over* $FPFS_E[U]$ *and is called Minkowski pseudo-similarity. Here* $p \in \mathbb{N}^+$.

Definition 6 [7] *Let* $A, B \in M_{n\times n}(\mathbb{R})$ *and* φ *be a nonzero n-dimensional vector. If there exists a scalar* λ *such that* $A\varphi = \lambda B\varphi$, *then* λ *is called generalised eigenvalue of* A *according to* B *or briefly eigenvalue of* A *according to* B. *The vector which contains all eigenvalues of* A *according to* B *is denoted by* $\operatorname{eig}(A,B) = \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_n \end{bmatrix}$.

Definition 7 [7] *Let* $u \in \mathbb{R}^n$. *Then, diagonal form of* $u := (u_1, u_2, \ldots, u_n)$ *is* $\begin{bmatrix} u_1 & 0 & \cdots & 0 \\ 0 & u_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_n \end{bmatrix}$ *and is denoted by* $\operatorname{diag}(u)$.

Definition 8 [7] *Let* A *and* B *be two diagonal matrices whose diagonal entries differ from zero. Then, the mapping* d_{ev} *defined by* $d_{ev}(A,B) := \sum\left|\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} - \operatorname{eig}(A,B)\right|$ *is called* A'*s quasi-distance to* B. *Here,* $\sum A$ *stands for the sum of all the entries of* A *and that* $|A|$ *represents a matrix whose entries equal the absolute values of the entries of* A.

Secondly, we present the pseudocode of CCE10 [10, 35] in Algorithm 1.

3. Fuzzy parameterized fuzzy soft aggregation classifier (FPFS-AC)

This section first provides the definitions and notations occurring in FPFS-AC. Across the present study, let $D = [d_{ij}]_{m\times(n+1)}$ denotes a data matrix and its last column contains class labels of the data. Here, m and n stand for the number of the samples and the number of the attributes in the data matrix, respectively. $(D_{train})_{m_1\times n}$, $(C)_{m_1\times 1}$, and $(D_{test})_{m_2\times n}$ represent the training matrix, class labels of the training matrix, and the test matrix obtained from D, respectively, such that $m_1 + m_2 = m$. $D_{i-train}$ and D_{i-test} denote i^{th} row of D_{train} and D_{test}, respectively. Similarly, $D_{train-j}$ and D_{test-j} denote j^{th} column of D_{train} and D_{test},

Algorithm 1 Pseudocode of CCE10

Input: *fpfs*-matrix $[a_{ij}]_{m\times n}$
Output: Score matrix $[s_{i1}]_{(m-1)\times 1}$, Decision matrix $[dm_{i1}]$, and Optimum alternatives' matrix $[op_{i1}]$

$[s] \leftarrow [0]_{(m-1)\times 1}$
for i from 1 to $m-1$ **do**
  for j from 1 to n **do**
    $s_{i1} \leftarrow s_{i1} + a_{0j}a_{ij}$
  end for
  $s_{i1} \leftarrow \frac{s_{i1}}{n}$
end for
for i from 1 to $m-1$ **do**
  $dm_{i1} \leftarrow \frac{s_{i1}}{\max\limits_{k\in I_{m-1}}\{s_{k1}\}}$
end for
$[op] \leftarrow \underset{k\in I_{m-1}}{\operatorname{argmax}}\{dm_{k1}\}$

respectively. $T_{m_2\times 1}$ and $T'_{m_2\times 1}$ stand for ground truth class matrix and predicted class matrix obtained from D_{train} and D_{test}, respectively.

Definition 9 *Let $u, v \in \mathbb{R}^n$. Then, Pearson correlation coefficient between u and v is defined by*

$$P(u,v) := \frac{n\sum_{i=1}^{n} u_i v_i - (\sum_{i=1}^{n} u_i)(\sum_{i=1}^{n} v_i)}{\sqrt{[n\sum_{i=1}^{n} u_i^2 - (\sum_{i=1}^{n} u_i)^2]\,[n\sum_{i=1}^{n} v_i^2 - (\sum_{i=1}^{n} v_i)^2]}}$$

Definition 10 *Let D_{train} has order $m_1 \times n$ and $C_{m_1\times 1}$ be the class column vector of D_{train}. fw is called the feature weight vector based on Pearson correlation coefficient of D_{train} and is defined by $fw_{j1} := |P(D_{train-j}, C)|$, $j \in I_n$*

Definition 11 *Let D_{train} has order $m_1 \times n$ and D_{test} has order $m_2 \times n$. $\widetilde{D}_{train}$ is called the feature fuzzifications of D_{train} and is defined by $\widetilde{d}_{ij-train} := \frac{d_{ij-train} - \min\limits_{r,s}\{d_{rj-train}, d_{sj-test}\}}{\max\limits_{r,s}\{d_{rj-train}, d_{sj-test}\} - \min\limits_{r,s}\{d_{rj-train}, d_{sj-test}\}}$ such that $i, r \in I_{m_1}$, $s \in I_{m_2}$, and $j \in I_n$.*

Definition 12 *Let D_{train} has order $m_1 \times n$ and D_{test} has order $m_2 \times n$. $\widetilde{D}_{test}$ is called the feature fuzzifications of D_{test}, and is defined by $\widetilde{d}_{ij-test} := \frac{d_{ij-test} - \min\limits_{r,s}\{d_{rj-train}, d_{sj-test}\}}{\max\limits_{r,s}\{d_{rj-train}, d_{sj-test}\} - \min\limits_{r,s}\{d_{rj-train}, d_{sj-test}\}}$ such that $r \in I_{m_1}$, $i, s \in I_{m_2}$, and $j \in I_n$.*

Definition 13 *Let A and B be two diagonal matrices whose diagonal entries differ from zero. Then, the mapping s_{ev} defined by $s_{ev}(A,B) := 1 - \left(\frac{2}{\pi}\arctan\left(d_{ev}(A,B)\right)\right)$ is called A's quasi-similarity to B based on generalised eigenvalues.*

This section then offers a new classification algorithm, i.e. FPFS-AC, based on the Hamming, Euclidean, Hausdorff, and Minkowski pseudo-similarities of *fpfs*-matrices and the generalised eigenvalue-based quasi-similarity. Its pseudocode is provided in Algorithm 2.

FPFS-AC employs Pearson correlation coefficient to obtain parameter weights based on their impacts on classification. Afterwards, it constructs two *fpfs*-matrices, namely train *fpfs*-matrix and test *fpfs*-matrix, via

Algorithm 2 Pseudocode of FPFS-AC

Input: $(D_{train})_{m_1\times n}, C_{m_1\times 1}$, and $(D_{test})_{m_2\times n}$
Output: $T'_{m_2\times 1}$

procedure FPFS-AC(D_{train}, C, D_{test})
  Compute fw using D_{train} and C
  Compute feature fuzzification of D_{train} and D_{test}, namely $\widetilde{D}_{train}$ and $\widetilde{D}_{test}$
  for i from 1 to m_2 **do**
    Compute the test *fpfs*-matrix $[a_{ij}]$ using fw and $\widetilde{D}_{i-test}$
    for j from 1 to m_1 **do**
      Compute the train *fpfs*-matrix $[b_{ij}]$ using fw and $\widetilde{D}_{j-train}$
      $f_{j1} \leftarrow s_H([a_{ij}],[b_{ij}])$
      $f_{j2} \leftarrow s_E([a_{ij}],[b_{ij}])$
      $f_{j3} \leftarrow s_{Hs}([a_{ij}],[b_{ij}])$
      $f_{j4} \leftarrow s^3_M([a_{ij}],[b_{ij}])$
      for all i and j **do**
        if $\widetilde{d}_{ij-train} = 0$ **then**
          $\widetilde{d}_{ij-train} \leftarrow 0.0001$
        end if
        if $\widetilde{d}_{ij-test} = 0$ **then**
          $\widetilde{d}_{ij-test} \leftarrow 0.0001$
        end if
      end for
      $f_{j5} \leftarrow s_{ev}(\mathrm{diag}(\widetilde{D}_{j-train}), \mathrm{diag}(\widetilde{D}_{i-test}))$
    end for
    for j from 1 to 5 **do**
      $sd_j \leftarrow \mathrm{std}(F^j)$
    end for
    $pw \leftarrow (1 - \widehat{sd})$
    Compute *fpfs*-matrix $[g_{ij}]$ using pw and F for soft decison-making
    $[[s_{k1}],[dm_{k1}],[op_{k1}]] \leftarrow \mathrm{CCE10}([g_{ij}])$
    $t'_{i1} \leftarrow C(op_{11}, 1)$
  end for
  return $T'_{m_2\times 1}$
end procedure

normalised train instance, normalised test instance, and parameter weights. Thereafter, the proposed classifier assigns the class label of the optimum train instance, obtained by CCE10, to the test instance. This process is similar in all the test instances. Finally, the predicted class matrix of the test data is constructed.

4. Experimental study

In this section, we detail the properties of the 15 classification datasets in the UCI machine learning repository [51]. We then present five performance metrics for performance evaluation in machine learning. Next, we perform some experiments to show that our proposed method is more efficient than kNN [3], fuzzy kNN [6], SVM [4], EigenClass [7], and BM-fuzzy kNN [8]. Finally, we carry out the statistical evaluation of the experimental results based on Friedman test [52] and Nemenyi post-hoc test [53].

4.1. UCI datasets and performance measures

In Table 1, we firstly present the properties of the datasets [51] used in the simulation herein: "Statlog (Australian credit approval)", "Banknote", "Breast tissue", "Cryotherapy", "Glass", "Hayes-Roth", "Ionosphere", "Iris", "Mice protein expression", "Parkinsons[sic]", "Parkinson's disease", "Image segmentation", "Connectionist bench (sonar, mines vs. rocks)", "Teaching assistant evaluation", and "Connectionist bench (vowel recognition-Deterding data)".

Table 1. Description of UCI datasets. (# stands for the number of)

No.	Name	#Instance	#Attribute	#Class
1	Australian	690	14	2
2	Banknote	1372	4	2
3	Breast Tissue	109	9	6
4	Cryotherapy	90	6	2
5	Glass	214	9	7
6	Hayes-Roth	132	5	3
7	Ionosphere	351	34	2
8	Iris	150	4	3
9	Mice	1077	72	8
10	Parkinsons[sic]	195	22	2
11	Parkinson's disease	756	754	2
12	Image segmentation	2310	19	7
13	Sonar	208	60	2
14	Teaching	151	5	3
15	Vowel	990	13	11

We subsequently provide the mathematical notations of five performance metrics, i.e. accuracy (Acc), precision(Pre), recall (Rec), macro F-score (MacF), and micro F-score (MicF), to compare the aforementioned methods. Let $D_{test} = \{x_1, x_2, \ldots, x_n\}$, $T = \{T_1, T_2, \ldots, T_n\}$, $T' = \{T'_1, T'_2, \ldots, T'_n\}$, and l be n samples to be classified, ground truth class sets of the samples, prediction class sets of the samples, and the number of the class of the samples, respectively.

$$\mathrm{Acc}(T,T') := \frac{1}{l}\sum_{i=1}^{l}\frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i},\ \mathrm{Pre}(T,T') := \frac{1}{l}\sum_{i=1}^{l}\frac{TP_i}{TP_i + FP_i}$$

$$\mathrm{Rec}(T,T') := \frac{1}{l}\sum_{i=1}^{l}\frac{TP_i}{TP_i + FN_i},\ \mathrm{MacF}(T,T') := \frac{1}{l}\sum_{i=1}^{l}\frac{2TP_i}{2TP_i + FP_i + FN_i}$$

$$\mathrm{MicF}(T,T') := \frac{2\sum_{i=1}^{l} TP_i}{2\sum_{i=1}^{l} TP_i + \sum_{i=1}^{l} FP_i + \sum_{i=1}^{l} FN_i}$$

where TP_i, TN_i, FP_i, and FN_i are the number of true positive, true negative, false positive, and false negative for the class i, respectively, and their mathematical notations are as follows:

$$TP_i := \left|\{x_k \,|\, i \in T_k \wedge i \in T'_k, 1 \le k \le l\}\right|, \quad TN_i := \left|\{x_k \,|\, i \notin T_k \wedge i \notin T'_k, 1 \le k \le l\}\right|$$

$$FP_i := |\{x_k \,|\, i \notin T_k \wedge i \in T'_k, 1 \leq k \leq l\}|, \quad FN_i := |\{x_k \,|\, i \in T_k \wedge i \notin T'_k, 1 \leq k \leq l\}|$$

4.2. Simulation results

In this part of the present paper, we focus on the comparison between our proposed FPFS-AC and the well-known and state-of-the-art classifiers, i.e. kNN [3], fuzzy kNN [6], SVM [4], EigenClass [7], and BM-fuzzy kNN [8]. We perform the simulation of the algorithms by utilising MATLAB R2020b and a workstation with I(R) Xeon(R) CPU E5-1620 v4 @ 3.5 GHz and 64 GB RAM. Each classifier is trained and tested by employing the k-fold cross-validation [54], in which the dataset is split into equal-sized k-part subsamples. This process is randomly carried out. One subsample is kept as validating data (testing data) and the remaining $k-1$ subsamples are operationalised to train the algorithm. Since the cross-validation process is repeated k times, each subsample is made use of only once as validating data. Thus, the entire dataset is exploited as both training and testing data.

The higher value of k results in a less biased model that large variance might get around to over-fit, whereas its lower value is like the train-test split approach. Moreover, the higher value of k leads to higher running time. As k gets larger, the difference in size between the training set and the resampling subsets gets smaller. As this difference decreases, the bias of the technique becomes smaller. Therefore, in most research, 5-folds and 10-folds are commonly employed as k-folds cross-validation [55]. In this study, we choose the k value as 5 for the cross-validation throughout the simulation. Here, utilising 5-folds cross-validation provides that a split of data 80% is a training set, and 20% is a testing set. In 5-fold cross-validation, the dataset is randomly divided into five parts. One part is used for testing, and the remaining four parts are used for training. This process is repeated five times, with each part being used as test data once. Afterwards, to obtain more reliable performance results, 10 runs are carried out, and average Acc, Pre, Rec, MacF, MicF, and running time results are obtained. The number of runs, i.e. 10 herein, is one of the commonly used numbers in the literature. Consequently, the performance results of a machine-learning algorithm in one run are avoided from being high by chance, and the results are stable.

Table 2 presents the accuracy, precision, recall, macro F-score, micro F-score, and running time results of the methods for the datasets. The results show that FPFS-AC produces the best performance in the datasets in terms of accuracy $(75\% - 100\%)$, precision $(64\% - 100\%)$, recall $(63\% - 100\%)$, macro F-score $(62\% - 100\%)$, and micro F-score $(63\% - 100\%)$ performance. Especially in the "Parkinson's disease" and "Teaching" datasets, FPFS-AC performs far better than the others. Additionally, in the other datasets, where the overall performance results do not exceed 90%, FPFS-AC outperforms the others. Furthermore, in "mice protein", the performance of FPFS-AC, just as of SVM, is 100% as far as the performance metrics are concerned.

Thanks to FPFS-AC's employing four pseudo-similarities of *fpfs*-matrices based on Pearson correlation coefficient and generalised eigenvalue-based quasi-similarity and obtaining the optimum training label for the test instances by utilising CCE10, it achieves remarkable classification success. On the other hand, employing the CCE10 by calculating the four pseudo-similarities and generalised eigenvalue-based quasi-similarity causes FPFS-AC to run slightly slower than the others except SVM. As clear from the mean results in Table 2, FPFS-AC is a more efficacious method than kNN, fuzzy kNN, SVM, EigenClass, and BM-fuzzy kNN.

Table 2. Comparative results for the datasets.

Datasets	Classifiers	Acc±SD	Pre±SD	Rec±SD	MacF±SD	MicF±SD	Running Time±SD
Australian	kNN	82.33±2.63	82.29±2.65	81.98±2.72	82.04±2.69	82.33±2.63	0.11432±0.06500
	SVM	76.35±10.25	79.48±10.57	75.27±11.18	74.33±12.04	76.35±10.25	4.36519±0.35485
	Fuzzy kNN	67.43±3.08	67.17±3.22	66.37±3.33	66.38±3.38	67.43±3.08	0.00628±0.00137
	EigenClass	81.62±3.47	82.34±2.98	79.96±3.81	80.42±4.06	81.62±3.47	0.78296±0.02962
	BM-fuzzy kNN	64.97±3.20	64.70±3.27	64.54±3.18	64.46±3.19	64.97±3.20	0.09246±0.00657
	FPFS-AC	**83.01±2.83**	**82.93±2.87**	**82.95±2.87**	**82.83±2.86**	**83.01±2.83**	2.05273±0.06616
Banknote	kNN	99.83±0.20	99.81±0.22	99.85±0.18	99.83±0.20	99.83±0.20	0.17910±0.00157
	SVM	98.82±0.53	98.79±0.56	98.84±0.53	98.81±0.54	98.82±0.53	0.09380±0.01582
	Fuzzy kNN	99.86±0.11	99.86±0.12	99.87±0.10	99.86±0.11	99.86±0.11	0.01364±0.00054
	EigenClass	97.97±0.93	97.85±0.96	98.15±0.85	97.96±0.93	97.97±0.93	1.84259±0.04760
	BM-fuzzy kNN	98.86±0.75	98.79±0.79	98.93±0.70	98.85±0.76	98.86±0.75	0.09742±0.00191
	FPFS-AC	**99.92±0.28**	**99.89±0.29**	**99.94±0.26**	**99.92±0.28**	**99.92±0.28**	5.75217±0.07779
Breast tissue	kNN	89.08±2.98	68.15±9.70	66.42±9.03	71.25±8.04	67.23±8.93	0.01947±0.00224
	SVM	88.38±3.12	67.19±10.81	63.87±9.64	68.20±8.32	65.15±9.35	2.57618±0.63478
	Fuzzy kNN	84.09±3.21	55.01±12.19	50.85±8.93	57.62±8.23	52.28±9.62	0.00041±0.00027
	EigenClass	87.19±3.39	63.48±10.85	60.58±10.32	65.08±8.29	61.58±10.16	0.01417±0.00199
	BM-fuzzy kNN	86.48±3.15	60.77±10.43	58.24±9.78	62.70±8.11	59.43±9.44	0.01120±0.00429
	FPFS-AC	**90.17±2.79**	**71.26±9.29**	**69.93±8.46**	**73.57±8.26**	**70.50±8.37**	0.04505±0.00650
Cryotherapy	kNN	85.22±8.59	86.18±8.62	85.42±8.56	85.09±8.63	85.22±8.59	0.01655±0.00104
	SVM	87.33±6.15	88.60±5.61	87.53±6.10	87.15±6.34	87.33±6.15	0.13549±0.09379
	Fuzzy kNN	90.56±8.27	91.30±7.79	90.47±8.42	90.37±8.50	90.56±8.27	0.00032±0.00022
	EigenClass	86.11±9.67	87.98±9.36	85.62±9.80	85.65±9.96	86.11±9.67	0.00902±0.00088
	BM-fuzzy kNN	81.56±11.63	82.49±11.55	81.54±11.60	81.26±11.83	81.56±11.63	0.00598±0.00059
	FPFS-AC	**92.22±7.86**	**92.74±7.73**	**92.14±7.84**	**92.14±7.89**	**92.22±7.86**	0.02958±0.00240
Glass	kNN	89.90±1.67	71.98±9.96	60.87±7.83	72.71±6.92	69.71±5.00	0.03097±0.00064
	SVM	88.30±1.72	71.41±9.04	56.05±8.39	71.10±7.40	64.91±5.16	0.25523±0.00953
	Fuzzy kNN	89.06±1.83	56.88±10.36	55.84±8.71	67.25±6.88	63.27±6.54	0.00076±0.00024
	EigenClass	89.00±2.18	69.79±10.42	62.77±7.73	69.67±5.92	67.00±6.54	0.05293±0.00178
	BM-fuzzy kNN	90.99±2.01	72.33±8.61	68.43±9.52	72.02±7.08	70.96±6.02	0.01869±0.00166
	FPFS-AC	**91.14±2.13**	**72.95±9.07**	**68.61±9.08**	**72.98±7.18**	**71.43±6.38**	0.16225±0.00567

Table 2. (Continued).

Datasets	Classifiers	Acc±SD	Pre±SD	Rec±SD	MacF±SD	MicF±SD	Running Time±SD
Hayes-Roth	kNN	65.69±4.51	60.12±5.56	48.57±7.55	50.49±7.76	48.54±6.77	0.02179±0.00068
	SVM	73.84±4.86	66.50±6.88	62.07±7.61	62.87±7.11	60.75±7.29	0.12284±0.11463
	Fuzzy kNN	64.38±3.96	54.65±12.50	41.85±6.01	45.38±5.99	46.57±5.94	0.00049±0.00026
	EigenClass	69.98±5.87	58.98±13.78	49.31±8.32	55.38±9.59	54.96±8.81	0.01779±0.00078
	BM-fuzzy kNN	60.85±6.11	42.03±13.64	38.03±8.99	41.77±9.48	41.27±9.16	0.00755±0.00052
	FPFS-AC	**85.01±5.26**	**81.21±8.27**	**76.07±8.69**	**76.70±8.98**	**77.51±7.90**	0.05717±0.00185
Ionosphere	kNN	84.73±3.41	88.50±3.60	79.46±4.46	81.49±4.53	84.73±3.41	0.05778±0.00262
	SVM	86.90±3.65	88.83±3.95	83.06±4.56	84.71±4.43	86.90±3.65	0.02434±0.00151
	Fuzzy kNN	84.98±3.37	89.03±3.23	79.64±4.52	81.73±4.54	84.98±3.37	0.00339±0.00039
	EigenClass	81.85±4.51	80.86±4.27	82.56±4.17	81.04±4.48	81.85±4.51	0.28505±0.00955
	BM-fuzzy kNN	80.35±4.57	81.32±5.66	75.35±5.58	76.65±5.94	80.35±4.57	0.13591±0.01050
	FPFS-AC	**88.46±3.41**	**91.41±2.97**	**84.40±4.54**	**86.37±4.32**	**88.46±3.41**	1.07337±0.01555
Iris	kNN	96.49±2.73	95.21±3.82	94.73±4.10	94.73±4.09	94.73±4.10	0.02373±0.00119
	SVM	98.18±1.83	97.54±2.51	97.27±2.75	97.25±2.77	97.27±2.75	0.06465±0.00406
	Fuzzy kNN	97.42±2.03	96.43±2.89	96.13±3.04	96.12±3.05	96.13±3.04	0.00055±0.00029
	EigenClass	96.58±2.30	95.22±3.34	94.87±3.45	94.85±3.46	94.87±3.45	0.02151±0.00091
	BM-fuzzy kNN	97.20±2.10	96.24±2.81	95.80±3.15	95.77±3.18	95.80±3.15	0.00766±0.00077
	FPFS-AC	**98.33±2.15**	**97.65±3.02**	**97.30±3.23**	**97.98±3.25**	**97.50±3.23**	0.07129±0.01141
Mice	kNN	99.85±0.14	99.42±0.59	99.42±0.55	99.40±0.59	99.41±0.58	0.18449±0.00319
	SVM	**100.00±0.00**	**100.00±0.00**	**100.00±0.00**	**100.00±0.00**	**100.00±0.00**	1.73284±0.02842
	Fuzzy kNN	99.88±0.13	99.56±0.50	99.53±0.54	99.53±0.54	99.54±0.54	0.02297±0.00160
	EigenClass	**100.00±0.00**	99.99±0.06	99.99±0.07	99.99±0.06	99.99±0.07	21.76134±0.82722
	BM-fuzzy kNN	99.98±0.05	99.91±0.19	99.92±0.18	99.91±0.19	99.92±0.18	0.71680±0.01891
	FPFS-AC	**100.00±0.00**	**100.00±0.00**	**100.00±0.00**	**100.00±0.00**	**100.00±0.00**	31.74698±1.75115
Parkinsons[sic]	kNN	91.44±4.23	88.84±5.83	88.73±5.92	88.50±5.60	91.44±4.23	0.03339±0.00149
	SVM	86.56±4.19	86.29±7.07	76.09±7.02	78.94±7.20	86.56±4.19	0.82134±0.19197
	Fuzzy kNN	86.05±4.28	82.10±5.89	79.82±7.01	80.41±6.40	86.05±4.28	0.00092±0.00022
	EigenClass	90.26±4.80	87.53±6.65	86.89±6.73	86.85±6.43	90.26±4.80	0.06002±0.00564
	BM-fuzzy kNN	79.28±5.81	73.06±7.16	74.07±6.95	73.07±7.01	79.28±5.81	0.04205±0.00178
	FPFS-AC	**92.97±3.84**	**91.11±5.25**	**90.78±4.87**	**90.62±4.76**	**92.97±3.84**	0.22190±0.00615

Table 2. (Continued).

Datasets	Classifiers	Acc±SD	Pre±SD	Rec±SD	MacF±SD	MicF±SD	Running Time±SD
Parkinson's disease	kNN	88.27±2.23	87.92±3.19	80.02±4.02	82.75±3.71	88.27±2.23	0.38537±0.03081
	SVM	74.60±0.29	74.60±0.29	50.00±0.00	85.45±0.19	74.60±0.29	0.03864±0.00430
	Fuzzy kNN	71.08±2.80	61.25±3.65	60.57±3.49	60.76±3.55	71.08±2.80	0.17090±0.01556
	EigenClass	82.02±2.24	80.31±4.99	68.82±3.80	71.37±4.16	82.02±2.24	1113.09220±63.87452
	BM-fuzzy kNN	50.77±3.30	50.90±2.99	51.19±3.93	47.60±3.20	50.77±3.30	5.73097±0.18794
	FPFS-AC	**93.47±2.11**	**91.74±2.99**	**91.13±3.00**	**91.33±2.75**	**93.47±2.11**	2751.71536±394.94782
Image segmentation	kNN	98.57±0.22	95.12±0.76	95.00±0.77	94.95±0.79	95.00±0.77	0.35440±0.01137
	SVM	98.95±0.23	96.42±0.77	96.34±0.80	96.35±0.80	96.34±0.80	8.34823±0.94517
	Fuzzy kNN	98.29±0.25	94.29±0.79	94.03±0.88	94.02±0.89	94.03±0.88	0.05078±0.00331
	EigenClass	98.54±0.28	94.99±0.98	94.88±0.98	94.86±0.98	94.88±0.98	7.66667±0.11306
	BM-fuzzy kNN	95.62±0.41	84.98±1.38	84.66±1.43	84.63±1.44	84.66±1.43	0.50286±0.01351
	FPFS-AC	**99.77±0.24**	**96.73±0.82**	**96.69±0.83**	**96.66±0.84**	**96.69±0.83**	28.80720±0.52446
Sonar	kNN	84.54±5.60	85.80±5.49	84.04±5.74	84.16±5.85	84.54±5.60	0.03587±0.00127
	SVM	78.03±5.56	79.09±5.79	77.51±5.59	77.52±5.69	78.03±5.56	0.01522±0.00041
	Fuzzy kNN	82.31±4.67	83.10±4.72	81.89±4.75	81.99±4.81	82.31±4.67	0.00193±0.00030
	EigenClass	81.16±5.57	82.06±5.53	81.64±5.53	81.12±5.59	81.16±5.57	0.47686±0.01037
	BM-fuzzy kNN	82.45±4.88	82.95±4.98	82.38±4.91	82.29±4.93	82.45±4.88	0.10877±0.00431
	FPFS-AC	**85.34±5.64**	**86.67±5.47**	**84.76±5.86**	**84.94±5.92**	**85.34±5.64**	0.75348±0.01256
Teaching	kNN	65.69±4.22	48.78±6.81	48.52±6.36	47.57±6.52	48.54±6.33	0.02408±0.00159
	SVM	68.57±5.23	54.51±7.89	53.02±7.83	51.73±8.38	52.85±7.85	0.14484±0.04950
	Fuzzy kNN	72.26±5.76	60.25±9.15	58.36±8.67	57.68±8.97	58.39±8.64	0.00049±0.00022
	EigenClass	70.45±5.52	56.86±8.56	55.67±8.24	55.12±8.35	55.68±8.28	0.02349±0.00203
	BM-fuzzy kNN	61.13±5.76	41.82±9.54	41.70±8.70	41.27±8.76	41.69±8.63	0.01177±0.00412
	FPFS-AC	**75.78±5.22**	**64.91±7.95**	**63.57±7.70**	**62.90±8.16**	**63.67±7.83**	0.07271±0.00131
Vowel	kNN	99.17±0.25	95.68±1.33	95.42±1.38	95.39±1.40	95.42±1.38	0.15490±0.00363
	SVM	96.37±0.41	81.10±2.20	80.02±2.27	79.91±2.29	80.02±2.27	2.98539±0.04392
	Fuzzy kNN	99.21±0.30	95.89±1.56	95.67±1.64	95.62±1.67	95.67±1.64	0.01271±0.00283
	EigenClass	96.46±0.56	83.15±2.69	80.52±3.10	80.58±3.01	80.52±3.10	1.19880±0.01669
	BM-fuzzy kNN	85.33±0.50	18.59±3.72	19.30±2.77	22.43±3.28	19.30±2.77	0.16834±0.00641
	FPFS-AC	**99.75±0.14**	**98.73±0.67**	**98.64±0.74**	**98.63±0.75**	**98.64±0.74**	4.19317±0.07033

Table 2. (Continued).

Datasets	Classifiers	Acc±SD	Pre±SD	Rec±SD	MacF±SD	MicF±SD	Running Time±SD
Mean	kNN	88.05±2.91	83.59±4.54	80.56±4.61	82.02±4.49	82.33±4.05	0.10908±0.00856
	SVM	86.75±3.20	82.02±4.93	77.13±4.95	80.96±4.90	80.39±4.41	1.44828±0.16618
	Fuzzy kNN	85.79±2.94	79.12±5.24	76.73±4.67	78.31±4.50	79.21±4.23	0.01910±0.00184
	EigenClass	87.28±3.42	81.43±5.69	78.82±5.13	80.00±5.02	80.70±4.84	76.48703±4.32951
	BM-fuzzy kNN	81.05±3.61	70.06±5.78	68.94±5.42	69.64±5.23	70.09±4.99	0.51056±0.01759
	FPFS-AC	**91.69±2.93**	**88.00±4.45**	**86.46±4.53**	**87.17±4.41**	**87.42±4.08**	188.45029±26.50007

Acc, Pre, Rec, MacF, and MicF results and their standard deviations (SD) are presented in percentage. Running time and its SD are presented in seconds.
The best performance is shown in bold.

We summarize Table 2 by ranking the number of the best performance results for every classifier to ease interpreting the results therein. Afterwards, we provide the ranking results in Table 3 and 4. Table 3 and Table 4 include ranking numbers of the best results and a pairwise comparison of the ranking results, respectively. Table 3 corroborates that FPFS-AC outperforms the other state-of-the-arts classifiers for 15 datasets. In addition, Table 3 manifests that FPFS-AC has the highest classification results of 75 for all the performance metrics. In contrast, kNN, SVM, fuzzy kNN, EigenClass, and BM-fuzzy kNN have the same classification results with FPFS-AC in the number of 0, 5, 0, 1, and 0 performance metrics according to only one dataset, respectively.

Table 3. Ranking number of the best results for all kNN-based classifier compared among each other.

Classifiers	**Acc**	**Pre**	**Rec**	**MacF**	**MicF**	**Total Rank**
kNN	0/15	0/15	0/15	0/15	0/15	0/75
SVM	1/15	1/15	1/15	1/15	1/15	5/75
Fuzzy kNN	0/15	0/15	0/15	0/15	0/15	0/75
EigenClass	1/15	0/15	0/15	0/15	0/15	1/75
BM-fuzzy kNN	0/15	0/15	0/15	0/15	0/15	0/75
FPFS-AC	15/15	15/15	15/15	15/15	15/15	75/75

Table 4. Ranking number of the best results for two kNN-based classifier compared versus each other

Classifiers	**Acc**	**Pre**	**Rec**	**MacF**	**MicF**
FPFS-AC versus kNN	15	15	15	15	15
FPFS-AC versus SVM	15	15	15	15	15
FPFS-AC versus fuzzy kNN	15	15	15	15	15
FPFS-AC versus EigenClass	15	15	15	15	15
FPFS-AC versus BM-fuzzy kNN	15	15	15	15	15

4.3. Statistical analyses of the simulation results

In this subsection, we employ the corrected Friedman test [52] and the Nemenyi post-hoc test [53] in a manner recommended by Demšar [56] to evaluate whether the overall differences in the performance results obtained in view of five performance metrics and running time are statistically significant. The Friedman test, a nonparametric test for multiple hypotheses testing, produces a performance-based ranking of the algorithms for each data set. Thereby, the rank of 1 refers to the best performing algorithm, the rank of 2 to the second best, etc. It assigns average ranks in the event that the ranks of the algorithms are equal.

Afterwards, the Friedman test first compares the average ranks of the algorithms and secondly calculates the Friedman statistic χ_F^2, distributed according to the χ_F^2 distribution with $k-1$ degrees of freedom. Here, k is the number of algorithms. If a statistically significant difference is detected in the performance, a post-hoc test should be used to detect which difference belong to which algorithm. The Nemenyi test is one of the post-hoc tests commonly used to compare all the classifiers with each other. In this test, if the average ranks of the two algorithms occur more than the critical distance, then the test shows that their performance is considerably different.

We first calculate the average rank of each algorithm considered in our experiments with $k = 6$ and $N = 15$ since the total number of the methods is 6 and the total number of the datasets is 15. If the accuracy, precision, recall, macro F-score, micro F-score, and running time values of the Friedman test statistic are $\chi_F^2 = 37.61$, $\chi_F^2 = 39.16$, $\chi_F^2 = 37.04$, $\chi_F^2 = 39.85$, $\chi_F^2 = 38.55$ and $\chi_F^2 = 51.84$, respectively, with 5 $(k-1)$ degrees of freedom and the critical value for the Friedman test [52] given for $k = 6$ and $N = 15$ is 11.07 at a significance level of $\alpha = 0.05$, we can conclude that the accuracy $(37.61 > 11.07)$, precision $(39.16 > 11.07)$, recall $(37.04 > 11.07)$, macro F-score $(39.85 > 11.07)$, micro F-score $(38.55 > 11.07)$, and running time $(51.84 > 11.07)$ values of the studied methods are significantly different. Now that the null hypothesis is rejected, we can proceed with a post-hoc test. The Nemenyi test [53] can be used when all classifiers are compared with each other [56].

The critical value in our experiments with $k = 6$ and $\alpha = 0.05$ is 1.9469. As a result, the accuracy, precision, recall, macro F-score, and micro F-score of the proposed FPFS-AC method is significantly different from fuzzy kNN, FSSC, FussCyier, HDFSSC, and BM-fuzzy kNN methods while its running time results are not significantly different from those of fuzzy kNN. Figure 1 presents the critical diagrams generated by the Nemenyi post-hoc test for the five evaluation measures and running time.

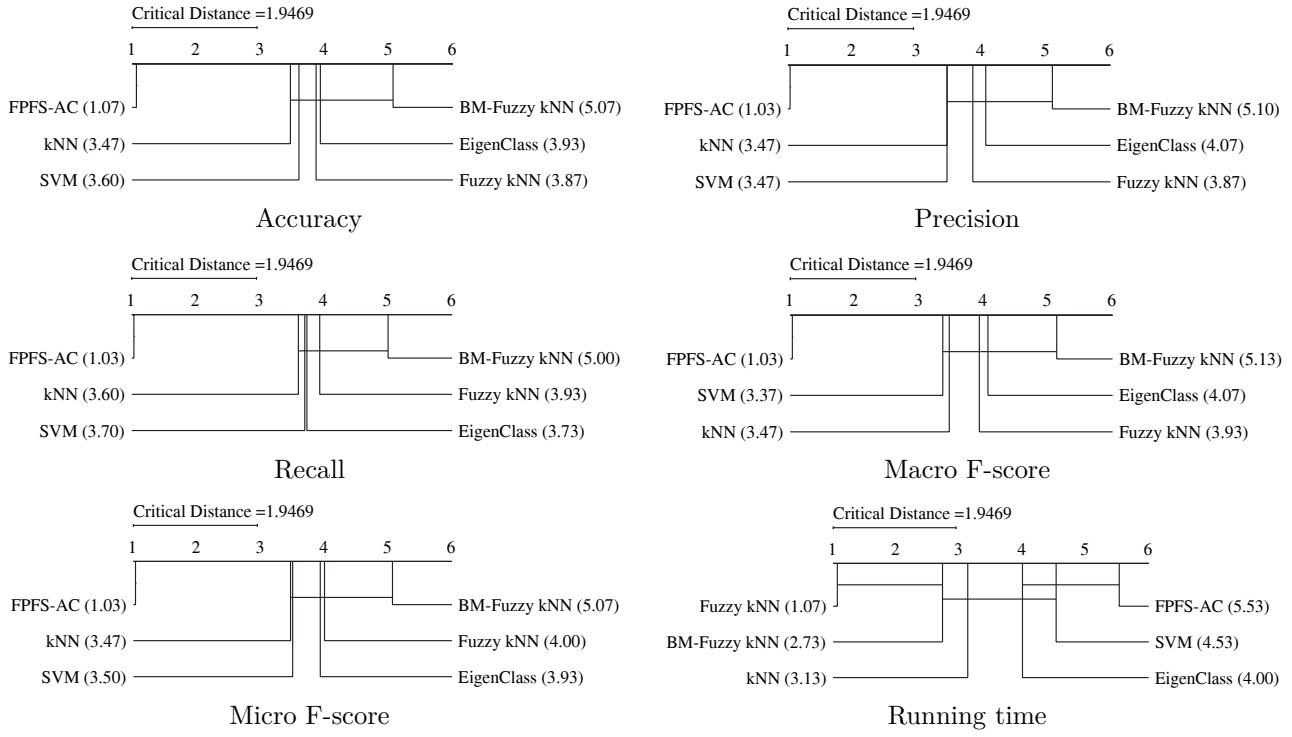

Figure 1. The critical diagrams for the five evaluation measures and running time: The results from the Nemenyi post-hoc test at 0.05 significance level and average rank scores from Friedman Test.

Figure 1 shows that the average ranks of FPFS-AC and the others were calculated to be more than a critical distance of 1.9469 but not in terms of running time results. Besides, Table 5 offers the pairwise comparison between the classifiers obtained via the critical distances in the Friedman test. Figure 1 and Table 5 manifest that FPFS-AC remarkably outperforms the others in terms of five performance measures.

Table 5. Pairwise performance comparison of the classifiers via Friedman test.

	kNN	SVM	Fuzzy kNN	EigenClass	BM-fuzzy kNN	FPFS-AC
kNN	–	–	–	–	–	+
SVM	–	–	–	–	–	+
Fuzzy kNN	–	–	–	–	–	+
EigenClass	–	–	–	–	–	+
BM-fuzzy kNN	–	–	–	–	–	+
FPFS-AC	+	+	+	+	+	–

Here, the symbol – represents compared classifiers' performances are not significantly different, whereas + stands for they are.

4.4. Computational complexity analysis of FPFS-AC

This section provides a comparison of FPFS-AC's computational complexity with those of the classifiers by utilising the big O notation besides their running time results obtained in 10 runs featuring the 15 UCI datasets in Table 2. As clear from Table 2, FPFS-AC generally seems to operate faster than SVM and slightly slower than kNN, fuzzy kNN, EigenClass, and BM-fuzzy kNN, which results from its processing all the training instances by exploiting CCE10 to predict the class label of the considered test instances. On the other hand, longer processing time in the specific datasets primarily stems from the MATLAB operation. To elaborate, parallel computing for multi-parameter eig(A, B) is not allowed by MATLAB. In future works, the running time may be remarkably decreased if the parallel computing problem is overcome for multiparameters. Despite this problem, FPFS-AC's running time occurs under 1 s for eight of 15 datasets. From the pseudocode of FPFS-AC, the computational complexity is $O(mn)$ since mn is higher than $m5$ for each test sample. Here, m and n are the number of the training samples and of their attributes, respectively. The computational complexities of the compared classifiers are provided in Table 6.

Table 6. Computational complexities of the classifiers.

Classifier	**Computational complexity**
kNN	$O(n \log k)$
SVM with kernel	$O(m^3)$
Fuzzy kNN	$O(n^2 \log k)$
EigenClass	$O(mn)$
BM-fuzzy kNN	$O(ln^3 \log k)$
FPFS-AC	$O(mn)$

k is number of nearest neighbour, m is the sample number of the training data, n is the parameter number of the training data, and l is the class number of the data.

5. Conclusion

In this study, we developed an efficient CCE10-based classification algorithm, namely FPFS-AC. In contrast to most of the available literature relying on fictitious problems, we applied the similarity measures of *fpfs*-matrices and an SDM method to a real-world problem (data classification). By doing so, we proposed FPFS-AC based

on multiple pseudo-similarities of *fpfs*-matrices, generalised eigenvalue-based quasi-similarity, and CCE10 for numerical data classification and compared FPFS-AC with kNN [3], SVM [4], fuzzy kNN [6], EigenClass [7], and BM-fuzzy kNN [8]. The results manifested that FPFS-AC outperformed the other well-known and state-of-the-art methods and SDM method using *fpfs*-matrices was efficacious in data classification. This paper is believed to inspire new research on how to apply the SDM methods based on *fpfs*-matrices to real-world problems, such as data classification.

FPFS-AC has various advantages, from classification performance to its developable algorithmic structure. The simulation results and statistical analyses prove that FPFS-AC achieves the highest classification results. Besides these performances, it can produce the highest classification performance in a great variety of datasets or problems. This issue is the most significant advantage of FPFS-AC. Furthermore, it has ease of development. For instance, utilising different similarity measures and SDM methods is possible without any complex procedure. Thus, FPFS-AC can be easily improved for specific datasets or problems to exceed the previous classification performance. Since we focus on proposing an efficacious classifier for any considered datasets, we did not develop a classifier herein for a specific dataset. On the other hand, FPFS-AC's drawback is to be employed several classical operations, such as Pearson's correlation coefficient and standard deviation whose classification performances have some inherent limitations, for the weighting of the similarity measures. To deal with this drawback, some new mathematical or statistical tools can be utilised or defined.

The results in the present study demonstrated that *fpfs*-matrices and SDM methods relying on these matrices had notable modelling abilities exploitable in data classification. Therefore, further research should be focused on SDM methods constructed by *fpfs*-matrices and their implementations in machine learning. Furthermore, it is possible to improve the proposed FPFS-AC, for example, by employing different SDM methods [10–13, 17–21, 44–47] and the similarity measures of *fpfs*-matrices. Researchers can also define similarity measures of the intuitionistic fuzzy parameterized intuitionistic fuzzy soft matrices [16] or interval-valued intuitionistic fuzzy parameterized interval-valued intuitionistic fuzzy soft sets/matrices [27]. In addition, new mathematical tools, such as picture fuzzy sets [59, 60] and picture fuzzy soft sets [59], can be utilised. One can also insert a preprocessing step in the training phase of FPFS-AC to decrease the negative effects of the unstable training instances in the considered datasets on classification success.

Acknowledgement

This research study was granted 2211-C Domestic Doctoral Fellowship for Priority Areas by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under Grant 1649B031905299.

Author Contributions

Samet Memiş produced the main conceptual ideas and developed the theoretical framework. Uğur Erkan carried out the simulations and statistical analyses. Serdar Enginoğlu supervised the findings of this work. Samet Memiş and Serdar Enginoğlu wrote the manuscript by consulting with Uğur Erkan. All the authors discussed the results and contributed to the final manuscript.

References

[1] Mehryar M, Rostamizadeh A, Talwalkar A. Foundations of Machine Learning. In: Bach F (editor). Introduction: Learning Scenarios, 2nd ed. London, England; The MIT Press, 2018. p. 6.

[2] Fix E, Hodges JL. Discriminatory analysis, nonparametric discrimination: Consistency properties. Texas, USA: USAF School of Aviation Medicine, Randolph Field, 1951.

[3] Cover TM, Hart PE. Nearest neighbor pattern classification. IEEE Transactions on Information Theory 1967; 13: 21-27. doi: 10.1109/TIT.1967.1053964

[4] Cortes C, Vapnik V. Support-vector networks. Machine learning 1995; 20 (3): 273-297. doi: 10.1007/BF00994018

[5] Zadeh LA. Fuzzy sets. Information and Control 1965; 8: 338-353. doi: 10.1016/S0019-9958(65)90241-X

[6] Keller JM, Gray MR, Givens JA. A fuzzy k-nearest neighbor algorithm. IEEE Transactions on Systems, Man, and Cybernetics 1985; 15: 580-585. doi: 10.1109/TSMC.1985.6313426.

[7] Erkan U. A precise and stable machine learning algorithm: Eigenvalue classification (EigenClass). Neural Computing and Applications 2021; 33 (10): 5381-5392. doi: 10.1007/s00521-020-05343-2

[8] Kumbure MM, Luukka P, Collan M. A new fuzzy k-nearest neighbor classifier based on the Bonferroni mean. Pattern Recognition Letters 2020; 140: 172-178. doi: 10.1016/j.patrec.2020.10.005

[9] Molodtsov D. Soft set theory-first results. Computers and Mathematics with Applications 1999; 37 (4-5): 19-31.

[10] Enginoğlu S, Memiş S. A configuration of some soft decision-making algorithms via fpfs-matrices. Cumhuriyet Science Journal 2018; 39 (4): 871-881. doi: 10.17776/csj.409915

[11] Enginoğlu S, Memiş S. Comment on fuzzy soft sets [The Journal of Fuzzy Mathematics 9(3), 2001, 589-602]. International Journal of Latest Engineering Research and Applications 2018; 3 (9): 1-9.

[12] Enginoğlu S, Memiş S. A review on an application of fuzzy soft set in multicriteria decision making problem [P. K. Das, R. Borgohain, International Journal of Computer Applications 38 (2012) 33-37]. In: International Conference on Mathematical Studies and Applications 2018; Karaman, Turkey; 2018, pp. 173-178.

[13] Enginoğlu S, Memiş S. A review on some soft decision-making methods. In: International Conference on Mathematical Studies and Applications 2018; Karaman, Turkey; 2018, pp. 437-442.

[14] Enginoğlu S, Ay M, Çağman N, Tolun V. Classification of the monolithic columns produced in troad and mysia region ancient granite quarries in northwestern anatolia via soft decision-making. Bilge International Journal of Science and Technology Research 2019; 3(Special Issue): 21-34. doi: 10.30516/bilgesci.646126

[15] Enginoğlu and Öngel T. Configurations of several soft decision-making methods to operate in fuzzy parameterized fuzzy soft matrices space. Eskişehir Technical University Journal of Science and Technology A-Applied Sciences and Engineering 2020; 21 (1): 58-71. doi: 10.18038/estubtda.562578

[16] Aydın T, Enginoğlu S. Configurations of SDM methods proposed between 1999 and 2012: A follow-up study. In: 4th International Conference on Mathematics "An İstanbul Meeting for World Mathematicians"; Istanbul, Turkey; 2020, pp. 192-211.

[17] Enginoğlu S, Aydın T, Memiş S, Arslan B. Operability-oriented configurations of the soft decision-making methods proposed between 2013 and 2016 and their comparisons. Journal of New Theory 2021; 34: 82-114.

[18] Enginoğlu S, Memiş S, Öngel T. A fast and simple soft decision-making algorithm: EMO18o. In: International Conference on Mathematical Studies and Applications 2018; Karaman, Turkey; 2018, pp. 179-186.

[19] Enginoğlu S, Memiş S, Arslan B. A fast and simple soft decision-making algorithm: EMA18an. In: International Conference on Mathematical Studies and Applications 2018; Karaman, Turkey; 2018, pp. 428-436.

[20] Enginoğlu S, Memiş S, Öngel T. Comment on soft set theory and uni-int decision-making [European Journal of Operational Research, (2010) 207, 848-855]. Journal of New Results in Science 2018; 7 (3): 28-43.

[21] Enginoğlu S, Memiş S, Arslan B. Comment (2) on soft set theory and uni-int decision-making [European Journal of Operational Research, (2010) 207, 848-855]. Journal of New Theory 2018; 25: 84-102.

[22] Sezgin A, Çağman N, Çıtak F. α-inclusions applied to group theory via soft set and logic. Communications Faculty of Sciences University of Ankara Series A1 Mathematics and Statistics 2019; 68 (1): 334-352.

[23] Özlü S, Sezgin A. Soft covered ideals in semigroups. Acta Universitatis Sapientiae, Mathematica 2020; 12 (2): 317-346. doi:10.2478/ausm-2020-0023

[24] Şenel G, Lee JG, Hur K. Advanced soft relation and soft mapping. International Journal of Computational Intelligence Systems 2021; 14 (1): 461-470. doi: 10.2991/ijcis.d.201221.001

[25] Enginoğlu S, Çağman N, Karataş S, Aydın T. On soft topology, El-Cezerî Journal of Science and Engineering 2015; 2 (3): 23-38. doi: 10.31202/ecjse.67135

[26] Riaz M, Hashm R, Farooq A. Fuzzy parameterized fuzzy soft metric spaces. Journal of Mathematical Analysis 2018; 9: 25-36.

[27] Aydın T, Enginoğlu S. Some results on soft topological notions. Journal of New Results in Science 2021; 10 (1): 65-75.

[28] Molodtsov DA. The Theory of Soft Sets. Moscow, Russia: URSS Publishers, 2004 (in Russian).

[29] Çağman N, Enginoğlu S. Soft set theory and uni-int decision making. European Journal of Operational Research 2010; 207: 848-855. doi: 10.1016/j.ejor.2010.05.004.

[30] Karaaslan F, Deli İ. Soft neutrosophic classical sets and their applications in decision-making. Palestine Journal of Mathematics 2020; 9 (1): 312-326.

[31] Garg H, Arora R. Algorithms based on COPRAS and aggregation operators with new information measures for possibility intuitionistic fuzzy soft decision-making. Mathematical Problems in Engineering 2020; 2020: Article ID 1563768, 1-20. doi: 10.1155/2020/1563768

[32] Maji PK, Biswas R, Roy AR. Fuzzy soft sets. The Journal of Fuzzy Mathematics 2001; 9 (3): 589-602.

[33] Çağman N, Enginoğlu S, Çıtak F. Fuzzy soft set theory and its applications. Iranian Journal of Fuzzy Systems 2011; 8: 137-147.

[34] Çağman N, Çıtak F, Enginoğlu S. FP-soft set theory and its applications. Annals of Fuzzy Mathematics and Informatics 2011; 2: 219-226.

[35] Çağman N, Çıtak F, Enginoğlu S. Fuzzy parameterized fuzzy soft set theory and its applications. Turkish Journal of Fuzzy Systems 2010; 1 (1): 21-35.

[36] Çağman N, Enginoğlu S. Soft matrix theory and its decision making. Computers and Mathematics with Applications 2010; 59: 3308-3314. doi: 10.1016/j.camwa.2010.03.015.

[37] Çağman N, Enginoğlu S. Fuzzy soft matrix theory and its application in decision making. Iranian Journal of Fuzzy Systems 2012; 9: 109-119.

[38] Enginoğlu S, Çağman N. Fuzzy parameterized fuzzy soft matrices and their application in decision-making. TWMS Journal of Applied and Engineering Mathematics 2020; 10: 1105-1115.

[39] Khameneh AZ, Kılıçman A. Multi-attribute decision-making based on soft set theory: A systematic review. Soft Computing 2019; 23 (16): 6899-6920. doi: 10.1007/s00500-018-3330-7

[40] Mushrif MM, Senqupta S, Ray AK. Texture classification using a novel, soft-set theory based classification algorithm. In: 7th Asian Conference on Computer Vision; Hyderabad, India; 2006, pp. 246-254. doi: 10.1007/11612032_26

[41] Handaga B, Onn H, Herawan T. FSSC: An algorithm for classifying numerical data using fuzzy soft set theory. International Journal of Fuzzy System Applications 2012; 3 (4): 29-46. doi: 10.4018/ijfsa.2012100102

[42] Lashari SA, Ibrahim R, Senan N. Medical data classification using similarity measure of fuzzy soft set based distance measure. Journal of Telecommunication, Electronic and Computer Engineering 2017; 9 (2-9): 95-99.

[43] Yanto ITR, Seadudin RR, Lashari SA, Haviluddin. A numerical classification technique based on fuzzy soft set using hamming distance. In: Third International Conference on Soft Computing and Data Mining; Johor, Malaysia; 2018, pp. 252-260. doi: 10.1007/978-3-319-72550-5_25

[44] Enginoğlu S, Memiş S, Çağman N. A generalisation of fuzzy soft max-min decision-making method and its application to a performance-based value assignment in image denoising. El-Cezerî Journal of Science and Engineering 2018; 6 (3): 466-481. doi: 10.31202/ecjse.551487

[45] Enginoğlu S, Memiş S, Karaaslan F. A new approach to group decision-making method based on TOPSIS under fuzzy soft environment. Journal of New Results in Science 2019; 8 (2): 42-52.

[46] Enginoğlu S, Memiş S. A new approach to the criteria-weighted fuzzy soft max-min decision-making method and its application to a performance-based value assignment problem. Journal of New Results in Science 2020; 9 (1): 19-36.

[47] Enginoğlu S, Aydın, T., Memiş S. Arslan, B. SDM methods' configurations (2017-2019) and their application to a performance-based value assignment problem: A follow up study. Annals of Optimization Theory and Practice 2021; 4 (1): 41-85. doi: 10.22121/AOTP.2021.287404.1069

[48] Memiş S, Enginoğlu S, Erkan U. Numerical data classification via distance-based similarity measures of fuzzy parameterized fuzzy soft matrices. IEEE Access 2021; 9: 88583-88601. doi: 10.1109/ACCESS.2021.3089849

[49] Memiş S, Enginoğlu S. An application of fuzzy parameterized fuzzy soft matrices in data classification. In: International Conferences on Science and Technology; Natural Science and Technology ICONST-NST 2019; Prizren, Kosovo; 2019. pp. 68-77.

[50] Memiş S, Enginoğlu S, Erkan U. A data classification method in machine learning based on normalised hamming pseudo-similarity of fuzzy parameterized fuzzy soft matrices. Bilge International Journal of Science and Technology Research 2019; 3 (Special Issue): 1-8. doi: 10.30516/bilgesci.643821

[51] Dua D, Graff C. UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences 2019.

[52] Friedman M. A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics 1940; 11 (1): 86-92.

[53] Nemenyi PB. Distribution-free multiple comparisons. PhD, Princeton University, Princeton, New Jersey, USA, 1963.

[54] Stone M. Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society. Series B (Methodological) 1974; 36: 111-147. doi: 10.1111/j.2517-6161.1974.tb00994.x

[55] Xiong Z. Cui Y. Liu Z. Zhao Y. Hu M. Hu J. Evaluating explorative prediction power of machine learning algorithms for materials discovery using k-fold forward cross-validation. Computational Materials Science 2020; 171: 109203. doi: 10.1016/j.commatsci.2019.109203

[56] Demšar J. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 2006; 7: 1-30.

[57] Enginoğlu S, Arslan B. Intuitionistic fuzzy parameterized intuitionistic fuzzy soft matrices and their application in decision-making. Computational and Applied Mathematics 2020; 39: Article Number: 325. doi: 10.1007%2Fs40314-020-01325-1

[58] Aydın T, Enginoğlu S. Interval-valued intuitionistic fuzzy parameterized interval-valued intuitionistic fuzzy soft sets and their application in decision-making. Journal of Ambient Intelligence and Humanized Computing 2021; 12 (1): 1541-1558. doi: 10.1007/s12652-020-02227-0

[59] Cuong BC. Picture fuzzy sets. Journal of Computer Science and Cybernetics 2014; 30 (4): 409-420. doi: 10.15625/1813-9663/30/4/5032

[60] Memiş S. A study on picture fuzzy sets. In: 7th IFS and Contemporary Mathematics Conference; Mersin, Turkey; 2021. pp. 125-132.