Turkish Journal of Electrical Engineering & Computer Sciences

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

Research Article

Turk J Elec Eng & Comp Sci
(2023) 31: 751 – 770

doi:10.55730/1300-0632.4016

Stepwise dynamic nearest neighbor (SDNN): a new algorithm for classification

Deniz KARABAŞ[1], **Derya BİRANT**[2,*], **Pelin YILDIRIM TAŞER**[3]
[1]The Graduate School of Natural and Applied Sciences, Dokuz Eylül University, İzmir, Turkiye
[2]Department of Computer Engineering, Faculty of Engineering, Dokuz Eylül University, İzmir, Turkiye
[3]Independent Researcher, İstanbul, Turkiye

Received: 20.03.2023 • **Accepted/Published Online:** 11.08.2023 • **Final Version:** 29.09.2023

Abstract: Although the standard k-nearest neighbor (KNN) algorithm has been used widely for classification in many different fields, it suffers from various limitations that abate its classification ability, such as being influenced by the distribution of instances, ignoring distances between the test instance and its neighbors during classification, and building a single/weak learner. This paper proposes a novel algorithm, called stepwise dynamic nearest neighbor (SDNN), which can effectively handle these problems. Instead of using a fixed parameter k like KNN, it uses a dynamic neighborhood strategy according to the data distribution and implements a new voting mechanism, called stepwise voting. Experimental results were conducted on 50 benchmark datasets. The results showed that the proposed SDNN method outperformed the KNN method, KNN variants, and the state-of-the-art methods in terms of accuracy.

Key words: Machine learning, classification, k-nearest neighbor, majority voting, ensemble learning

1. Introduction

The k-nearest neighbor (KNN) method is one of the popular machine learning techniques due to its intriguing characteristics, including easy implementation and high generalization ability [1]. It is a nonparametric lazy learner who predicts a sample's class label based on the majority vote of its k nearest neighbors who also have that class label. The KNN algorithm uses particular distance metrics, such as Euclidean, Manhattan, and Minkowski measures, to identify the k closest neighbors of a sample data point. Even though it is simple, the KNN approach typically matches or surpasses more complex and sophisticated methods in terms of generalization ability. It was generally utilized for classification [2–4], regression [5], and clustering [6] tasks in a variety of research areas.

Although KNN is effective in many situations, it has some shortcomings that abate its classification ability. Using a fixed k parameter is a considerable limitation since it ignores the distribution of nearest neighbors of a query instance, and the data distribution affects the neighborhood concept, so all of these affect the accuracy. Another shortcoming of the classical KNN algorithm is that it builds a single/weak learner, and it does not take advantage of the strengths of ensemble learning (EL) in classifying data instances. Even though EL is performed, a problem related to majority voting is that it ignores the fact that some learners that lie in the minority sometimes produce more accurate results since it does not explicitly address diversity [7].

*Correspondence: derya@cs.deu.edu.tr

The proposed method in this paper decreases these limitations by taking into account dynamic parameter k according to the data distribution and implementing a novel voting mechanism.

The novelty and main contributions of this study are highlighted as follows. (i) It proposes a novel algorithm, called stepwise dynamic nearest neighbor (SDNN), that applies a dynamic adjustment for the neighborhood number k for each instance in each iteration by considering the distribution. (ii) It proposes a novel voting mechanism, called stepwise voting, that divides the classifiers' outputs into some sequential groups and aggregates the votes in a step-by-step process to obtain the final output. (iii) It improves the KNN method and its variants in terms of recall, precision, F-measure, and accuracy measures. (iv) The proposed method outperformed the state-of-the-art methods with 5.4% improvement on average.

The remainder of the paper is organized as follows. Section 2 briefly explains the related literature. After that, Section 3 introduces the proposed SDNN method and the stepwise voting approach. Section 4 presents the experimental studies and the results obtained by the proposed method. The paper is concluded by Section 5. The last section also gives possible future plans.

2. Related work

Since the KNN method is straightforward, efficient, and easy to use, it has been widely studied by many researchers in health [8–10], education [11], finance [12], biomedicine [13], and text mining [14]. For example, Okediran et al. [8] applied the KNN algorithm for detecting and reporting COVID-19 symptoms in patients. The researchers tested this approach, and experimental results showed that the KNN algorithm provided good accuracy rate results on training and test data, respectively. In another study [9], KNN was used to build a machine learning model for classifying the sleep apnea types. Here, it is reported that KNN achieved higher classification accuracy, specificity, and sensitivity than its counterparts, including support vector machines, multilayer perceptron, and C4.5 decision tree. While Dilmaç et al. [10] applied KNN for electrocardiogram (ECG) heartbeat classification, Tang et al. [11] used it to predict the green consumption behaviors of college students. In another study [12], the authors implemented the KNN algorithm to predict a customer's loan repayment capability behavior.

The KNN algorithm has been adapted to numerous modifications to decrease its limitations/challenges and increase its accuracy/applicability, so there are different KNN variants or forms. For example, several different approaches have been proposed in the literature for adding weight to instances, such as uniform KNN [15], weighted KNN [16], and KNN learning with graph neural networks (KNNGNN) [17]. Wolff et al. [15] implemented uniform KNN regression for photovoltaic power predictions based on numerical weather measurements. Researchers in [16] focused on improving the classification accuracy of 1-nearest neighbor (1NN), relying on the majority-voting KNN and distance-weighted KNN with four common elastic distance measures. In a study [17], the KNNGNN approach, which generates the k-nearest neighbor rules in the architecture of a graph neural network including the weighting and distance functions embedded, was proposed for allowing more robust KNN learning.

In the literature, the dynamic determination of k has been adopted to the classical KNN algorithm in different ways. Wu et al. [18] proposed the dynamic k-nearest-neighbor with distance and attribute weighted (DKNDAW) method that contains two main parts. In the first part, the best k value was learned. After that, the target class label was calculated using the k nearest neighbors for each given test instance. Zhong et al. [19] proposed an improved KNN algorithm, called dynamic k-nearest neighbor (Dk-NN). In replace of fixed k value, they designed k as dynamic value as follows. First, a preprocessing step was designed and added to the classical KNN algorithm for determining a dynamic k interval. Afterward, each class's percentage of a given test instance was calculated within the dynamic k interval. Finally, three criteria were analyzed to predict the class label according to the variation tendency of the percentage curves.

Our method differs from the previous work in several points. First, it proposes a novel approach, SDNN, that considers the distribution of neighbors for each instance using a distance function. Second, it applies a dynamic adjustment for the neighborhood number k for each instance. Third, this study proposes a novel voting mechanism, called stepwise voting, that divides the outputs of the classifiers into a number of consecutive groups and combines the votes in a step-by-step process to produce the final output.

3. Proposed method

This paper proposes a novel algorithm, stepwise dynamic nearest neighbor (SDNN), that dynamically adjusts the neighborhood number k for each instance in each iteration by taking into account the distribution. In this method, multiple dynamic nearest neighbor (DNN) models that are trained with multiple training datasets created with the bootstrapping method are constructed. Considering accuracy, stability, and robustness, this approach outperforms traditional individual algorithms because they fully utilize the information provided by the learning members. The predictions by each DNN model are aggregated using a novel voting mechanism, stepwise voting, that divides each prediction into some sequential groups and combines them in a step-by-step process to obtain the final prediction.

Figure 1 shows the general overview of the proposed SDNN method. First, the original dataset is sampled using the bootstrap procedure, and multiple training sets are constructed. While the number of instances of the multiple training sets was kept equal to the number of instances of the original dataset through resampling with replacement, the number of attributes was reduced by randomly choosing some of them. For example, selecting 3/4 of the attributes can be large enough to consider different features and reasonably small to provide diversity. After that, the algorithm finds the nearest neighbors of a given query data on these multiple training sets, determines an appropriate k value for each one according to the data distribution, and so, multiple DNN models were generated. Finally, each model votes for a certain class label, the obtained outputs from each individual DNN model are aggregated, and the stepwise vote of these outputs is chosen as the final output value.

The traditional KNN algorithm simply performs a neighborhood search according to a fixed parameter k and ignores the distribution of data points. The k nearest data objects to the query instance are simply selected based on the Euclidean distance. These data objects are selected as neighbors without considering the data distribution and their distances from the query instance. However, the k-neighborhood of each data instance is generally related to the data distribution. Thus, the density of regions should have a straightforward meaning.

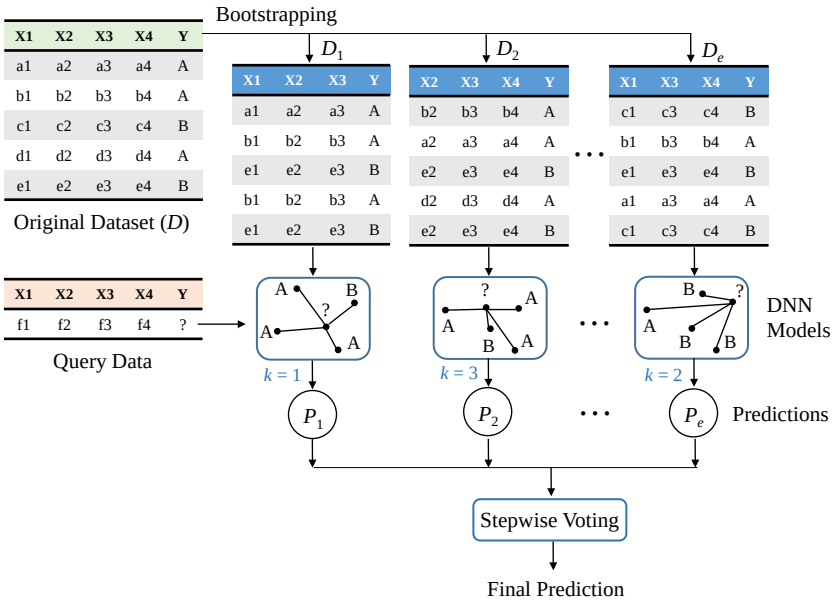

Figure 1. An overview of the proposed SDNN method.

Under nonuniform data distribution, its effect is often poor. The proposed SDNN overcomes this limitation by distinguishing the nearest and farthest neighbors using a dynamic parameter for each instance individually.

Figure 2 illustrates an example. When k is set to 5 for Query B, it searches for the five nearest neighbors and finds that two are of class 0, and three are of class 1. Then it uses the majority voting strategy to classify its class as 1. Here, three objects are selected as neighbors while their distances are very further than the other neighbors. It is because of the fixed parameter k and nonuniform data distribution. The outputs of 2-neighborhood and 5-neighborhood of Query B are different from each other. This difference is due to the effect of irregular data distribution. Thus, taking into consideration the variable number of neighbors and a neighborhood concept based on the data distribution can be useful to improve classification accuracy.

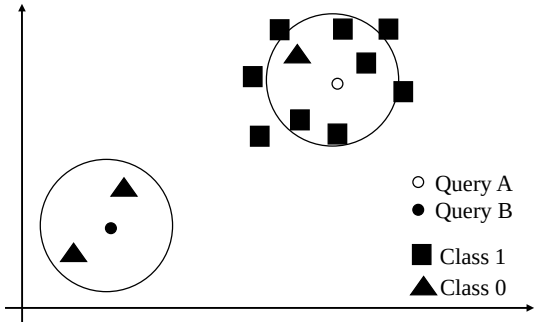

Figure 2. An example illustrating the SDNN algorithm.

The proposed SDNN method adopts a dynamic adjustment for the neighborhood number k in each iteration. It takes into consideration the distribution of neighbors for each instance using a distance function. It analyzes the distribution of local data points, tends to eliminate relatively far neighbors under a probability

constraint, and thereby improves the model's accuracy. While a larger k value will have a higher potential value for the data-intensive areas, a smaller k value will probably be selected for the data-sparse areas.

In the proposed SDNN method, the number of neighbors is not constant, and the selection of the actual neighbors depends on the data distribution. Here, a neighbor with a very further distance than the other neighbors may not be selected as a neighbor. Since the rejected neighbor is probably from a different category from the query instance, the classifier's performance is improved.

Majority voting (MV) is a simple and straightforward method for classification problems as it selects the class with the most votes. However, it has some drawbacks. A class label supported by many learners does not necessarily mean that it should be the true answer because learners' ability highly determines the quality of final output. This paper proposes a novel voting mechanism called stepwise voting.

In the proposed stepwise voting method, the truth discovery problem is formulated as an iterative procedure, which starts by grouping the answers, then combines the outputs of each group individually, and iterates by repeating the same process until a single output is reached. Therefore, an iterative procedure is performed by combining sequential group-based votes, which can result in assigning the correct answers and getting more correct over iterations. In other words, it can eliminate wrong answers in local decisions.

An example of stepwise voting is illustrated in Figure 3. In this example, the group size was set to 3, meaning that each DNN model's outputs will be grouped into three iteratively. For example, while the first group contains the outputs of the first three models, the second group includes the outputs of the second, third, and fourth models. The common answer in each group is selected as the winner. In other words, the winner of the group is the label that achieves the highest number of votes. The winner of each group gradually goes up to the next step. This local group-based voting process continues until one final output is reached. According to this example, the groups (illustrated with blue, orange, and green colors) are sequentially generated in the first iteration. After that, the outputs of classifiers compete in each group and the common answer is determined and lined up step by step. In the next step, the same grouping process is applied to these new outputs and then the new winner in each group is determined again. For example, the winner of the first group $[A\ B\ A]$ in the first step is selected as class label A and passed to the next step to be competed again with the outputs of other groups. Similarly, the second group $[B\ A\ B]$ in the first step produces the output B since the dominant class in this group is B. These procedures continue until only one output value remains. In this example, the class label B remains as the last value, so the final output of the ensembles of the DNNs is obtained as B. On the other hand, when we perform majority voting for the same example, the output is uncertain. Because the class label "A" has 5 votes and the class label "B" has 5 votes. The algorithm cannot decide in such a tie case.

3.1. Formal definition

Suppose a dataset D with n instances such that $D = \{(x_i, y_i)\}_{i=1}^{n}$ where $x_i \in X$ denotes input vector and $y_i \in Y = \{c_1, c_2, ..c_r\}$ is the class label associated with instance x_i and r is the number of class labels. Let $F = \{f_1, f_2, ..., f_m\}$ be the set of features, each instance is a point in the m-dimensional feature space. The term $X^j \in R^{d \times n_j}$ expresses the training instances from the j-th class, where n_j is the number of instances in the j-th category, and $\sum_{j=1}^{r} n_j = n$ The aim of SDNN is to learn a mapping function $f : R^m \rightarrow R^r$ such that distances between instance pairs belonging to the same class in the neighborhood are smaller than those

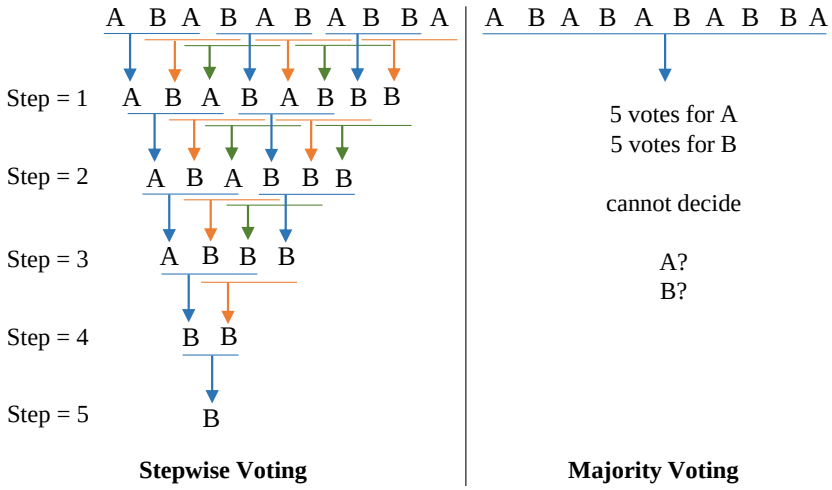

Figure 3. An example illustrating the difference between stepwise voting and majority voting.

between instance pairs belonging to the different classes. The function $d(x_i, x_j)$ is defined as the distance between points on training instance x_i and testing instance x_j. Euclidean distance is typically used for d. The algorithm finds a mapping that satisfies the following relation:

$$d(x_i, x_j) \ll d(x_i, x_l) \tag{1}$$

for all $1 \leq i, j, l \leq k$ such that $y_i = y_j$ and $y_i \neq y_l$.

Definition 1 (Local-distance) *For a given positive number k, the local-distance of instance x, which is denoted as* $L_k(x)$*, is described as the distance* $d\left(x, o^{'}\right)$ *between an object* $o^{'} \in D$ *and* x *such that:*

$$L_k(x) = d\left(x, o^{'}\right) \tag{2}$$

where $o' \in D$ and it holds $d(x, o) \leqslant d\left(x, o^{'}\right)$ for at least k objects $o \in D$

Definition 2 (Local-neighborhood) *For a given instance* x *of dataset* D *and a positive number* k*, the local-neighborhood of* x*, which is denoted as* $N_k(x)$*, contains every object* o *whose distance from* x *is not greater than the* $L_k(x)$*.*

$$N_k(x) = \{o | o \in D \text{ and } d(x, o) \leq L_k(x)\} \tag{3}$$

These objects o are called the k-nearest neighbors of instance x. It should be noted here that the $L_k(x)$ is well defined for any positive number k; however, the object o may not be unique. In this case, the size of $N_k(x)$ is greater than k. For example, assume that the distance of 1 object is 1 unit from x; the distances of 2 objects are 2 units from x; and the distances of 3 objects are 3 units from x. Even though the value of k is given like k=4, the size of $N_4(x)$ is greater than 4, in this case size of the $N_4(x)$ will be 6. The main

aim of the SDNN method is to optimize two criteria: (i) maximize the classification accuracy of the model, and (ii) minimize the number of neighbors involved in voting. The objective is to maximize the function ϕ given in Equation (4).

$$\phi = B * A(k) + (1 - B) * \sum_{o \in N_k(x)} \frac{1}{d(x, o)}, \tag{4}$$

where k is the candidate number of neighbors, $A(k)$ is an accuracy obtained by the model with the parameter k, and B is a user-specified parameter between 0 and 1. The output of the function increases as the model performance increases, and decreases as the sum of the distances between instance x and its neighbors increases. Increasing in the value of B means that the model performance is more important than the total distance. It is possible to establish a trade-off between the distance and accuracy by adjusting B.

In the proposed SDNN approach, the number of neighbors (k) can be variable $(1 \leq k \leq K_{max})$, which may help in the better interpretation of the neighborhood in the nonuniform data distribution. Searching k nearest objects to the given query instance provides an estimation about the data distribution. The probability of each object being selected as the nearest neighbor for a given query instance x is calculated by the algorithm. In other words, SDNN calculates the probabilistic chances of neighbors involved in the final classification decision process. It works by using a K_{max} value to find the k-nearest neighbors of a query instance. It sorts a list of nearest neighbors according to their distances and calculates the cumulative sum of inverse distances.

Definition 3 (Local-probability) *For a given query instance x, the local-probability of determining k as the number of neighborhood parameter is defined as the ratio of the sum of the distances in $N_k(x)$ to the cumulative sum of distances in $N_{Kmax}(x)$.*

$$p(k) = \frac{\sum_{o \in N_k(x)} \frac{1}{d(x,o)}}{\sum_{o \in N_{Kmax}(x)} \frac{1}{d(x,o)}}. \tag{5}$$

This probability assignment aims to give more chance to nearer objects while giving less chance to faraway objects. As the distance between the object o and query instance x increases, the function decreases the probability of being selected as the nearest neighbor. The frequency of selected nearest neighbors is then used to estimate the class label for a given query sample. The class with the most votes is then deemed the predicted class label for the testing data instance.

Every classifier in the ensemble makes a prediction for each query instance, and the final output prediction is the class label that receives more votes. This paper proposes a novel voting mechanism called stepwise voting. Stepwise voting is based on consensus decision-making among a group of learners iteratively.

Definition 4 (Stepwise voting) *Stepwise voting is a metaalgorithm that performs the decision process by dividing the classifiers' outputs into sequential groups and combining the votes in a step-by-step process.*

Formally, the stepwise voting result is calculated as follows:

$$argmax_{y \in Y} \left(\sum_{a=1}^{e-v-1} \sum_{b=1}^{e-v-a} argmax_{y \in Y} \left(\sum_{j=b}^{j+v-1} w\left(j, y\right) \right) \right), \tag{6}$$

where y is a class of Y label set, e is the ensemble size (number of models), v is the size of each group, and $w\left(j, c\right)$ is defined as in the following:

$$w\left(j, y\right) = \begin{cases} 1, if\ c = argmax_{y \in Y} P_j u \\ 0, otherwise \end{cases}, \tag{7}$$

where $P_j u$ is the output value of the j-th model to the u-th label in Y.

Algorithm 1 presents the pseudocode of the SDNN approach. In the first step, multiple training sets are created by resampling the original dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$ using the bootstrap method. After that, the user-defined K_{max} value is used for finding the k-nearest neighbors of a query instance of the D_i dataset. The nearest neighbors are found according to their distances, and the cumulative sum of inverse distances of them are calculated. In the following step, the k value is randomly determined based on the probability according to the ratio of the sum of the distances to the cumulative sum of distances. Finally, each DNN model in the ensemble classifies a sample instance considering the dynamic k value, and the outputs from each model are aggregated using the stepwise voting method to get the final output value.

Algorithm 1: Stepwise dynamic nearest neighbor (SDNN)

Inputs :

D: training set $D = \{(x_i, y_i)\}_{i=1}^{n}$
X: input feature space, an input vector $x_i \in X$
Y: class attribute, a class label $y_i \in Y = \{c_1, c_2, ..., c_r\}$
m: the number of features
s: ratio of features that will be selected
K_{max}: the maximum number of nearest neighbors
e: ensemble size
v: size value of stepwise voting procedure
T: test set that will be predicted

Output:

C: the estimated class labels

---

for $i = 1$ **to** e **do**
  D_i = bootstrap samples from D with s of m
end
foreach x *in* T **do**
  for $i = 1$ *to* e **do**
    $N_{Kmax}(x)$ = NearestNeighbors(D_i, x, K_{max})
    foreach *object in* $N_{Kmax}(x)$ **do**
      distances.Add(EuclidianDistance(*object*, x))
    end
    distanceSum = 0.0
    for $j = 1$ *to distances.Length* **do**
      distanceSum += *distances*[j]
    end
    cumulativeProbability = 0.0
    probability = random [0, *distanceSum*]
    for $k = 1$ *to distances.Length* **do**
      cumulativeProbability += (*distances*[k] / *distanceSum*)
      if *cumulativeProbability* > *probability* **then**
        break
      end
    end
    $N_k(x)$ = NearestNeighbors(D_i, x, k)
    y = Prediction($N_k(x), x$)
  end
  //Stepwise Voting
  $c = argmax_{y \in Y} \left(\sum_{a=1}^{e-v-1} \sum_{b=1}^{e-v-a} argmax_{y \in Y} \left(\sum_{j=b}^{j+v-1} w(j, y) \right) \right)$
  $w(j, y) = \begin{cases} 1, if\ c = argmax_{y \in Y} P_{ju} \\ 0, otherwise \end{cases}$
  $C = C \cup c$
end

The time complexity of the SDNN method is $O(T + L(n) * e)$, where e is the number of classifiers in the ensemble, T refers to the time needed for bootstapping, and $L(n)$ is the time required for running the nearest neighbor algorithm on n instances. Thanks to the recent developments in high-performance computing, it is possible to reduce the computational cost by the parallel execution of the method.

4. Experimental studies

In the experiments of this research, the proposed SDNN approach was applied to 50 different benchmark and commonly preferred datasets to demonstrate its classification ability. The application was implemented in C# language using the Weka open source data mining library [20]. Our method was compared with the traditional KNN algorithm, its variants, and the state-of-the-art methods in terms of recall, precision, F-measure, and accuracy rate metrics.

4.1. Dataset description

This study uses 50 different real-world datasets, which are available in the UCI Machine Learning Repository for public use to present the capabilities of the proposed SDNN method. These datasets include a number of instances ranging from 36 to 13611, classes ranging from 2 to 10, and attributes ranging from 4 to 61. The datasets come from different domains, including health, banking, marketing, environment, animal science, and automobile. They have different types of values, including numerical, categorical, or mix-type. It is obvious that many studies [21–23] in the literature use these datasets extensively.

4.2. Experimental results

In each experiment, the proposed SDNN approach was compared with the (i) traditional KNN algorithm, (ii) its variants, and the (iii) state-of-the-art methods by using the n-fold cross-validation technique selecting n as 10. The maximum number of nearest neighbors (K_{max}) was set to 5 in these experiments. The majority dynamic nearest neighbor (MDNN) method, which aggregates the obtained outputs from each individual DNN model using the classical majority voting mechanism, was also compared with the SDNN method. When testing the performances of the methods on the 50 different real-world datasets; the precision, recall, F-measure, and accuracy rate values were evaluated.

Table 1 presents the accuracy rate results obtained by the KNN and SDNN algorithms on the experimental datasets. As can be seen from this table, SDNN demonstrated higher classification ability over KNN, which are marked by bullets. For example, SDNN achieved significantly better classification performance (94.58%) than KNN (84.84%) on the monks dataset. According to the results given in Table 1, it is obviously seen that the proposed SDNN approach surpassed the traditional KNN algorithm in 43 of 50 datasets. Also, when the average accuracy rates obtained from the application of the KNN and SDNN approaches are considered in general, it is observed that SDNN (83.17%) provided better classification performance than KNN (79.39%) on average. This is because, thanks to the dynamic adjustment of the neighborhood number and stepwise voting mechanism, classification accuracy greatly improved compared to the classical KNN method.

Precision, recall, and F-measure values of the KNN and SDNN approaches were also calculated in addition to accuracy rate findings to provide additional insight into the classification performances by assessing all facets of the classifiers. Figure 4 plots these methods' average recall, precision, and F-measure scores on the 50 different real-world datasets. These metrics use a scale of 0 to 1 to indicate the success of the proposed method. The model with a higher rate is one that it has better classification ability than the rest. In this graph, it is clearly seen that the SDNN produces robust classification results with greater precision (0.834), recall (0.832), and F-measure (0.833) values.

Table 1. The accuracy rate (%) results of the traditional KNN and SDNN algorithms.

Datasets	KNN	SDNN	Datasets	KNN	SDNN
appendicitis	82.08	86.55 •	parkinsons	96.41 •	94.34
bank-marketing	86.51	88.92 •	pasture	69.44	78.33 •
blood-transfusion-service	70.45	75.93 •	pendigits	99.36	99.46 •
climate-simulation-craches	86.11	91.29 •	planning-relax	65.93	68.54 •
colon	82.26	86.90 •	postoperative-patient	66.67	71.11 •
credit-approval	81.16	85.51 •	saheart	63.20	67.97 •
dermatology	94.54	98.09 •	seeds	94.29 •	93.81
dresses-sales	54.00	60.20 •	sonar	86.54 •	84.55
dry-bean	90.30	92.41 •	spect-heart	68.91 •	67.37
ecoli	80.36	85.45 •	statlog-australian-credit	80.14	86.09 •
fertility-diagnosis	83.00	88.00 •	statlog-german-credit	72.00	75.20 •
habermans-survival	67.65	74.85 •	thoracic-surgery	77.23	83.40 •
heart-disease-cleveland	57.10	59.03 •	vehicle	69.86	72.93 •
heart-disease-hungarian	56.12	65.71 •	volcanoes-e1	87.74	91.55 •
heart-statlog	75.19	82.96 •	volcanoes-e2	86.76	91.11 •
hepatitis	80.65	83.83 •	volcanoes-e3	87.47	91.86 •
horse-colic-surgical	77.33	84.00 •	volcanoes-e4	86.10	91.37 •
indian-liver-patient	64.49	70.64 •	volcanoes-e5	86.69	90.83 •
ionosphere	86.32	88.32 •	wdbc	95.96	96.84 •
iris	95.33 •	94.67	white-clover	63.49	66.91 •
led7digit	70.80	72.40 •	wholesale-channel	87.95	90.45 •
liver-disorders	62.90	69.87 •	wholesale-region	53.41	68.18 •
lymph	82.43	82.48 •	wilt	95.10 •	94.94
molecular-promotor-gene	85.85	86.82 •	wine	94.94	96.63 •
monks	84.84	94.58 •	zoo	96.04 •	95.09

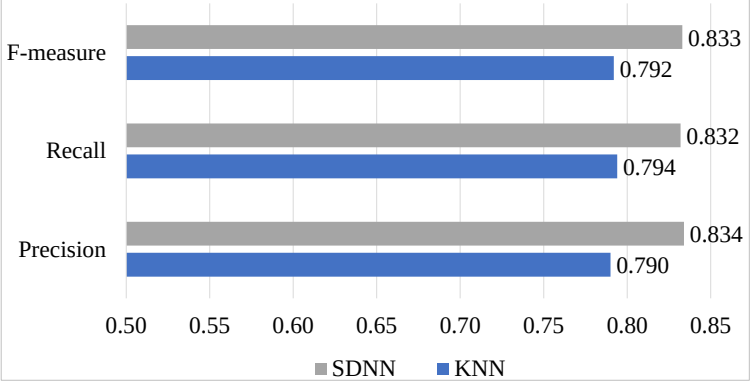

Figure 4. The average precision, recall, and F-measure results of the KNN and SDNN algorithms.

Furthermore, the DNN models proposed in this research were also compared with each other according to the majority voting (MDNN) and stepwise voting (SDNN) mechanisms they used. Table 2 shows the comparison of the accuracy rates obtained from the application of MDNN and SDNN algorithms on the experimental datasets. The results from Table 2 reveal that the SDNN approach achieved equal or higher accuracy rates than

MDNN in 43 of 50 datasets, which are marked by bullets. This result proved that stepwise voting provided more accurate results by eliminating the wrong outputs in local decisions.

Table 2. The accuracy rate (%) results of the traditional MDNN and SDNN algorithms.

Datasets	**MDNN**	**SDNN**	**Datasets**	**MDNN**	**SDNN**
appendicitis	87.55 •	86.55	parkinsons	93.82	94.34 •
bank-marketing	88.87	88.92 •	pasture	75.00	78.33 •
blood-transfusion-service	76.46 •	75.93	pendigits	99.42	99.46 •
climate-simulation-craches	91.29	91.29 •	planning-relax	67.43	68.54 •
colon	86.90	86.90 •	postoperative-patient	71.11	71.11 •
credit-approval	86.38 •	85.51	saheart	67.09	67.97 •
dermatology	97.82	98.09 •	seeds	92.86	93.81 •
dresses-sales	59.60	60.20 •	sonar	84.10	84.55 •
dry-bean	92.40	92.41 •	spect-heart	66.62	67.37 •
ecoli	85.44	85.45 •	statlog-australian-credit	85.65	86.09 •
fertility-diagnosis	88.00	88.00 •	statlog-german-credit	75.20	75.20 •
habermans-survival	74.53	74.85 •	thoracic-surgery	83.83 •	83.40
heart-disease-cleveland	58.38	59.03 •	vehicle	71.99	72.93 •
heart-disease-hungarian	65.71	65.71 •	volcanoes-e1	91.55	91.55 •
heart-statlog	82.60	82.96 •	volcanoes-e2	90.92	91.11 •
hepatitis	84.46 •	83.83	volcanoes-e3	91.86	91.86 •
horse-colic-surgical	83.67	84.00 •	volcanoes-e4	91.29	91.37 •
indian-liver-patient	70.13	70.64 •	volcanoes-e5	90.83	90.83 •
ionosphere	88.03	88.32 •	wdbc	96.84	96.84 •
iris	95.33 •	94.67	white-clover	66.91	66.91 •
led7digit	72.40	72.40 •	wholesale-channel	90.23	90.45 •
liver-disorders	69.03	69.87 •	wholesale-region	67.96	68.18 •
lymph	82.43	82.48 •	wilt	94.92	94.94 •
molecular-promotor-gene	86.82	86.82 •	wine	96.67 •	96.63
monks	93.86	94.58 •	zoo	94.09	95.09 •

4.3. Parameter analysis

The performances of the algorithms change under different parameter configurations. The accuracy rates obtained with the different distance metrics, several k parameter settings, a number of group sizes for stepwise voting, and various ensemble sizes are given in Tables 3–6, respectively.

The aim of the first parameter analysis is to compare the effect of different distance metrics (Euclidean, Manhattan, and Minkowski distance) on the classification performance of the SDNN method. As can be seen in Table 3, SDNN with Euclidean distance outperformed or at least performed as well as other SDNNs with the Manhattan and Minkowski distance measures. For example, the SDNN using Euclidean distance (84.55%) has better classification performance than the SDNNs using Manhattan (83.64%) and Minkowski distances (83.60%) on the sonar dataset. Similarly, Euclidean distance is a more suitable distance metric for the SDNN approach than others for the horse-colic-surgical dataset. When the results are considered in general, it is possible to say that the accuracy values are close to each other in all distance methods except for a few datasets such as the ionosphere and saheart datasets.

Table 3. The accuracy of SDNN measured according to different distance metrics.

Datasets	**Euclid.**	**Manh.**	**Minkow.**	**Datasets**	**Euclid.**	**Manh.**	**Minkow.**
appendicitis	86.55	87.55	86.55	parkinsons	94.34	94.32	94.34
bank-marketing	88.92	88.92	88.90	pasture	78.33	78.33	78.33
blood-transfusion-service	75.93	76.46	75.93	pendigits	99.46	99.47	99.44
climate-simulation-craches	91.29	91.29	91.30	planning-relax	68.54	67.98	68.51
colon32	86.90	86.90	85.24	postopera-tive.patient.data	71.11	71.11	71.11
credit-approval	85.51	86.09	85.80	saheart	67.97	69.47	67.96
dermatology	98.09	97.54	98.09	seeds	93.81	93.81	94.29
dresses-sales	60.20	60.40	60.60	sonar	84.55	83.64	83.60
dry-bean	92.41	91.92	92.34	spect-heart	67.37	68.10	67.37
ecoli	85.45	85.74	86.63	statlog-australian-credit	86.09	86.23	85.65
fertility-diagnosis	88.00	88.00	88.00	statlog-german-credit	75.20	75.30	74.80
habermans-survival	74.85	74.86	75.17	thoracic-surgery	83.40	83.62	83.62
heart-disease-processed-cleveland	59.03	59.71	58.38	vehicle	72.93	72.83	72.10
heart-disease-reprocessed-hungarian	65.71	67.07	66.39	volcanoes-e1	91.55	91.55	91.55
heart-statlog	82.96	82.97	82.59	volcanoes-e2	91.11	91.02	91.11
hepatitis	83.83	84.42	83.83	volcanoes-e3	91.86	91.86	91.94
horse-colic-surgical	84.00	82.33	83.00	volcanoes-e4	91.37	91.45	91.45
indian-liver-patient	70.64	70.47	71.32	volcanoes-e5	90.83	90.74	90.83
ionosphere	88.32	90.88	86.05	wdbc	96.84	96.84	96.49
iris	94.67	94.67	94.67	white-clover	66.91	65.24	66.91
led7digit	72.40	72.40	72.40	wholesale-channel	90.45	91.36	90.23
liver-disorders	69.87	68.42	68.71	wholesale-region	68.18	67.27	69.09
lymph	82.48	82.48	83.14	wilt	94.94	94.87	94.96
molecular-promotor-gene	86.82	86.82	86.82	wine	96.63	96.63	96.63
monks	94.58	94.58	94.58	zoo	95.09	95.09	95.09

In the second parameter analysis, we aimed to analyze the results of different K_{max} values to finally choose a proper value that gives a good classification performance. Table 4 presents the accuracy rates of the SDNN approaches selecting k values as 5, 10, and 15, respectively. According to the average accuracy results, it can be concluded that the best classification performance was achieved by selecting the parameter k as 5. The higher number of k values usually produced a lower classification performance since the dataset sizes are a little bit small. The maximum $k = 5$ is a reasonable choice since if k is set to larger than this, the neighborhood may cover many instances from other classes. There are significant accuracy differences between the parameter settings in a few datasets, including monks and pasture.

Table 4. The accuracy of SDNN measured according to different K_{max} values.

Datasets	**K=5**	**K=10**	**K=15**	**Datasets**	**K=5**	**K=10**	**K=15**
appendicitis	86.55	87.55	86.64	parkinsons	94.34	92.79	91.76
bank marketing	88.92	88.76	88.76	pasture	78.33	72.50	70.00
blood-transfusion-service	75.93	76.86	76.86	pendigits	99.46	99.36	99.27
climate-simulation-craches	91.29	91.66	91.48	planning-relax	68.54	69.65	70.18
colon	86.90	86.90	85.24	postoperative-patient	71.11	71.11	71.11
credit-approval	85.51	86.52	86.52	saheart	67.97	68.18	69.05
dermatology	98.09	97.55	97.28	seeds	93.81	92.86	92.86
dresses-sales	60.20	59.40	59.60	sonar	84.55	83.09	82.14
dry-bean	92.41	92.32	92.52	spect-heart	67.37	67.75	66.99
ecoli	85.45	85.73	85.44	statlog-australian-credit	86.09	85.80	86.23
fertility-diagnosis	88.00	88.00	88.00	statlog-german-credit	75.20	74.70	74.50
habermans-survival	74.85	74.19	74.86	thoracic-surgery	83.40	84.25	84.89
heart-disease-cleveland	59.03	57.71	56.73	vehicle	72.93	71.98	71.04
heart-disease-hungarian	65.71	66.39	65.01	volcanoes-e1	91.55	91.63	91.55
heart-statlog	82.96	84.45	83.34	volcanoes-e2	91.11	91.11	91.11
hepatitis	83.83	83.79	82.54	volcanoes-e3	91.86	91.70	91.62
horse-colic-surgical	84.00	84.00	84.33	volcanoes-e4	91.37	91.37	91.45
indian-liver-patient	70.64	70.99	69.62	volcanoes-e5	90.83	90.83	90.83
ionosphere	88.32	86.32	84.89	wdbc	96.84	97.19	97.02
iris	94.67	94.67	94.00	white-clover	66.91	70.00	71.67
led7digit	72.40	74.40	74.60	wholesale-channel	90.45	90.45	90.91
liver-disorders	69.87	68.70	67.61	wholesale-region	68.18	70.68	71.36
lymph	82.48	82.43	81.71	wilt	94.94	94.77	94.73
molecular-promotor-gene	86.82	83.91	83.00	wine	96.63	97.22	96.66
monks	94.58	87.90	90.24	zoo	95.09	94.09	94.09

The experimental results with the group sizes $\in \{50, 60, 70\}$ in the stepwise voting can be found in Table 5. From this table, it is possible to interpret that the best performance was obtained when the group size is 50. The SDNN approach with GS=50 achieved equal or higher accuracy rates on more than half of all datasets compared to other SDNN models with group sizes 60 and 70. Therefore, the increase in group size could not guarantee an increase in classification ability.

The parameter sensitivity of the learning rate of ensemble sizes was analyzed under the set of {75, 100, 125}. Related accuracy rate performances of the SDNN models were reported in Table 6. According to the table, the best suitable ensemble size was determined as 100. Increasing the ensemble size above 100 did not

Table 5. The accuracy of SDNN measured according to different group sizes (GS) in stepwise voting.

Datasets	**GS=50**	**GS=60**	**GS=70**	**Datasets**	**GS=50**	**GS=60**	**GS=70**
appendicitis	86.55	86.55	86.55	parkinsons	94.34	94.34	93.82
bank marketing	88.92	88.90	88.92	pasture	78.33	78.33	75.83
blood-transfusion-service	75.93	75.93	76.19	pendigits	99.46	99.45	99.45
climate-simulation-craches	91.29	91.29	91.29	planning-relax	68.54	68.54	69.65
colon	86.90	86.90	86.90	postoperative-patient	71.11	71.11	71.11
credit-approval	85.51	85.51	85.65	saheart	67.97	68.40	68.18
dermatology	98.09	98.09	97.82	seeds	93.81	93.81	93.33
dresses-sales	60.20	59.20	59.40	sonar	84.55	84.55	83.6
dry-bean	92.41	92.45	92.45	spect-heart	67.37	67.37	67.75
ecoli	85.45	86.04	85.74	statlog-australian-credit	86.09	85.94	85.65
fertility-diagnosis	88.00	88.00	88.00	statlog-german-credit	75.20	75.30	75.10
habermans-survival	74.85	75.18	74.53	thoracic-surgery	83.40	83.83	84.04
heart-disease-cleveland	59.03	58.70	58.05	vehicle	72.93	72.93	72.70
heart-disease-hungarian	65.71	65.38	65.72	volcanoes-e1	91.55	91.55	91.55
heart-statlog	82.96	82.59	82.22	volcanoes-e2	91.11	91.11	91.11
hepatitis	83.83	84.46	84.46	volcanoes-e3	91.86	91.86	91.86
horse-colic-surgical	84.00	84.00	84.00	volcanoes-e4	91.37	91.37	91.29
indian-liver-patient	70.64	70.12	69.78	volcanoes-e5	90.83	90.83	90.83
ionosphere	88.32	87.74	87.46	wdbc	96.84	96.66	96.66
iris	94.67	94.67	94.67	white-clover	66.91	66.91	65.24
led7digit	72.40	72.60	72.60	wholesale-channel	90.45	90.45	90.23
liver-disorders	69.87	69.56	69.27	wholesale-region	68.18	68.41	67.95
lymph	82.48	83.81	83.81	wilt	94.94	94.94	94.92
molecular-promotor-gene	86.82	86.82	86.82	wine	96.63	96.63	97.22
monks	94.58	94.4	94.77	zoo	95.09	95.09	95.09

significantly increase performance, and it even brought higher computational costs. Therefore, a large number of classifiers cannot guarantee a remarkably more satisfying result. We consider that a possible reason behind this is that although increasing the ensemble size often leads to better performance in terms of accuracy, there is still a risk of overfitting.

4.4. Comparison with the KNN variants

The proposed SDNN approach was also compared with the KNN variants in the literature, named uniform KNN [15], weighted KNN [16], KNN learning with graph neural networks (KNNGNN) [17], dynamic k-nearest-

Table 6. The accuracy of SDNN measured according to different ensemble sizes (ES).

Datasets	ES=75	ES=100	ES=125	Datasets	ES=75	ES=100	ES=125
appendicitis	86.64	86.55	85.73	parkinsons	94.84	94.34	94.84
bank marketing	88.83	88.92	88.79	pasture	78.33	78.33	79.17
blood-transfusion-service	76.87	75.93	76.20	pendigits	99.44	99.46	99.47
climate-simulation-craches	91.48	91.29	91.48	planning-relax	67.98	68.54	69.12
colon	85.24	86.90	88.57	postoperative-patient	71.11	71.11	71.11
credit-approval	86.38	85.51	87.25	saheart	67.09	67.97	66.65
dermatology	97.54	98.09	97.82	seeds	93.33	93.81	92.86
dresses-sales	59.40	60.20	56.80	sonar	84.07	84.55	82.66
dry-bean	92.41	92.41	92.31	spect-heart	67.01	67.37	67.39
ecoli	86.03	85.45	85.15	statlog-australian-credit	85.36	86.09	84.78
fertility-diagnosis	88.00	88.00	88.00	statlog-german-credit	75.60	75.20	74.00
habermans-survival	74.53	74.85	75.82	thoracic-surgery	83.40	83.40	84.25
heart-disease-cleveland	58.38	59.03	59.04	vehicle	72.58	72.93	72.22
heart-disease-hungarian	66.38	65.71	66.39	volcanoes-e1	91.55	91.55	91.55
heart-statlog	82.22	82.96	82.22	volcanoes-e2	91.11	91.11	91.20
hepatitis	83.21	83.83	82.62	volcanoes-e3	91.86	91.86	91.86
horse-colic-surgical	82.67	84.00	84.33	volcanoes-e4	91.29	91.37	91.29
indian-liver-patient	70.13	70.64	69.79	volcanoes-e5	90.65	90.83	90.74
ionosphere	87.74	88.32	88.02	wdbc	97.02	96.84	96.84
iris	94.67	94.67	96.00	white-clover	66.90	66.91	68.57
led7digit	73.20	72.40	72.80	wholesale-channel	90.00	90.45	90.68
liver-disorders	66.98	69.87	71.08	wholesale-region	67.73	68.18	67.50
lymph	84.52	82.48	84.48	wilt	95.02	94.94	94.94
molecular-promotor-gene	86.64	86.82	88.55	wine	96.67	96.63	96.67
monks	94.77	94.58	95.48	zoo	94.09	95.09	94.09

neighbor with distance and attribute weighted (DKNDAW) [18], weight-adjusted k-nearest-neighbor (WAKNN) [18], k-nearest-neighbor with distance weighted (KNNDW) [18], k-nearest-neighbor with distance and attribute weighted (KNNDAW) [18], and dynamic k-nearest-neighbor (DKNN) [18] in terms of accuracy rates they produced. The results for the datasets, which were given in the papers, were directly taken from the referenced studies. According to Table 7, it is possible to say that the SDNN algorithm outperformed all the 8 KNN variants in 11 of 15 datasets. For example, SDNN (94.34%) showed its superiority over KNNGNN (84.64%) on the parkinsons dataset. SDNN achieved the highest classification ability on the wine dataset with an accuracy of 96.63%. The SDNN method (95.09%) obtained the highest improvement (17.78%) compared with the uniform

KNN method (77.31%) on the zoo dataset. On average, SDNN (86.69%) outperformed uniform KNN, weighted KNN, and KNNGNN with accuracy rates of 81.75%, 83.93%, and 85.54%, respectively. Similarly, on average, our method (86.69%) performed better than DKNDAW (83.21%), WAKNN (80.07%), KNNDW (82.77%), KNNDAW (81.79%), and DKNN (81.35%). The findings from the experiments revealed that the classification task was improved since our method relied on implicit and explicit distances.

Table 7. Comparison of the proposed SDNN method against the KNN variants on the same datasets in terms of classification accuracy (%).

Datasets	**Uniform KNN** [15]	**Weighted KNN** [16]	**KNN GNN** [17]	**DKN DAW** [18]	**WA KNN** [18]	**KNN DW** [18]	**KNN DAW** [18]	**DKNN** [18]	**SDNN (proposed)**
ecoli	81.47	82.97	81.51	-	-	-	-	-	**85.45**
heart-statlog	79.19	79.07	78.74	81.15	80.56	82.70	80.74	78.44	**82.96**
hepatitis	-	-	-	81.35	81.12	**84.22**	80.49	78.34	83.83
horse-colic-surgical	-	-	-	76.52	74.37	72.57	74.72	74.10	**84.00**
ionosphere	81.92	82.10	91.62	**92.51**	90.54	90.17	90.83	91.79	88.32
iris	91.14	92.34	94.27	94.93	93.87	93.47	**96.40**	92.73	94.67
lymph	-	-	-	81.97	80.77	81.29	80.36	81.16	**82.48**
parkinsons	83.66	86.27	84.64	-	-	-	-	-	**94.34**
seeds	91.91	92.21	91.60	-	-	-	-	-	**93.81**
sonar	70.71	73.16	76.13	79.71	72.42	82.01	73.63	80.54	**84.55**
statlog-australian-credit	-	-	-	85.80	84.65	85.51	84.94	81.93	**86.09**
statlog-german-credit	-	-	-	74.82	74.43	72.90	74.36	70.74	**75.20**
vehicle	70.49	71.20	72.25	69.89	65.29	69.83	66.87	68.94	**72.93**
wine	89.65	91.39	94.78	-	-	-	-	-	**96.63**
zoo	77.31	88.55	89.88	**96.65**	82.71	95.84	96.33	96.15	95.09

4.5. Comparison with the state-of-the-art methods

Table 8 shows the comparison of our proposed method's results with the results of different algorithms (Bsnsing [21], sparse projection oblique randomer forests (SPORF) [22], random forest (RF) [22], extreme gradient boosting (XGBoost) [22], random rotation random forest (RR-RF) [22], canonical correlation forest (CCF) [22], and Adaptative Credal C4.5 (AdaptativeCC4.5) [23]) presented in previous work [21–23] on the same datasets. As shown in this table, the SDNN method usually reached higher accuracy rate values than the previous methods applied to the same dataset. For example, the SDNN performed better on the hepatitis dataset in comparison to the state-of-the-art methods. It can be concluded that the SDNN approach outperformed the other methods with 5.4% improvement on average.

Table 8. Comparison of the proposed SDNN method against the state-of-the-art methods on the same datasets in terms of classification accuracy (%).

Datasets	**BSNS-ING [21]**	**SPORF [22]**	**RF [22]**	**XGBoost [22]**	**RR-RF [22]**	**CCF [22]**	**Adaptative CC4.5 [23]**	**SDNN**
bank-marketing	90.1	91.7	**91.9**	**91.9**	91.3	91.6	-	88.92
blood-transfusion-service	-	71.00	72.00	72.00	71.00	70.00	-	**75.93**
climate-simulation-craches	**92.50**	-	-	-	-	-	-	91.29
credit-approval	85.10	77.00	78.00	77.00	74.00	75.00	-	**85.51**
dermatology	91.50	98.00	98.00	97.00	96.00	96.00	93.96	**98.09**
ecoli	-	83.00	81.00	81.00	83.00	81.00	81.72	**85.45**
fertility- diagnosis	86.00	-	-	-	-	-	-	**88.00**
habermans-survival	72.90	63.00	61.00	63.00	57.00	54.00	71.31	**74.85**
heart-disease-cleveland	**76.80**	47.00	48.00	47.00	51.00	45.00	-	59.03
heart-disease-hungarian	-	**89.00**	87.00	87.00	81.00	85.00	-	65.71
heart-statlog	-	-	-	-	-	-	80.59	**82.96**
hepatitis	79.80	44.00	42.00	37.00	38.00	54.00	80.62	**83.83**
horse-colic-surgical	-	76.00	74.00	80.00	77.00	77.00	**85.24**	84.00
indian-liver-patient	68.20	60.00	59.00	60.00	62.00	63.00	-	**70.64**
ionosphere	88.10	85.00	82.00	81.00	88.00	86.00	88.27	**88.32**
iris	94.50	91.00	94.00	92.00	94.00	**96.00**	94.13	94.67
led7digit	-	68.20	68.60	69.50	67.90	67.60	-	**72.40**
liver-disorders	-	-	-	-	-	-	67.32	**69.87**
lymph	81.00	-	-	-	-	-	78.58	**82.48**
monks	87.10	86.00	86.00	81.00	87.00	81.00	-	**94.58**
parkinsons	85.80	69.00	75.00	67.00	67.00	75.00	-	**94.34**
pendigits	99.4	99.50	99.10	99.10	99.30	**99.60**	96.54	99.46
planning-relax	61.70	60.00	60.00	48.00	62.00	59.00	-	**68.54**
seeds	92.20	91.00	90.00	89.00	88.00	90.00	-	**93.81**
sonar	72.10	-	-	-	-	-	71.67	**84.55**
spect-heart	-	60.00	50.00	40.00	60.00	50.00	-	**67.37**
statlog-australian-credit	85.10	77.00	78.00	76.00	72.00	74.00	-	**86.09**
statlog-german-credit	67.60	66.40	65.00	63.60	61.60	64.30	71.70	**75.2**
vehicle	**95.20**	74.00	68.00	69.00	69.00	77.00	73.07	72.93
wdbc	94.40	-	-	-	-	-	-	**96.84**
wine	91.10	95.00	94.00	**97.00**	96.00	**97.00**	92.47	96.63
zoo	**99.50**	93.00	94.00	93.00	94.00	94.00	93.22	95.09

5. Conclusion and future work

One of the most popular machine learning algorithms is the KNN algorithm, which has intriguing qualities, including simple implementation and excellent generalizability. Although several approaches have been proposed

to the literature to increase its precision, the KNN still has some limitations on its classification ability. For this purpose, this study proposes a novel algorithm, SDNN, that utilizes a dynamic neighborhood procedure according to the data distribution and applies a new voting mechanism named stepwise voting. This voting mechanism focuses on splitting the outputs of the classifiers into consecutive groups and sequentially aggregates the votes to get the final output. In the experiments, the proposed SDNN approach was applied to 50 different benchmark datasets and compared with the traditional KNN algorithm, its variants, and the state-of-the-art methods in terms of recall, precision, F-measure, and accuracy measures. The obtained results present that the proposed SDNN method outperformed the KNN and MDNN in terms of the average accuracy rates. Also, it is seen from the experimental results that the SDNN achieved the highest performance (83.17% on average) by using Euclidean distance and setting k, ensemble size, and group size for stepwise voting parameters as 5, 100, and 50, respectively. In addition to these results, the SDNN also surpassed the KNN variants and the state-of-the-art methods.

For future studies, the SDNN can be modified by using different distance measures instead of Euclidean distance. In addition, the proposed stepwise voting mechanism can be adapted to different ensemble structures to enhance the classification ability of the models.

References

[1] García-Pedrajas N, Del Castillo JAR, Cerruela-García G. A proposal for local k values for k-nearest neighbor rule. IEEE Transactions on neurol networks and learning systems 2017; 28 (2): 470-475. https://doi.org/10.1109/TNNLS.2015.2506821

[2] Lu J, Qian W, Li S, Cui R. Enhanced k-nearest neighbor for intelligent fault diagnosis of rotating machinery. Applied Science 2021; 11 (3): 919-934. https://doi.org/ 10.3390/app11030919

[3] Gou J, Ma H, Ou W, Zeng S, Rao Y et al. A generalized mean distance-based k-nearest neighbor classifier. Expert Systems with Applications 2019; 115: 356-372. https://doi.org/10.1016/j.eswa.2018.08.021

[4] Saadatfar H, Khosravi S, Joloudari JH, Mosavi A, Shamshirband S. A new k-nearest neighbors classifier for big data based on efficient data pruning. Mathematics 2020; 8 (2): 1-12. https://doi.org/10.3390/math8020286

[5] Durbin M, Wonders MA, Flaska M, Lintereur A. K-nearest neighbors regression for the discrimination of gamma rays and neutrons in organic scintillators. Nuclear Instruments and Methods in Physics Research 2021; 987: 1-9. https://doi.org/10.1016/j.nima.2020.164826

[6] Xia J, Zhang J, Wang Y, Han L, Yan H. WC-KNNG-PC: Watershed clustering based on k-nearest-neighbor graph and Pauta Criterion. Pattern Recognition 2022; 121: 1-14. https://doi.org/10.1016/j.patcog.2021.108177

[7] Yu L, Yue W, Wang S, Lai KK. Support vector machine based multiagent ensemble learning for credit risk evaluation. Expert Systems with Applications 2010; 37 (2): 1351-1360. https://doi.org/10.1016/j.eswa.2009.06.083

[8] Okediran TM, Vincent OR, Abayomi-Alli AA, Adeniaran OJ. An IOT-based COVID-19 detector using k-nearest neighbor. Decision Sciences for COVID-19, International Series in Operations Research & Management Science 2022; 320: 27-43. https://doi.org/10.1007/978-3-030-87019-5_2

[9] Timuş O, Doğru Bolat E. K-NN-based classification of sleep apnea types using ECG. Turkish Journal of Electrical Engineering and Computer Sciences 2017; 25 (4): 3008-3023. https://doi.org/10.3906/elk-1511-99

[10] Dilmaç S, Ölmez Z, Ölmez T. Comparative analysis of MABC with KNN, SOM, and ACO algorithms for ECG heartbeat classification. Turkish Journal of Electrical Engineering and Computer Sciences 2018; 26 (6): 2819-2830. https://doi.org/10.3906/elk-1712-328

[11] Tang H, Xu Y, Lin A, Heidari AA, Wang M et al. Predicting green consumption behaviors of students using efficient firefly grey wolf-assisted k-nearest neighbor classifiers. IEEE Access 2020; 8: 35546-35562. https://doi.org/10.1109/ACCESS.2020.2973763

[12] Hemachandran K, George PM, Rodriguez R, Kulkarni RM, Roy S. Performance analysis of k-nearest neighbor classification algorithms for bank loan sectors. Smart Intelligent Computing and Communication Technology 2020; 38: 9-13. https://doi.org/10.3233/APC210004

[13] Qureshi S, Karrila S, Vanichayobon S. Human sleep scoring based on k-nearest neighbors. Turkish Journal of Electrical Engineering and Computer Sciences 2018; 26 (6): 2802-2818. https://doi.org/10.3906/elk-1805-12

[14] Dilmaç F, Alpkoçak A. Automatic keyword assignment system for medical research articles using nearest-neighbor searches. Turkish Journal of Electrical Engineering and Computer Sciences 2022; 30 (5): 1821–1838. https://doi.org/10.55730/1300-0632.3907

[15] Wolff B, Lorenz E, Kramer O. Statistical learning for short-term photovoltaic power predictions. Computational Sustainability 2016; 645: 31–45. https://doi.org/10.1007/978-3-319-31858-5_3

[16] Geler Z, Kurbalija V, Ivanovic M, Radovanovic M. Weighted kNN and constrained elastic distances for time-series classification. Expert System with Applications 2020; 162: 1-12. https://doi.org/10.1016/j.eswa.2020.113829

[17] Kang S. K-nearest neighbor learning with graph neural networks. Mathematics 2021; 9 (8): 1-12. https://doi.org/10.3390/math9080830

[18] Wu J, Cai Z, Gao Z. Dynamic k-nearest-neighbor with distance and attribute weighted for classification. In: International Conference on Electronics and Information Engineering; Kyoto, Japan; 2010. pp. 356-360.

[19] Zhong X-F, Guo S-Z, Gao L, Shan H, Zheng J-H. An improved k-NN classification with dynamic k. In: International Conference on Machine Learning and Computing; Singapore; 2017. pp. 211–216.

[20] Witten IH, Frank E, Hall MA, Pal CJ. Data Mining: Practical Machine Learning Tools and Techniques. 4th ed. Cambridge, MA, USA: Morgan Kaufmann, 2016.

[21] Liu Y. Bsnsing: a decision tree induction method based on recursive optimal boolean rule composition. Informs Journal on Computing 2022; 34 (6): 2867-3350. https://doi.org/10.1287/ijoc.2022.1225

[22] Tomita TM, Browne J, Shen C, Chung J, Patsolic JL et al. Sparse projection oblique randomer forests. Journal of Machine Learning Research 2020; 21 (1): 4193–4231.

[23] Abellan J, Mantas CJ, Castellano JG. AdaptativeCC4.5: Credal C4.5 with a rough class noise estimator. Expert Systems with Applications 2018; 92: 363–379. https://doi.org/10.1016/j.eswa.2017.09.057