TÜBİTAK

Turkish Journal of Electrical Engineering & Computer Sciences

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

Research Article

Turk J Elec Eng & Comp Sci
(2021) 29: 1401 – 1428
© TÜBİTAK
doi:10.3906/elk-2005-59

# An improved version of multi-view k-nearest neighbors (MVKNN) for multiple view learning

**Elife OZTURK KIYAK**[1], **Derya BIRANT**[2,*], **Kokten Ulas BIRANT**[2]
[1]The Graduate School of Natural and Applied Sciences, Dokuz Eylül University, İzmir, Turkey
[2]Department of Computer Engineering, Faculty of Engineering, Dokuz Eylül University, İzmir, Turkey

**Received:** 08.05.2020 • **Accepted/Published Online:** 12.03.2021 • **Final Version:** 31.05.2021

**Abstract:** Multi-view learning (MVL) is a special type of machine learning that utilizes more than one views, where views include various descriptions of a given sample. Traditionally, classification algorithms such as k-nearest neighbors (KNN) are designed for learning from single-view data. However, many real-world applications involve datasets with multiple views and each view may contain different and partly independent information, which makes the traditional single-view classification approaches ineffective. Therefore, this article proposes an improved MVL algorithm, called multi-view k-nearest neighbors (MVKNN), based on the existing KNN algorithm. The experimental results conducted in this research show that a significant improvement is achieved by the proposed MVKNN algorithm compared to the well-known machine learning algorithms (KNN, support vector machine, decision tree, and naive bayes) in the case of multi-view data. The results also show that our method outperforms the state-of-the-art multi-view learning methods in terms of accuracy.

**Key words:** Machine learning, multi-view learning, classification, k-nearest neighbors

## 1. Introduction

*Classification* is one of the most significant supervised machine learning (ML) techniques that has been widely used in many fields such as health [1], energy [2], computer networks [3], transportation [4], and agriculture [5]. The main aim of classification task is to build an accurate model from a training dataset to correctly assign appropriate class labels to previously unseen data instances. Most of the previous classification studies in the literature have been performed on single-view data and used conventional classification algorithms, such as artificial neural network, decision tree, and support vector machine. Our study differs from these previous works since we especially focused on *multi-view classification*, which is based on a multi-view learning (MVL) framework.

*Multi-view learning* is a typical ML approach, which considers learning from multiple different feature sets (formally called views) to improve the generalization ability. Here, data is collected from multiple information sources. A *view* is a feature set that is obtained from a different data source. For each example, there is a corresponding instance in each view.

Many real-world applications involve multi-view datasets, where the input feature space contains multiple feature vectors. For example, in web mining, web pages contain rich multi-view data, such as textual content, images, and hyperlinks [6]. In multimedia mining, images can be described by distinct types of feature sets,

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

such annotated text (one view) and color histograms (the other view), and so it is desirable to improve image classification performance by fusion of information from different views [7]. Another typical example is the natural language processing task, in which the same document can have multiple representations in different languages [8]. In music emotion recognition, multiple views (such as lyrics and music) coexist in each song instance, where each song has only one class label (an emotion) [9]. As another example, in television broadcast understanding, broadcast segments can be described by audio signals but also can be represented by video signals - each of these views would be sufficient to determine the identity [10]. In short, MVL is a very promising and attractive research topic with wide applicability.

In machine learning, we usually can't put together all the features from different views because each view has diverse properties and has its own intrinsic structure. Therefore, multi-view learning techniques are needed to analyze data with multiple distinct types of views. To accomplish this goal, this paper introduces a multi-view learning algorithm.

In this article, we propose an improved multi-view learning, especially multi-view classification, method based on the k-nearest neighbors (KNN) algorithm. The proposed method, named *multi-view k-nearest neighbors (MVKNN)*, has two major phases. In the first phase, the algorithm learns from each view data separately to construct a weak classifier for each view. In the second phase, it combines the classifiers trained in each view to build a strong multi-view model.

The major contributions of this article can be highlighted as follows:

- *Novelty*: This paper proposes a MVL algorithm, called multi-view k-nearest neighbors (MVKNN).
- *Significance*: The experimental results reported in this paper show that multi-view KNN achieves higher classification accuracy and has better generalization ability compared to KNN since it considers the complementary and consistency properties of different views. The results also show that the proposed MVKNN method outperforms the well-known machine learning methods (i.e., support vector machine (SVM), decision tree (DT), and naive Bayes (NB)) and the existing multi-view learning methods (multi-view least squares support vector machines (MV-LSSVM) [11], KNN-based multi-view learning (MVL-KNN) [12], multi-view learning with the least square loss function (MVL-LS) [13], and multi-view decision tree (MVDT) [14].
- *Benefits*: Multi-view KNN has been proposed to combine the benefits of increasing completeness and robustness of classification models, achieving different tasks, taking the advantages of ensemble learning, facilitating learning tasks, dealing with high-dimensional data, and extending the application areas.

The rest of this article is simply structured as follows: In Section 2, we give a summary of related works on multi-view learning and k-nearest neighbors. In Section 3, we explain the KNN algorithm briefly, and then introduce our proposed approach and describe its benefits. In Section 4, the experimental studies are presented in detail, and obtained results are discussed. The last section, Section 5, presents concluding remarks as well as future directions.

## 2. Related work

### 2.1. Related studies on multi-view learning

With the rapid development of information systems, data can be obtained from multiple sources. Combining information from various data sources into a single record has been widely used in a variety of classification studies. However, different data sources may include different and partially independent information, which makes the previous single-view approaches ineffective. The multi-view learning (MVL) is useful for reducing the noise, as well as for improving statistical inference to reflect the underlying structure more explicitly and exploring the interactions between views to obtain more higher-level and refined knowledge [15]. By this motivation, in this study, we propose an improved MVL method that learns from different feature sets concerning each view individually via a joint structured model.

The views can be multiple measurement modalities, such as jointly represented images + text [16], audio + video [17], environmental + genetic information in ecological applications, chemical + biological data in drug discovery [18], documents (i.e., title, author, journal) + co-citation network (graph), or parallel text in two different languages [8].

In general, with abundant information, learning from multi-view data leads to an improvement in classification accuracy [16, 17, 19, 20]. Therefore, multi-view learning (MVL) has been used in a variety of areas, such as health [18], security [21], and music [9]. The MVL has been commonly applied for supervised learning [9], unsupervised learning [15] and semi-supervised learning [6, 8, 10, 22]. In the literature, many studies on multi-view learning have been focused on classification problems [7, 9, 21]; however, recently, exploiting multiple views to improve clustering has received increasing attention from the researchers [23]. To ensure the effectiveness of MVL, view evaluation has been considered in several studies [19, 23].

Existing MVL algorithms in the literature can be generally categorized into three main groups: co-training, subspace learning, and multiple kernel learning style algorithms [20]. The *co-training* style methods build separate learners for each view and enforce the agreement of the learners. Robust co-training [22] and Bayesian co-training [24] are typical examples of the co-training style algorithms. The *subspace learning* style methods purpose to obtain an appropriate latent subspace shared between multiple different views [25]. The *multiple kernel learning* (MKL) methods involve a set of predefined kernels which correspond to different views [26].

Some standard learning algorithms were adapted to directly deal with multi-view data such as multi-view support vector machine [11, 13, 27], multi-view decision tree [14], multi-view Adaboost [28], and multi-view neural network [21]. Sun and Zhang [29] proposed an ensemble learning method using multiple views and multiple learners (MVML) as in our study. In other words, similar to MVML, our method adopts ensemble styled learning paradigm. However, MVML is a framework for semi-supervised learning, which uses both labeled and unlabeled data, whereas our method is a supervised learning method. Furthermore, as a base learner, the KNN algorithm is used in MVKNN, while neural networks are utilized in MVML. MVML involves different techniques such as co-training and principal component analysis. MVML gives theoretical support to co-training working with two views, while MVKNN considers a combination of multiple views, all together.

In the literature, combined-style fusion strategy studies benefit from both early and late fusion strategies. For instance, Houthuys et al. [11] proposed a multi-view least squares SVM method (MV-LSSVM), which combines the late and early fusion by allowing for a different regularization parameter and a different kernel function for the different views. They introduced a coupling parameter that minimizes a combination of the errors from all views. Another algorithm, called multi-view learning with the least-squares loss function (MVL-LS), was proposed by Minh et al. [13]. It is a multi-view semi-supervised classification method based on SVM; therefore, it has the ability to handle both labeled and unlabeled data. Their study incorporates between-view interactions by adding up pairwise regularization terms between two views. Although their approach is a widely-used method in multi-view learning studies, it may fail to include higher-order correlations (interactions between three or more views) since it considers co-regularization.

In the study of [14], the authors introduced a multi-view one-versus-all (OVA) model based on decision tree (MVDT) for multi-classification tasks to simplify the structure of the decision tree and improve the generalization ability. They adopted various decision tree algorithms as base learners individually, including C4.5, classification and regression tree (CART), Tsallis entropy information metric (TEIM), size constrained decision tree (SCDT), and naive Bayes tree (NBTree). However, they divided a multi-class classification task into multiple sub-tasks, and the MVDT method builds a separate decision tree for each subtask.

Differently from the previous researches, we enhanced the KNN algorithm and proposed the multi-view KNN method, which separately treats properties from different views before combining them with a voting mechanism.

### 2.2. Related studies on k-nearest neighbors

Until now, a number of classification algorithms have been developed, among which KNN has been recognized as one of the top 10 machine learning algorithms [30], because of its simplicity, efficiency, and easy implementation. Therefore, the KNN algorithm has been widely used in many ML applications, especially for classification and regression. As shown in Table 1, different multiple variations of the KNN algorithm have been proposed until now [32–37], such as multi-label KNN [32], multi-instance KNN [34, 37], and multi-task KNN [35]. In contrast to these present types, a different kind of the KNN algorithm, called multi-view KNN, is proposed in this study.

**Table 1**. Multiple types of KNN-based algorithms.

| Author | Year | Algorithm | Type of Learning |
|---|---|---|---|
| Jiang et al. [31] | 2020 | MV-LLKNN | Multi-view local linear learning |
| Srivastava and Singh [32] | 2019 | ML-KNN | Multi-label learning |
| Xia et al. [33] | 2017 | DEMST-KNN | Multi-class learning |
| Villar et al. [34] | 2016 | Fuzzy-Citation-KNN | Multi-instance learning |
| Gupta et al. [35] | 2016 | ssMTTL-KNN | Multi-task learning |
| Peng et al. [36] | 2014 | ID-KNN | Multi-instance multi-label learning |
| Zhao et al. [37] | 2013 | MICkNN | Multi-instance learning |
| Liu et al. [38] | 2013 | MultiNMF | Multi-view clustering |
| Liang et al. [12] | 2012 | MVL-KNN | Multi-view learning |
| Our study | | MVKNN | Multi-view learning |

Liang et al. [12] proposed a KNN-based multi-view learning method to assess the therapeutic effect of clinical acupuncture treatment. Our proposed method differs from their approach in three respects. First, they find $k$ nearest neighbors of a test sample for each view, then the intersection of nearest records from all views yields a reference set. However, we classify a test sample according to each different view, and then we apply a voting mechanism to combine the predictions of views for final prediction. Second, they use a single $k$ value for each view. However, we propose different $k$ values for each view and for each instance to improve classification results. Third, we adapt the traditional KNN algorithm without any modification in its structure; however, they modified the standard KNN algorithm, which leads to implementation difficulties.

Jiang et al. [31] proposed a multi-view local linear KNN (MV-LLKNN) method for image classification.

Our method completely differs from their approach in many respects. First, they handle special multi-view scenarios where data from each view keep the same features. However, our method has not such a limitation. Second, they used completely different techniques such as fast iterative shrinkage thresholding algorithm (FISTA), clustering effect of the nearest neighbors (CENN), and Bayesian decision rule. Third, we adapt the traditional KNN algorithm without any modification in its structure. However, they modified it using the linear combination of samples considering locality and sparsity, which leads to implementation difficulties. Four, they consider views in a pairwise manner, i.e., view1-view2, view2-view3, and so on, whereas we consider the combination of all views, all together. Lastly, they proposed their method for a specific problem (image classification), i.e., each view involves the same face image under different lighting conditions and different facial expressions. However, our method is designed for general-purpose and can be applied to various domains.

Liu et al. [38] proposed an NMF-based (nonnegative matrix factorization) multi-view clustering algorithm, called MultiNMF. Our method differs from their method in many respects. First, they used different techniques such as matrix factorization, normalization, iterative optimization, and probabilistic latent semantic analysis. Second, MultiNMF is a clustering algorithm to solve an unsupervised learning problem, whereas MVKNN is a classification algorithm that can be used to solve a supervised learning problem. Third, MultiNMF creates coefficient matrices learned from different views to form a consensus, while we use majority voting for consensus. Since the researchers used the same dataset (UCI multi-feature digit) as our study, it is possible two compare the performances of the methods. According to experimental results, the accuracy value obtained by the MultiNMF method (88.1%) is lower than the accuracy value achieved by our MVKNN method (95.45%).

## 3. Material and methods

### 3.1. K-nearest neighbors

Given a training dataset with $n$ labeled instances $D = \{(x_1, y_1), (x_2, y_2), ...., (x_n, y_n)\}$, each instance $x_i$ is a feature vector with $d$-dimension such that $x_i = (x_{i1}, x_{i2}, ..., x_{id})$ and belongs to an instance space $X$, $x_i \epsilon X$. Assuming there are $c$ different class labels, each class label, $y_i$, is from a set of $Y = \{1, 2, ..., c\}$. A *classifier* is a function in the form of $f : X \rightarrow Y$ that maps an unseen instance $x \epsilon X$ onto an item of $Y$. Given a set of training instances, the classification task is to provide a definition for the function $f$. The KNN algorithm works as following: given a distance measure (i.e. Euclidean, Manhattan distance metrics), the set $N_{x,k}$ denotes the $k$ nearest neighbors of a data instance $x$ in the dataset $D$. The class label $y$ assigned to instance $x$ is the major class within this $k$ most similar neighbors ($N_{x,k}$).

The KNN algorithm is typically working on single-view data. A typical solution is to consider concatenating all multiple views into a single view and to apply the KNN algorithm directly on the combined dataset. However, this strategy not only causes overfitting problems on small training sets but also neglects the particular statistical characteristic of each view. For this reason, in this article, we propose the multi-view KNN method that learns one function to model each view individually and then collectively combines these functions to improve the classification accuracy.

### 3.2. The proposed approach: multi-view k-nearest neighbors

In this article, we propose a multi-view classification algorithm, called multi-view k-nearest neighbors (MVKNN). We chose the KNN algorithm to apply multi-view learning because of its advantages such as simplicity, easy implementation, and effectiveness [39]. The advantages of KNN also include ease of understanding and interpretation of the results, and useful for nonlinear data. Furthermore, it is robust to noisy training data and can be implemented for multi-class classification [40]. Moreover, it is capable of predicting both categorical or discrete variables; hence, it can be used for both classification and regression tasks. The other advantages of KNN are independent of any data distribution, the usage of local information, and easy interpretation of the results. Besides, it easily supports incremental data; so training is not required for the newly-arrived training samples. KNN has been proven to be a very effective classifier in various applications [40] and therefore, it has been widely used in many fields [1, 3, 12, 31–37, 39, 40].

Supposed that $(X^v, Y)$ is a sample of view $v$ for $v = 1, 2, ..., V$, where $X^v$ refers to the feature set of view $v$, $Y$ is the class label, and $V$ is the number of views. Given that, $D^v \in \mathbb{R}^{m^v xn}$ is the set of data instances of view $v$, where $n$ is the number of instances and $m^v$ is the number of features of each instance of view $v$. More specifically, we have $v$ views and each view can be expressed as $X^v$, and includes a feature set such that $X^v = (x_1^v, x_2^v, ..., x_n^v)$, where $x_i^v \in \mathbb{R}^{m^v}$. Views are disjointed from each other, so they include different feature sets such that $\forall_{p,q,} X^p \cap X^q = \emptyset$. Note that, all the views share the same class label $Y$ since they are the various representation of the same object, and $Y = (y_1, y_2, ..., y_n)$. The goal is to predict $Y$ for a given test sample. Formally, given a multi-view dataset of $n$ instances, $D = \{(X, Y)\} = \{(X_i, y_i), i = 1, 2, ..., n\} = \{(X_i^1, X_i^2, ..., X_i^v, y_i)\}$, where $X_i = (X_i^1, X_i^2, ..., X_i^v)$ is the $i^{th}$ example, and $y_i \in Y$ is its class label, $X_i^j$ is the instance of the $i^{th}$ example in the $j^{th}$ view. *View-based classification* aims at classifying $X^j$ into its corresponding class $y_j$, and so $f_j : X^j \to Y$ be the classifier in each view. *Multi-view-based classification* aims to train a set of classifiers $\{f_j\}$ by maximizing their consensus on the labeled data. To determine the final prediction, an aggregation mechanism is used by combining classifiers' results associated with different views. This approach is only applicable to multi-view problems when the views are individually sufficient for classification.

**Definition 1** ***(View-based classification)*** *View-based classification is defined as the process of constructing a separate classifier for each view, in which a weak KNN classifier is built with a different k, starting from one to the square root of the training set size, and then the weak classifiers are combined to form a strong classifier for the related view. When a view is expressed as $X^i$ and the classifier of this view is denoted by $f_i(X^i)$, the training procedure is repeated for different k values from 1 to $\sqrt{n}$ in the step of 1 to construct a set of weak classifiers such that $f_i(X^i) = \{f_{i,1}(X^i), f_{i,2}(X^i), ..., f_{i,\sqrt{n}}(X^i)\}$. Further, a combiner is used to form the individual models constructed for each different k value and generate a strong classifier for the related view.*

**Definition 2** ***(Multi-view based classification)*** *When each view is expressed as $X^v$, a set views can be represented as $X = (X^1, X^2, ..., X^V)$, where $v = 1, 2, ..., V$ and $V$ is the number of views. Assume that the function $f$ is used to learn the $i^{th}$ view $X^i$ of the data, which is defined as $f_i : X^i \to Y$. Thus, a set of view-based learners can be represented as $F_\alpha X = \{f_1(X^1), f_2(X^2), ..., f_v(X^v)\}$. Multi-view based classification is defined as a classification technique that combines multiple learners of the views for the classification; thus, the classification accuracy of the test dataset $D_{test}$ can be represented as $A(D_{test}) = \varepsilon_{i=1}^{V}\left(f_i\left(X^i\right)\right)$, where $\varepsilon$ is an ensemble method.*

Figure 1 presents the base structure of the proposed MVKNN method.

(i) *Data collection*: In many scientific data processing problems, raw data has been collected from multiple data sources or created by different feature extractors. Hence, the features of the domain are represented by disjoint feature sets (views), such as image-based features (view1), text-based features (view2), and network-based features (view3). Each view is typically considered to include more complete and useful information to learn the target task.

(ii) *Data preprocessing*: Data preprocessing techniques such as feature extraction, feature transformation, and feature selection may be applied to enhance the data quality as well as decrease the analysis time. In particular, a feature selection technique can be used to deal with high-dimensional data. In this study, feature extraction and feature transformation were not required since the datasets are ready-to-use. We performed four different feature selection (FS) methods: information gain, gain ratio, Pearson's correlation, and relief. They didn't provide a considerable improvement in terms of accuracy; however, they decreased the computational time of the MVKNN method. We didn't use the FS methods when comparing our results with the results presented in the previous studies [11, 14] to evaluate them under the same circumstances.

(iii) *View-based classification*: The proposed approach constructs a classifier to model the learning from each view. In this stage, each instance is classified using various numbers of nearest neighbors, instead of using a single value of $k$. Because the performance of the KNN algorithm is extremely sensitive to the parameter. When identifying the class label of a particular view, the proposed method uses a voting mechanism such that the class with the highest number of votes (1-NN, 2-NN, 3-NN, ..., $\sqrt{n}$-NN) is selected, where $n$ is the number of instances in the respective data. The maximum value of $k$ was determined as $\sqrt{n}$ as suggested in many previous studies [41–44]. If $k$ is large, the neighborhood may cover many samples from other classes; on the other hand, if $k$ is small, then the classification result can be highly sensitive to noisy samples. Therefore, the square root is a reasonable choice because the probability of overfitting dramatically increases if $k$ is determined as too large or too small. Gunarathna et al. [41] mentioned that, in most cases, it is used as the square root of the number of instances of the dataset. Park and Lee [42] stated that a good empirical rule of thumb is the setting of $k$ as the square root of the data size. Murphy et al. [43] indicated that they set the value of $k$ with the square-root of the number of instances in all cases and experimentation with the different values of $k$ did not yield any improvement. Lall and Sharma [44] presented a generalized cross-validation (GCV) score function that considers the average influence of observations for prediction at each instance and approximates the estimated squared error of prediction. By using the GCV criteria, they theoretically proved that the choice of $k = \sqrt{n}$ is the optimal selected value and it is compatible with the GCV score. For instance, with a sample size $n$ of 50 to 200, this corresponds to a selection of $k$ ranging from 7 to 14. When calculating the GCV score with the same data size, similar values of $k$ are also obtained.

(iv) *Multi-view-based classification*: In this stage, a combiner is utilized to combine the individual models constructed for each view and produce the final output related to the target concept.

Algorithm 1 presents the main steps of MVKNN. The algorithm applies the KNN search procedure using a given distance measure (i.e. Euclidean) and finds the set $N_{x,k}$, which denotes the $k$ nearest neighbors of an instance $x$, according to different views of the training dataset $D_{train}$. $N_{x,k,v}$ refers to the $k$-nearest neighbors of a data instance $x$ available in the view $v$. The class label $y$ is assigned to instance $x$ by considering the major class within these $k$ nearest neighbors ($N_{x,k,v}$). Here, $C_{x,k,v}$ contains predicted class labels for the instance $x$ obtained within $k$ most similar neighbors in the view $v$. This process is repeated for different $k$ values, starting

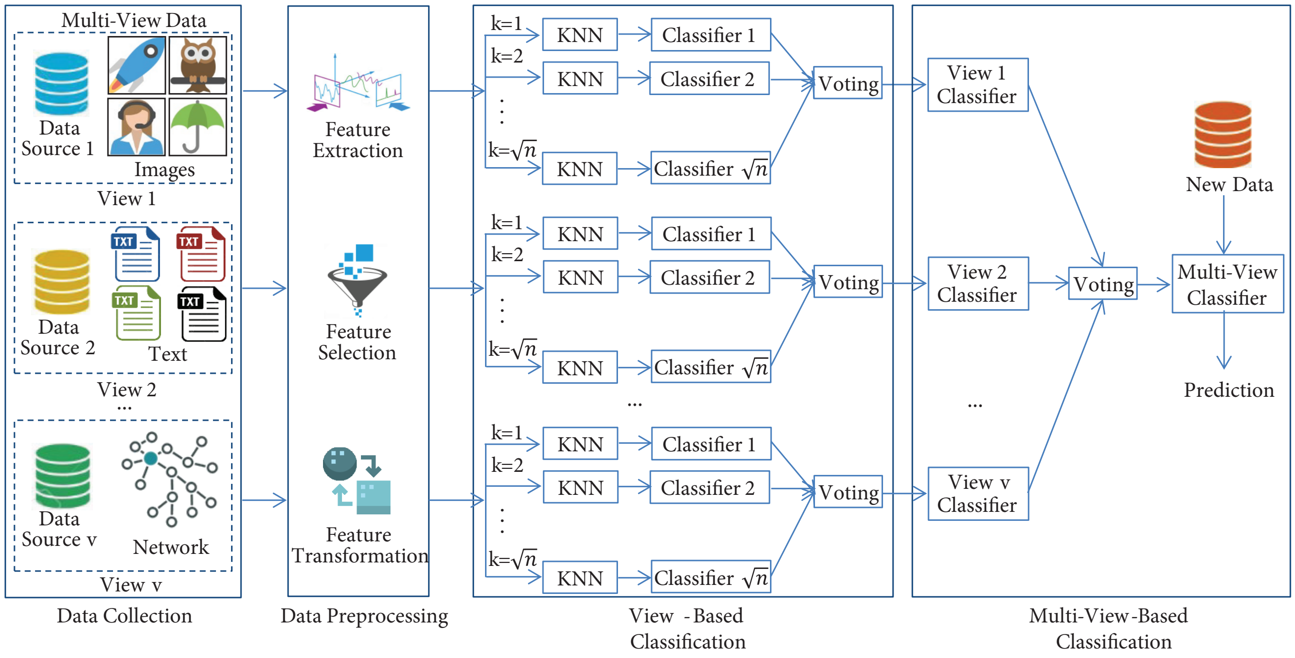


**Figure 1**. The general structure of the Multi-View KNN method.

from one to the square root of the size of the training set. After that, the second majority voting mechanism is applied to combine the individual models constructed for each different $k$ value. In this context, $C_{x,v}$ denotes the predicted class label for the input $x$ in the view $v$. Finally, view-based classifiers are combined by the third voting procedure to generate a multi-view classifier. After predicting the class of the instance $x$ (denoted by $C_x$), the whole process is repeated for other instances in the test set. The list $P$ contains all predicted class labels of the test set. For clarity, the meanings of all symbols are given in Table 2. An implementation of our method is available at the website https://github.com/elifeozturk/MVKNN.

**Algorithm 1:** Multi-view k-nearest neighbors

```
Inputs :
        $D_{train}$: Multi-view training set with $n$ instances
        $D_{test}$: Test set
Output:
        $P$: Predicted class labels of test set
foreach $x \in D_{test}$ do
    foreach view $v$ do
        for $k = 1$ to $\sqrt{n}$ do
            $N_{x,k,v} = \text{KNN}(D_{train} \in v, x, k)$
            $y = Voting(N_{x,k,v})$   // Predicted class for instance $x$ within $k$ neighbors in the view $v$
            $C_{x,k,v} = C_{x,k,v} \cup y$
        end
        $C_{x,v} = Voting(C_{x,k,v})$   // View-Based Classification
    end
    $C_x = Voting(C_{x,v})$   // Multi-view-based classification
    $P = P \cup C_x$
end
```

When classifying an instance, the time complexity of the proposed MVKNN algorithm is given by $O(v.T(n).\sqrt{n})$ where $v$ is the number of views, $T$ refers to the time needed for the execution of a traditional KNN algorithm, and $n$ is the number of instances in the dataset. As given in [45], the space complexity of the KNN algorithm is estimated to be $O(nd)$, where $n$ is the number of instances and $d$ (dimension) is the number of features in an input vector. In MVKNN, KNN is applied $\sqrt{n}$ times for each view; however, it uses the same space at each time. As MVKNN learns, it also stores the class label predictions of each view for each different input parameter. Therefore, the overall space complexity of MVKNN is estimated to be as $O(nd + p)$, where $p$ is the list of predicted class labels. It is noted here that $p$ can be ignored since its practical value is rather small compared to the entire big training data ($p \ll nd$).

**Table 2**. Notation table.

| Symbol | Description |
|---|---|
| $x$ | an instance |
| $n$ | number of instances |
| $D_{train}$ | training set with n instances |
| $D_{test}$ | test set |
| $k$ | number of nearest neighbors |
| $v$ | a view of dataset $D$ |
| $y$ | predicted class label |
| $N_{x,k}$ | $k$-nearest neighbors of instance $x$ |
| $N_{x,k,v}$ | $k$-nearest neighbors of instance $x$ in a view $v$ |
| $C_{x,k,v}$ | predicted class labels for instance $x$ within $k$ neighbors in a view $v$ |
| $C_{x,v}$ | prediction of a single view after the second majority voting (view-based classification) |
| $C_x$ | final prediction of multiple views after the third majority voting (multi-view-based classification) |
| $P$ | Predicted class labels of test set |

## 3.3. The advantages of the proposed method (MVKNN)

The proposed algorithm (MVKNN) has a number of advantages that can be summarized as follows:

- *Constructing a complete model:*
  Single-view data is highly dependent on the viewpoint and includes incomplete information while multi-view data generally includes complementary information. Even if the information included in a single-view data is complete, some undesirable noises may exist. On the contrary, MVKNN leverages an abundance of information in each view to understanding the underlying structure of the data more explicitly. A relative complete model could be constructed by combining complementary information from multiple views, when the weakness of one view is complemented by the strengths of other views. At the same time, the common pattern shared by all these views is emphasized by such a voting mechanism. This complementary property of MVKNN is able to overcome the drawbacks and limitations of single-view learning.

- *Constructing a robust model:*
  In single-view learning, the presence of noise sometimes makes the detection of patterns more difficult

and causes unsatisfactory classification performance. On the contrary, MVKNN can be able to eliminate the side-effect of noise in one view and directly emphasize the common pattern shared by multiple views. Compared with the traditional KNN algorithm, which is developed for single-view data, MVKNN is expected to yield more robust classification outputs by investigating the complementary information in all views. For example, it may fail to diagnose illness via just considering clinical data (one view); however, if more information such as MR images (another view) can be obtained, it can be possible to find out a more accurate diagnosis.

In the MVKNN method, noise within one view can be reduced through a voting mechanism among multiple views. Learning from multi-view data yields robust results since it automatically reduces the bias and variance of learning methods thanks to the ensemble approach. It is a fact that all ensemble methods are more robust compared to the methods with a single weak classifier. Furthermore, the dependency of the classifier on the properties and characteristics of a single view is eliminated.

- *Constructing a reliable model:*
  Since the different number of neighbors' values are considered and in this way multiple classifiers are built, MVKNN tends to construct a reliable model by integrating the outputs of multiple classifiers. Thanks to the ensemble approach, the MVKNN method provides a significant potential for reliable classification and yields a constructive result.

- *Achieving different tasks:*
  The MVKNN method allows us to perform not only classification tasks but also regression tasks by calculating the average of the numerical target of the $k$ nearest neighbors.

- *Extending the application areas:*
  As many multi-view classification methods, MVKNN can be used for both single-view and multi-view data. Nevertheless, many traditional classification algorithms can be implemented for single-view data only due to their limitations. The complementary property of the MVKNN algorithm on multi-view data overcomes the limitations and drawbacks of single-view data and expands the application field of the standard KNN algorithm.

- *Facilitating learning task:*
  MVKNN takes advantage of data obtained from multiple different sources. Due to the heterogeneity of data sources, it can improve the classification accuracy by transferring knowledge across the views. MVKNN offers needed flexibility and well manipulate the cases that involve heterogeneous information sources.

- *Dealing with high-dimensional data:*
  In many ML applications, objects are usually represented with high-dimensional data and often include multiple kinds of features, i.e. features of images, videos, documents, and web pages. MVKNN deals with high-dimensional data in two ways. First, as shown in Figure 1, it is designed with the feature selection step, which eliminates the irrelevant or redundant features and therefore reduces the dimension to provide efficiency. Second, high-dimensional data can be considered as a union of multiple low-dimensional subspaces, called views. In other words, data is segmented into proper views that each of them corresponds to a separate feature set. The base learning algorithm individually learns from each view data. Hence, it runs on a small portion of the data (low-dimensional data), instead of the whole data (high-dimensional data). Thus, MVKNN has the capability to deal with issues related to the high dimensionality of data.

Despite their advantages, MVKNN has also several limitations, since constructing reliable models from multi-view data is a very challenging task. First, the MVKNN method considers inter-view correlations at the ensemble level; however, it fails to explore the implicit correlations between features across multiple views. Second, although the MVKNN algorithm has a good performance, it is computationally more expensive than the traditional KNN algorithm since it individually learns from each view data and collectively learns from multiple $k$ values, instead of from a single/fixed $k$ value. It is a fact that all ensemble learning methods increase the computational cost; however, they are widely used in the machine learning community since they usually increase classification accuracy considerably. Similarly, the computational complexity of MVKNN can be ignored since it significantly improves learning performance. Furthermore, its computational cost can be easily decreased in different ways such as distributed/parallel computing, in-memory computing, feature selection, instance sampling, and dynamic resource allocation.

## 4. Experimental studies

In this article, we propose an algorithm (MVKNN) that aims to improve the generalization ability of a classification model by combining multiple views with a voting mechanism. This section presents the experimental results obtained by the application of the MVKNN algorithm on various datasets.

We conducted several experiments on twelve multi-view datasets to illustrate the efficiency and validity of the proposed MVKNN algorithm. We compared the MVKNN algorithm with the existing KNN algorithm in terms of classification accuracy, precision, recall, and F-measure. *Accuracy* is the most commonly used measure to evaluate classification performance, which is the proportion of correctly classified cases to the total number of cases. In other words, accuracy is calculated as the ratio of correctly classified examples (TP, TN) to the total examples (TP, TN, FN, FP) as given in Equation (1).

$$Accuracy = \frac{TN + TP}{FP + TN + FN + TP} \tag{1}$$

where *false positives* (FP) is the number of data examples incorrectly classified, *true positive* (TP) is the number of correctly classified positive data examples, *false negative* (FN) is the number of misclassified positive data examples, and *true negative* (TN) is the number of correctly predicted negative data examples. *Precision* is the proportion of the number of correctly predicted positive observations (TP) to the total number of positive predictions (TP, FP), and it is calculated as given in Equation (2).

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

*Recall* is the ratio of correctly predicted positive observations to all the observations in actual class. (Equation (3))

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

*F-measure* is the harmonic mean of precision and recall, which is calculated by the formula given in Equation (4) [46]:

$$Fmeasure = \frac{2 \times Precision \times recall}{Precision + recall} \tag{4}$$

We implemented the proposed method in the C# programming language and wrote the main parts of the code here such as input/output operations, data processing, the combination of the results of multiple views, and the calculation of evaluation metrics (i.e., precision, recall). We used the WEKA machine learning library [47] for running base learners and the standard machine learning algorithms such as KNN, SVM, DT, and NB. Performance results were obtained by using the 10-fold cross-validation technique, in which the data is randomly divided into ten disjoint and equal-sized partitions, and one of the partitions is kept for the testing process, while the remaining partitions are utilized for the training process. Random seed selection of this technique can be tolerated since the procedure is repeated 10 times to reduce sensitivity, and thanks to the ensemble-styled learning paradigm. As a result, the MVKNN algorithm was run 10 times with different train/test sets and the overall accuracy was calculated as the average of all 10 runs. The obtained results were validated by using statistical tests to ensure the significance of differences among the methods on the datasets. We used two well-known non-parametric statistical tests: Friedman aligned ranks and quade tests [48].

### 4.1. Dataset description

Experiments were carried out on twelve multi-view datasets obtained from UCI (University of California at Irvine) and Tera-Promise repositories. MVKNN is fundamentally designed for learning from multi-view data. However, if it is applied to single-view data, it tends to reflect a simple ensemble learning behaviour, in which a set of classifiers are built by setting different input parameter values to the same algorithm. In this case, only the first stage of the method (view-based classification) is considered; however, the second stage (multi-view-based classification) is ignored. If a single dataset conceptually has a multi-view perspective, it can be separated into views in a logical manner by an expert. Another way is to use a view generation algorithm [19] to find various feature subsets from available features that will correspond to views.

Table 3 summarizes the properties of the dataset, including the number of views (#Views), the number of classes (#Classes), the number of data instances (#Instances), and the number of features (#Features) belonging to each view. The detailed descriptions about the datasets are given below.

**Dermatology dataset:**[1] This dataset was created to diagnose the type of eryhemato-squamous disease among six classes, including psoriasis, cronic dermatitis, lichen planus, pityriasis rubra pilaris, seboreic dermatitis, and pityriasis rosea. The data consists of two views: clinical view (12 features) and histopathological view (22 features). The first view describes the clinical evaluation of the patients, whereas the second view includes the features obtained by an analysis of the skin samples of patients under a scientific microscope. All features, except from age and family history, indicate a degree in the range of 0 and 3, from nonexist to high respectively.

**Multi-Feature Digit Dataset:**[2] This dataset includes feature sets of hand-written digits (0-9) obtained from a collection of utility map. Each class (digit) has 200 samples, so the dataset has 2000 instances in total. It comprises of six feature sets (views), namely mfeat-kar (Karhunen-Love coefficients), mfeat-fou (Fourier coefficients), mfeat-fac (profile correlations), mfeat-zer (Zernike moments), mfeat-pix (pixel averages), and mfeat-mor (morphological features).

**One-Hundred Plant Species Leaves (OPSL) Dataset:**[3] This dataset contains the features about

[1]Ilter N, Guvenir HA (1998). Dermatology Dataset [online]. Website http://archive.ics.uci.edu/ml/datasets/dermatology [accessed 30 April 2020]

[2]Duin RPW (1998). Multi-Feature Digit Dataset [online]. Website http://archive.ics.uci.edu/ml/datasets/Multiple+Features [accessed 30 April 2020]

[3]Mallah C, Cope J, Orwell J (2012). One-Hundred Plant Species Leaves Dataset [online]. Website http://archive.ics.uci.

1600 leaf samples collected from one-hundred plant species. Since three feature vectors were extracted for each sample, there are three views: shape (a shape descriptor), texture (texture histogram), and margin (fine scale edge), each of which is 64 dimensional.

**Parkinson's Disease Dataset:**[4] Since parkinson disease (PD) affect speech in the early stages of the disease, speech characteristics have been successfully used in the assessment of PD. This dataset contains speech characteristics, which were collected from 188 patients and 64 healthy persons with three repetitions of vowels by using a microphone. Speech recording features were obtained by different signal processing algorithms, where each one corresponds to a view, i.e., mel-frequency cepstral coefficient (MFCC) and tunable q-factor wavelet transform (TQWT).

**SPECTF Heart DataSet:**[5] This dataset contains diagnosing of SPECT (Single Proton Emission Computed Tomography) images of cardiac patients categorized into with two classes: abnormal and normal. The data was collected in two phases (views): stress and rest.

**WISDM Smartphone and Smartwatch Activity and Biometrics Dataset:**[6] This dataset contains accelerometer and gyroscope sensor values collected from two devices (views), smartphone and smartwatch, to be able to classify 18 daily human activities such as walking, jogging, sitting, eating, drinking, and writing.

**Software Defect Datasets:**[7] The datasets, namely Log4j, Jedit, Poi, Redaktor, Velocity, and Xerces, were obtained from the Tera-Promise repository. Each dataset consists of 20 independent object-oriented software metrics and one dependent defect variable that indicates the source code is buggy or not. As explained in the previous study [49], each dataset contains code metrics that indicate the characteristics of cohesion, complexity, coupling, scale, and inheritance features in software programs. Therefore, each dataset has five views.

## 4.2. Experimental results

To demonstrate the effectiveness of MVKNN, we compared it with well-known machine learning algorithms (KNN, SVM, DT, and NB) (Section 4.2.1) and also compared with the state-of-the-art multi-view learning methods (MV-LSSVM [11], MVL-KNN [12], MVL-LS [13], and MVDT [14]) (Section 4.2.2).

### 4.2.1. Comparison with existing methods

We conducted various experiments on twelve multi-view datasets explained in the previous section and obtained empirical results. Since MVKNN is a KNN-based method, we especially compared it with the KNN algorithm in terms of classification accuracy, precision, recall, and F-measure. In addition, we also compared MVKNN with other well-known machine learning techniques such as support vector machine (SVM), decision tree (DT), and naive Bayes (NB). Table 4 presents the comparison results in terms of accuracy (%). It is clearly seen that, for all the datasets, the MVKNN algorithm is better in classification compared to the KNN algorithm. Similarly, MVKNN has higher accuracy than SVM, DT, and NB on average for rest of the datasets. This is because of

`edu/ml/datasets/One-hundred+plant+species+leaves+data+set` [accessed 30 April 2020]

[4]Sakar CO et al. (2018). Parkinson's Disease Dataset [online]. Website `http://archive.ics.uci.edu/ml/datasets/Parkinson%27s+Disease+Classification` [accessed 30 April 2020]

[5]Cios KJ, Lukasz AK (2001). SPECTF Heart DataSet [online]. Website `http://archive.ics.uci.edu/ml/datasets/SPECTF+Heart` [accessed 30 April 2020]

[6]Weiss G (2019). WISDM Smartphone and Smartwatch Activity and Biometrics Dataset [online]. Website `https://archive.ics.uci.edu/ml/datasets/WISDM+Smartphone+and+Smartwatch+Activity+and+Biometrics+Dataset+` [accessed 30 April 2020]

[7]Tantithamthavorn C (2015). Tera-Promise Dataset [online]. Website `https://github.com/klainfo/DefectData` [accessed 6 January 2021]

**Table 3**. Main characteristics of the datasets.

| Dataset Name | #Views | #Classes | #Instances | #Features | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dermatology | 2 | 6 | 366 | Clinical | | | Histopathological | | |
| | | | | 12 | | | 22 | | |
| Jedit | 5 | 2 | 312 | Cohesion | Complexity | Coupling | Scale | Inheritance | |
| | | | | 3 | 3 | 5 | 6 | 3 | |
| Log4j | 5 | 2 | 109 | Cohesion | Complexity | Coupling | Scale | Inheritance | |
| | | | | 3 | 3 | 5 | 6 | 3 | |
| Multi-Feature Digit | 6 | 10 | 2,000 | Mfeat-fac | Mfeat-pix | Mfeat-mor | Mfeat-kar | Mfeat-fou | Mfeat-zer |
| | | | | 216 | 240 | 6 | 64 | 76 | 47 |
| OPSL | 3 | 100 | 1,600 | Margin | | Shape | | Texture | |
| | | | | 64 | | 64 | | 64 | |
| Parkinson's Disease | 6 | 2 | 756 | Baseline | Time Freq. | MFCC | Vocal Fold | Wavelet | TQWT |
| | | | | 21 | 11 | 84 | 22 | 182 | 432 |
| Poi | 5 | 2 | 442 | Cohesion | Complexity | Coupling | Scale | Inheritance | |
| | | | | 3 | 3 | 5 | 6 | 3 | |
| Redaktor | 5 | 2 | 176 | Cohesion | Complexity | Coupling | Scale | Inheritance | |
| | | | | 3 | 3 | 5 | 6 | 3 | |
| SPECTF Heart | 2 | 2 | 267 | Stress | | | Rest | | |
| | | | | 22 | | | 22 | | |
| Velocity | 5 | 2 | 196 | Cohesion | Complexity | Coupling | Scale | Inheritance | |
| | | | | 3 | 3 | 5 | 6 | 3 | |
| WISDM | 2 | 18 | 306 | Phone | | | Watch | | |
| | | | | 86 | | | 86 | | |
| Xerces | 5 | 2 | 453 | Cohesion | Complexity | Coupling | Scale | Inheritance | |
| | | | | 3 | 3 | 5 | 6 | 3 | |

the fact that MVKNN takes into account the complementary and consistency characteristics of different views, unlike single-view classification. For instance, MVKNN (95.45%) significantly outperformed KNN (85.03%), SVM (88.18%), DT (80.02%), and NB (82.90%) on the multi-feature digit dataset. In particular, the biggest accuracy difference of MVKNN and the rest was observed on the OPSL dataset, where MVKNN increased the accuracy by over 23% against KNN. It is followed by the poi (18.55%) and multi-feature digit (15.43%) datasets against NB and DT. The common characteristic of the OPSL and multi-feature digit datasets is that the number of instances is higher than the others. This means that the improvement in the performance of MVKNN increases as the number of instances increases. When the number of classes is taken into consideration, the MVKNN method achieved improvement compared to the KNN, SVM, DT, and NB methods in the datasets with a large number of classes, such as the OPSL dataset and the WISDM dataset. The MVKNN method achieved the best accuracy (96.73%) on the WISDM dataset. This means that the model can recognize daily human activities very correctly. The experimental studies showed that the MVKNN algorithm achieved 84.82% accuracy on average, while the KNN, SVM, DT, and NB algorithms reached only 79.31%, 80.44%, 80.86%, and 77.11% accuracy, respectively. Therefore, it can be concluded that, compared to the traditional single view classification algorithms (SVM, DT, NB, and KNN), the proposed method yields better results.

In addition to accuracy, we also compared the performances of the algorithms in terms of precision, recall, and F-measure as given in Table 5, Table 6, and Table 7, respectively. The values of these metrics are ranged between 0 and 1, where 1 is the best value. In other words, the higher the value, the better the performance. As seen from Table 5; the precision values obtained by the MVKNN algorithm are closer to 1 than KNN,

**Table 4**. Comparison of MVKNN and existing methods (SVM, DT, NB, KNN) on various datasets in terms of accuracy (%).

| Dataset | Accuracy (%) | | | | |
|---|---|---|---|---|---|
| | SVM | DT | NB | KNN | MVKNN |
| Dermatology | **91.12** | 87.16 | 90.71 | 83.06 | 85.79 |
| Jedit | 75.38 | 79.17 | 77.05 | 78.14 | **81.73** |
| Log4j | 73.58 | 74.86 | 74.86 | 75.41 | **80.73** |
| Multi-Feature Digit | 88.18 | 80.02 | 82.90 | 85.03 | **95.45** |
| OPSL | 67.85 | 76.90 | 71.23 | 56.44 | **79.94** |
| Parkinson's Disease | 77.91 | 76.94 | 65.96 | 78.22 | **78.97** |
| Poi | 68.33 | 76.47 | 62.22 | 75.93 | **80.77** |
| Redaktor | 85.80 | 86.70 | 82.16 | 87.27 | **89.20** |
| SPECTF Heart | **79.40** | 74.54 | 65.17 | 75.46 | 77.90 |
| Velocity | 78.16 | 81.12 | 76.84 | 80.92 | **82.14** |
| WISDM | 94.78 | 90.52 | 91.99 | 90.03 | **96.73** |
| Xerces | 84.77 | 85.96 | 84.28 | 85.83 | **88.52** |
| Avg. | 80.44 | 80.86 | 77.11 | 79.31 | **84.82** |

SVM, and DT algorithms for 11 out of 12 datasets. This indicates that MVKNN often yields better results than the rest. It is clearly seen that the MVKNN algorithm outperformed all other algorithms (KNN, SVM, DT, NB) on average. The same achievement of MVKNN was also observed for recall and F-measure metrics on average. For example, MVKNN (0.97) outperformed KNN (0.91), SVM (0.94), DT (0.91), and NB (0.92) on the WISDM dataset in terms of F-measure. In particular, the biggest F-measure difference of MVKNN and KNN was observed on the OPSL dataset, where MVKNN increased the performance by over 0.2. The experimental studies showed that the MVKNN algorithm achieved the F-measure value of 0.85 on average, while the KNN algorithm reached only the value of 0.79. According to the results, it can be concluded that the MVKNN algorithm is better in classification compared to the rest in terms of accuracy, precision, recall, and F-measure.

It experimentally confirmed that the characteristics of the data have an important impact on the classification performance of the algorithm. For instance, the MVKNN algorithm has a lower precision value (0.66) for the SPECTF heart dataset compared to the other datasets, like the SVM algorithm (0.63). This is probably because of two reasons: small data size and imbalanced data. When the dataset size is small, the insufficient data may not represent patterns in the data. This is especially true when some classes are similar to each other, where small-sized data may not contain useful information for distinguishing them. The SPECTF heart dataset is also relatively imbalanced, where one of the classes has 212 intances (80%) while the other class has only 55 instances (20%). When the dataset is imbalanced, the instances from the minority class do not exist in data enough for learning. The presence of redundant or irrelevant features may also mislead the algorithm to make incorrect predictions. Furthermore, a model cannot give good results, if the training data does not represent patterns in the data. Performance declines could be observed when there was a mismatch between the classes available in the training set and the classes present in the test set.

Although the MVKNN method has generally higher accuracy, the obtained results should be validated by the statistical tests to determine whether the differences in performance are statistically significant or not.

**Table 5**. Comparison of MVKNN and existing methods (SVM, DT, NB, KNN) on various datasets in terms of precision.

| Dataset | Precision | | | | |
|---|---|---|---|---|---|
| | SVM | DT | NB | KNN | MVKNN |
| Dermatology | 0.90 | 0.87 | **0.91** | 0.86 | 0.88 |
| Jedit | 0.72 | 0.78 | 0.75 | 0.77 | **0.83** |
| Log4j | 0.73 | 0.74 | 0.75 | 0.76 | **0.82** |
| Multi-Feature Digit | 0.88 | 0.79 | 0.83 | 0.85 | **0.96** |
| OPSL | 0.69 | 0.48 | 0.73 | 0.60 | **0.85** |
| Parkinson's Disease | 0.73 | 0.75 | 0.75 | 0.76 | **0.81** |
| Poi | 0.69 | 0.77 | 0.71 | 0.77 | **0.81** |
| Redaktor | 0.86 | 0.85 | 0.80 | 0.85 | **0.88** |
| SPECTF Heart | 0.63 | 0.74 | **0.83** | 0.73 | 0.66 |
| Velocity | 0.78 | 0.80 | 0.73 | 0.80 | **0.84** |
| WISDM | 0.94 | 0.91 | 0.93 | 0.92 | **0.97** |
| Xerces | 0.85 | 0.84 | 0.81 | 0.81 | **0.89** |
| Avg. | 0.78 | 0.78 | 0.79 | 0.79 | **0.85** |

**Table 6**. Comparison of MVKNN and existing methods (SVM, DT, NB, KNN) on various datasets in terms of recall.

| Dataset | Recall | | | | |
|---|---|---|---|---|---|
| | SVM | DT | NB | KNN | MVKNN |
| Dermatology | **0.93** | 0.87 | 0.91 | 0.83 | 0.86 |
| Jedit | 0.75 | 0.79 | 0.77 | 0.78 | **0.82** |
| Log4j | 0.74 | 0.75 | 0.75 | 0.75 | **0.81** |
| Multi-Feature Digit | 0.88 | 0.80 | 0.83 | 0.85 | **0.95** |
| OPSL | 0.68 | 0.47 | 0.71 | 0.56 | **0.80** |
| Parkinson's Disease | 0.78 | 0.77 | 0.66 | 0.78 | **0.79** |
| Poi | 0.68 | 0.76 | 0.62 | 0.76 | **0.81** |
| Redaktor | 0.86 | 0.87 | 0.82 | 0.87 | **0.89** |
| SPECTF Heart | **0.79** | 0.75 | 0.65 | 0.76 | 0.78 |
| Velocity | 0.78 | 0.81 | 0.77 | 0.81 | **0.82** |
| WISDM | 0.94 | 0.91 | 0.92 | 0.90 | **0.97** |
| Xerces | 0.85 | 0.86 | 0.84 | 0.86 | **0.89** |
| Avg. | 0.81 | 0.78 | 0.77 | 0.79 | **0.85** |

In a classical statistical test, the null hypothesis (H0) is that there are no performance differences among the methods on the datasets; otherwise, another hypothesis (H1) is present when there are significant performance differences among the methods. The p-value is defined as the probability under the null hypothesis of obtaining results and we reject the null hypothesis (H0) with a small p-value (p-value $\leq$ 0.05), hence we prove that the difference between the results is significant. For verification, we used two well-known non-parametric statistical tests by one-vs-all comparisons: Friedman aligned ranks test and quade test [48]. The p-values obtained from

**Table 7**. Comparison of MVKNN and existing methods (SVM, DT, NB, KNN) on various datasets in terms of F-measure.

| | F-measure | | | | |
|---|---|---|---|---|---|
| Dataset | SVM | DT | NB | KNN | MVKNN |
| Dermatology | **0.91** | 0.87 | **0.91** | 0.84 | 0.87 |
| Jedit | 0.74 | 0.79 | 0.76 | 0.77 | **0.82** |
| Log4j | 0.73 | 0.75 | 0.75 | 0.75 | **0.81** |
| Multi-Feature Digit | 0.88 | 0.79 | 0.83 | 0.85 | **0.95** |
| OPSL | 0.68 | 0.47 | 0.71 | 0.58 | **0.82** |
| Parkinson's Disease | 0.75 | 0.76 | 0.67 | 0.77 | **0.80** |
| Poi | 0.68 | 0.77 | 0.67 | 0.76 | **0.81** |
| Redaktor | 0.86 | 0.86 | 0.81 | 0.86 | **0.88** |
| SPECTF Heart | 0.70 | **0.74** | 0.68 | **0.74** | 0.72 |
| Velocity | 0.78 | 0.81 | 0.75 | 0.80 | **0.83** |
| WISDM | 0.94 | 0.91 | 0.92 | 0.91 | **0.97** |
| Xerces | 0.85 | 0.85 | 0.82 | 0.83 | **0.89** |
| Avg. | 0.79 | 0.78 | 0.77 | 0.79 | **0.85** |

these statistical tests are 0.00128 and 0.00290, respectively. Thereby, it is possible to say that the results are statistically significant since all p-values are smaller than the significance threshold (0.05).

We also evaluated the algorithms for each view individually. Figure 2 shows the view-based results. The results show that the MVKNN algorithm outperforms the KNN algorithm when each view is considered separately. For instance, in the case of dermatology disease, the MVKNN method achieved 81.69% and 90.16% accuracy rates for each view separately; however, the KNN method reached only 78.14% and 87.98% performance values for the views respectively. It can be noted that the classification accuracies of the algorithms are independent of the number of views. No technique has yet been discovered to determine prior information about how many views are appropriate for any particular problem. Each view, each problem, even each dataset, has its own characteristics. Therefore, a combination of empirical evaluation and theoretical analysis should be used to determine the best situation for the given problem.

Figure 3 shows the precision, recall, and F-measure evaluations for each view separately. From this figure, we observe that the MVKNN algorithm has usually better performance than the KNN algorithm for all metrics for all datasets, except just a few views. For instance, in the case of dermatology disease, the MVKNN method achieved the F-measure values of 0.83 and 0.91 for each view individually; however, the KNN method reached only 0.8 and 0.89 values for the views, respectively. The MVKNN method achieved the best precision, recall, and F-measure values on the WISDM dataset, where all of them are higher than 0.9.

In order to take full advantage of multiple views, our proposed approach considers two main principles: the complementary and consensus principles. With the *consensus principle*, it maximizes the agreement on multiple views since minimizing the disagreement among the views reduces their individual error rates. With the *complementarity principle*, it improves the learning performance by comprehensively utilizing information from multiple views since each view of the data may contain an important information non-exist in other views. As a result, both consensus and complementary principles play important roles in our algorithm.

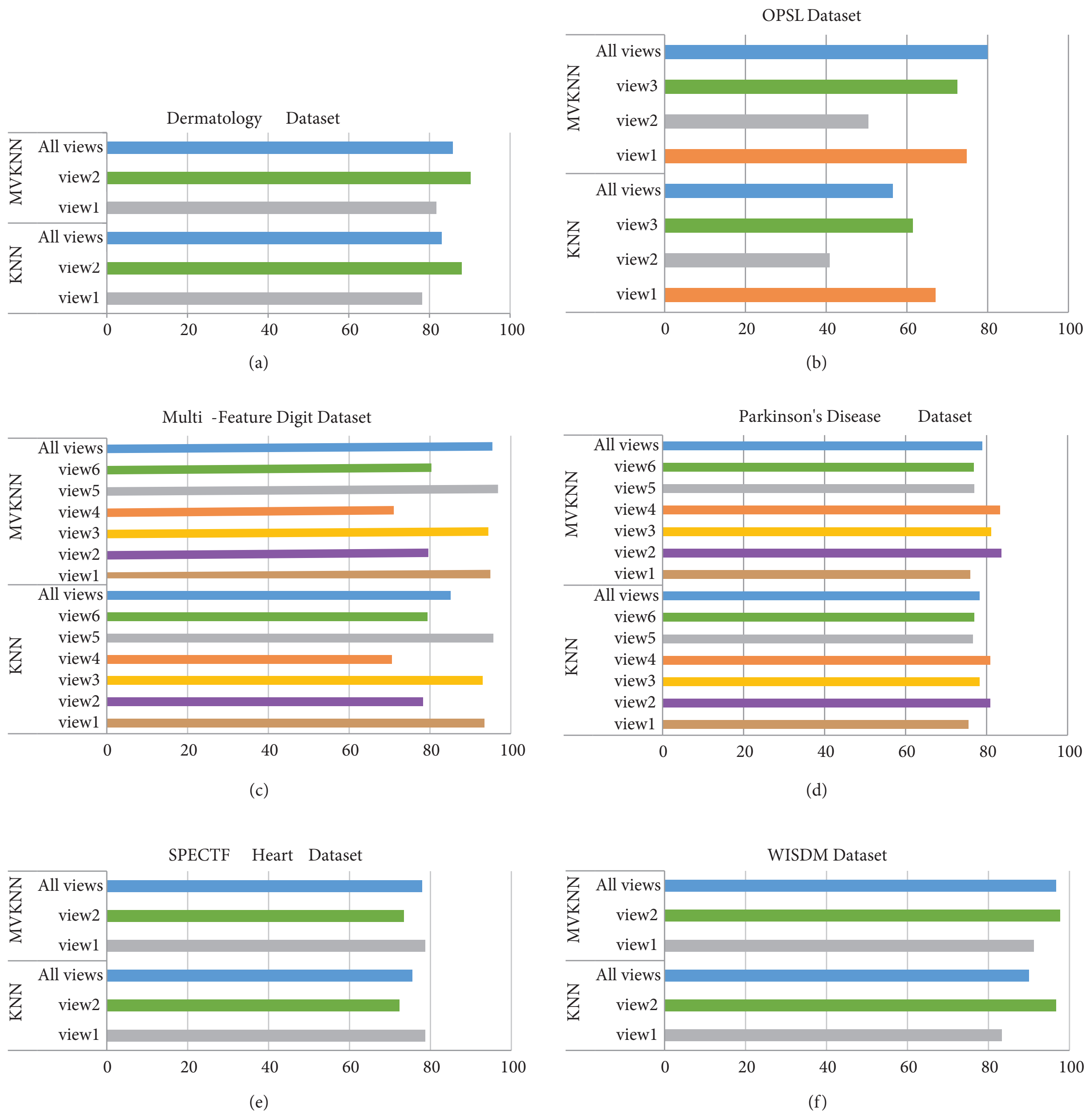


**Figure 2**. View-based comparison of the MVKNN and KNN algorithms in terms of accuracy (%).

The basic assumption of the conventional KNN algorithm is that a data instance is assigned to the same class as the majority of its $k$ nearest neighbors, where $k$ is fixed for all data instances to be classified. Nevertheless, many datasets have an imbalanced and irregular distribution of data instances. Due to the density variations, we considered different $k$ parameters for data instances; e.g., a simple intuition would be to benefit from more neighbors in dense areas and less in the sparse areas. In this article, we address this issue by proposing an ensemble-based classification, in which the training procedure is repeated for different $k$ values from 1 to $\sqrt{n}$, where $n$ is the number of data instances. The maximum value of $k$ was selected as $\sqrt{n}$ based on many

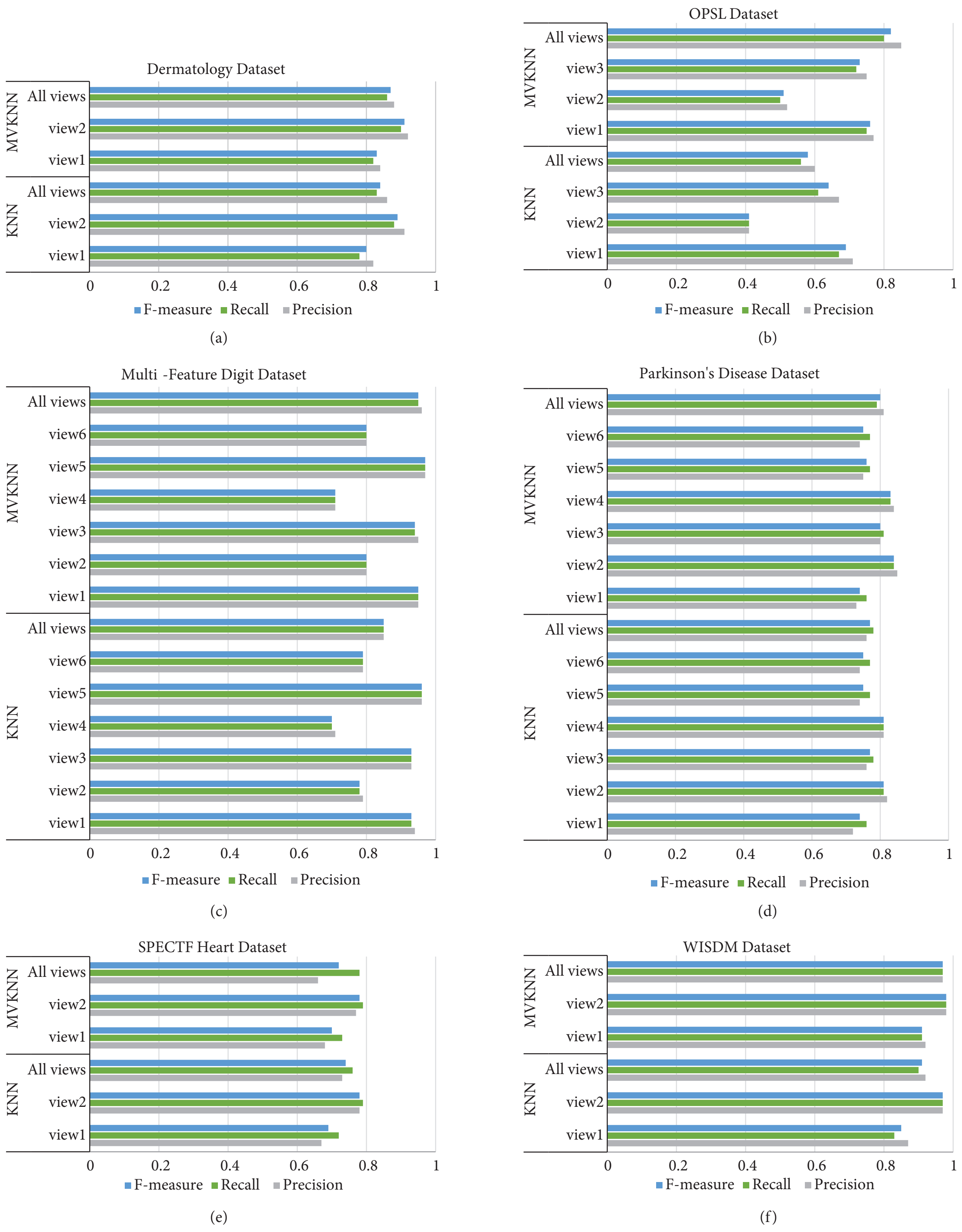


**Figure 3**. View-based comparison of the MVKNN and KNN algorithms in terms of precision, recall, and F-measure.

studies [41–44]. The maximum $k = \sqrt{n}$ is a reasonable choice because if $k$ is determined as larger than this, the neighborhood may cover many samples from other classes. Park and Lee [42] stated that a good empirical rule of thumb is the setting of $k$ as the square root of the data size. Lall and Sharma [44] have also theoretically proven this by using a generalized cross-validation (GCV) score function.

Instead of only using a single and fixed $k$ value, the algorithms were tested with various $k$ values to explore the impact of the input parameter on the results. Figure 4 shows the accuracy results obtained with different $k$ values ranging from 1 to $\sqrt{n}$ for each dataset separately. Hence, the effect of the $k$ parameter can be analyzed at each step distinctly. It is clearly observed that the MVKNN algorithm significantly performed better than the KNN algorithm for almost all $k$ values for all datasets. Only in the dermatology dataset, there is uncertainty between two methods when $k \leq 7$. The higher number of $k$ values usually results in low classification performance since the dataset sizes are a little bit small and the number of classes is relatively large. Although the accuracies of the KNN algorithm sometimes decreases as the $k$ value increases, there is not such a dramatic decline in the MVKNN algorithm results. The MVKNN method usually shows small fluctuations. The gap between the MVKNN and KNN sometimes increases; however, sometimes tends to be close. It can be also noted that the MVKNN method has the ability to reach the best accuracy in the early stages.

In addition to the accuracy metric, we compared the algorithms in terms of F-measure with various $k$ values. Figure 5 shows the F-measure values obtained with different $k$ values ranging from 1 to $\sqrt{n}$ for each dataset, separately. It presents only the F-measure since it is an informative evaluation metric that involves both precision and recall. From Figure 5, it can be seen that the MVKNN algorithm significantly performed better than the KNN algorithm for almost all $k$ values for 5 out of 6 datasets. The MVKNN method usually showed small fluctuations. It can be also noted that the MVKNN method has the ability to reach the best F-measure value in the early stages.

Robustness of the classification models with respect to noise is a desirable property since some noise is often present in data and might affect the learning process. In this experiment, we observed the impact of noise on the performances of the KNN and MVKNN methods. We added noise to the datasets at varying levels. In other words, some data instances were chosen randomly at the rate of $p\%$ and their class labels were changed from their current value to one of the other possible ones, also randomly. Table 8 shows the classification performances of KNN and MVKNN at three noise levels (2%, 6%, and 10%). Due to the close results, intermediate levels are not shown. From this analysis, we can conclude that MVKNN is still better than KNN since MVKNN has higher accuracy values compared to KNN in all datasets at all noise levels. This is because of two main reasons. First, in the MVKNN method, noise within one view can be reduced through a voting mechanism among multiple views thanks to the ensemble approach. As expected, multiple classifiers increase robustness by combining the decisions of weak classifiers. Second, MVKNN learns from multiple $k$ values (the number of neighbors), instead of from a single/fixed $k$ value. Therefore, MVKNN can be able to eliminate the side-effect of noise in one $k$ value and directly emphasize the common pattern shared by multiple $k$ values.

Table 9 shows the execution times (in seconds) of the KNN and MVKNN methods for each view individually. The experiments were performed on a laptop with the following configuration: a 2.4 GHz quad-core processor and 12 GB of RAM. As can be seen from the results, MVKNN takes more time than KNN since it adopts ensemble styled learning paradigm. MVKNN individually learns from each view data using multiple $k$ values (from 1 to $\sqrt{n}$), instead of a single/fixed $k$ value. Although the difference between algorithms in small

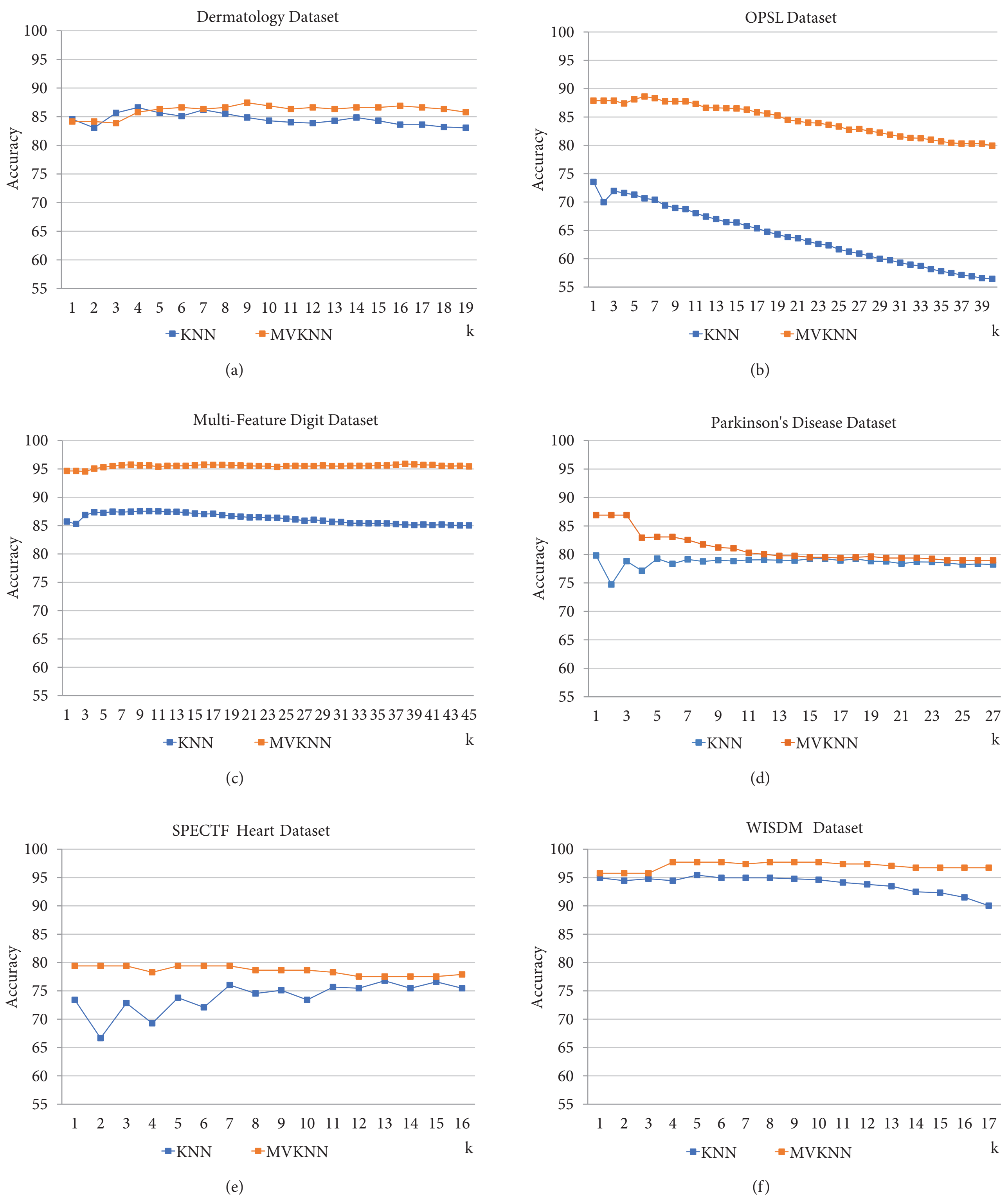
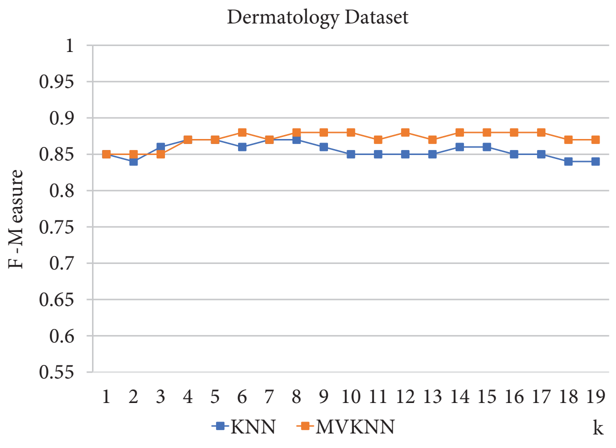
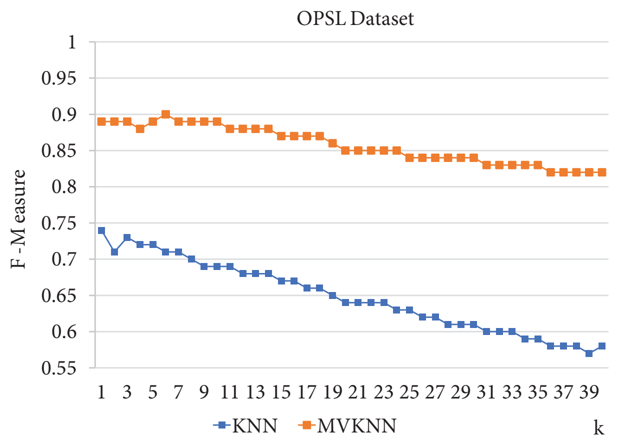


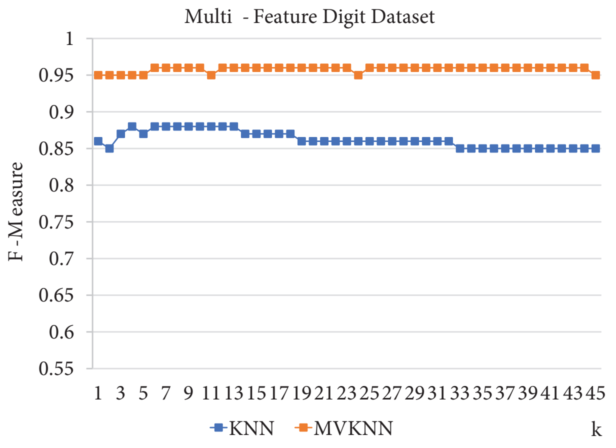
**Figure 4**. Comparison of the MVKNN and KNN algorithms on various $k$ values in terms of accuracy (%).



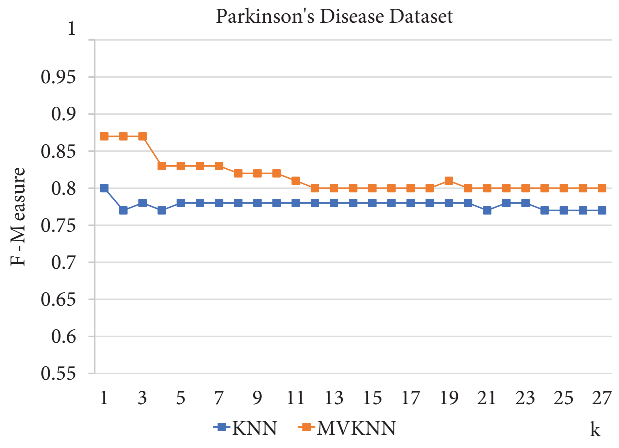
(a)



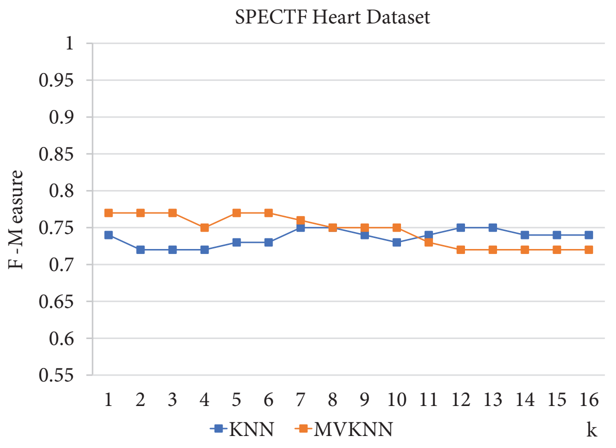
(b)



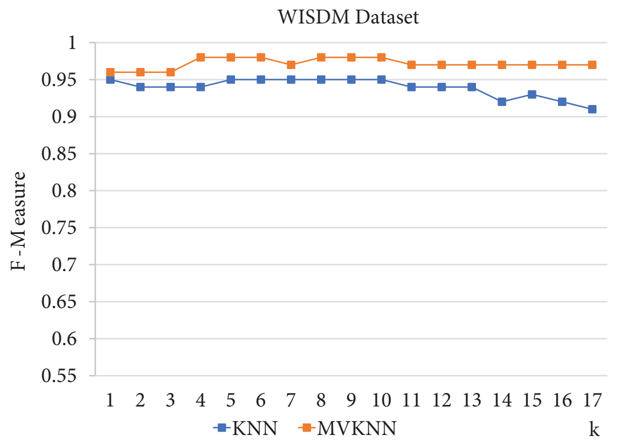
(c)



(d)



(e)



(f)

**Figure 5**. Comparison of the MVKNN and KNN algorithms on various $k$ values in terms of F-measure.

datasets is low (e.g., 1–3 s), it increases as the data size increases. It is a fact that all ensemble learning methods increase the computational cost; however, they are widely used in the machine learning community since they usually increase classification accuracy considerable. Similarly, the computational complexity of MVKNN can be ignored since it significantly improves the learning performance. Furthermore, its computational cost can be easily decreased in different ways such as distributed/parallel computing, in-memory computing, feature selection, instance sampling, and dynamic resource allocation.

**Table 8**. Comparison of MVKNN and KNN in terms of accuracy (%) at three noise levels (2%, 6%, and 10%).

| Noise | Dermatology | | Multi-feature digit | | OPSL | | Parkinson's disease | | SPECTF | | WISDM | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Percent (%) | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN |
| Original | 83.06 | **85.79** | 85.03 | **95.45** | 56.44 | **79.94** | 78.22 | **78.97** | 75.46 | **77.90** | 90.03 | **96.73** |
| 2% | 83.20 | **83.88** | 83.10 | **93.95** | 56.44 | **79.94** | 77.01 | **77.91** | 75.09 | **76.78** | 90.03 | **96.73** |
| 6% | 77.32 | **80.33** | 80.18 | **90.00** | 55.17 | **74.06** | 74.01 | **74.34** | 73.04 | **73.41** | 82.03 | **91.18** |
| 10% | 74.59 | **75.41** | 76.73 | **85.90** | 48.35 | **68.19** | 72.22 | **73.54** | 68.36 | **71.91** | 79.42 | **85.29** |
| Avg. | 78.37 | **79.87** | 80.00 | **89.95** | 53.32 | **74.06** | 74.41 | **75.26** | 72.16 | **74.03** | 83.82 | **91.07** |
| Noise | Log4j | | Poi | | Jedit | | Xerces | | Redaktor | | Velocity | |
| Percent (%) | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN |
| Original | 75.41 | **80.73** | 75.93 | **80.77** | 78.14 | **81.73** | 85.83 | **88.52** | 87.27 | **89.20** | 80.92 | **82.14** |
| 2% | 73.76 | **77.06** | 74.89 | **77.38** | 76.73 | **79.81** | 84.11 | **86.53** | 86.02 | **89.2** | 79.39 | **82.14** |
| 6% | 71.56 | **77.06** | 71.22 | **74.43** | 73.85 | **77.56** | 81.85 | **83.44** | 83.98 | **86.36** | 76.12 | **77.04** |
| 10% | 68.44 | **71.56** | 69.64 | **73.98** | 71.67 | **75.32** | 79.07 | **80.35** | 81.48 | **82.39** | 72.86 | **75.00** |
| Avg. | 72.29 | **76.60** | 72.92 | **76.64** | 75.10 | **78.61** | 82.72 | **84.71** | 84.69 | **86.79** | 77.32 | **79.08** |

**Table 9**. Execution times of the MVKNN and KNN methods in seconds per each view.

| | Dermatology | | SPECTF | | WISDM | | OPSL | | Parkinson's | | Multi-Feature | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN | KNN | MVKNN |
| view1 | 1.89 | 3.01 | 1.88 | 2.38 | 2.07 | 5.25 | 6.22 | 288.50 | 2.14 | 14.81 | 25.75 | 1625.05 |
| view2 | 0.12 | 4.29 | 0.07 | 2.09 | 0.23 | 5.38 | 3.80 | 213.78 | 0.24 | 11.68 | 30.92 | 2304.86 |
| view3 | | | | | | | 5.33 | 343.23 | 1.77 | 82.59 | 0.90 | 74.14 |
| view4 | | | | | | | | | 0.46 | 20.89 | 9.45 | 707.60 |
| view5 | | | | | | | | | 2.91 | 119.28 | 11.60 | 839.61 |
| view6 | | | | | | | | | 7.94 | 340.21 | 5.73 | 416.17 |

### 4.2.2. Comparison with state-of-the-art methods

In order to demonstrate the effectiveness of our method, we compared it with the state-of-the-art multi-view learning methods (MV-LSSVM [11], MVL-KNN [12], MVL-LS [13], and MVDT [14]) mentioned in Section 2. Since the researchers used the same dataset as our study, the results were taken directly from the referenced studies [11, 14].

***MVL-KNN***: Our method was compared with the method proposed by Liang et al. [12], which is a KNN-based multi-view study, on the same datasets. Table 10 shows the comparison results in terms of accuracy, precision, recall, and F-measure. It can be seen that MVKNN (85.80%) significantly outperformed the existing method (77.12%) in classification on average. For all the evaluation metrics, the proposed MVKNN algorithm is better than the other algorithm in 4 out of 6 datasets. For example, our method (77.90%) achieved better performance than the existing method (70.41%) in the SPECTF Heart dataset in terms of accuracy. In particular, the biggest accuracy differences between the methods were observed on the multi-feature digit and Parkinson's disease datasets, where our method increased the accuracy by over 30% and 22%, respectively. The common characteristic of these two datasets is that the number of views is higher than the others. This means that the improvement in the performance of MVKNN compared to the existing method increases as the number of views increases.

**Table 10**. Comparison of the proposed MVKNN method with the existing multi-view learning method [12] in terms of accuracy, precision, recall, and F-measure.

| Dataset | Accuracy | | Precision | | Recall | | F-measure | |
|---|---|---|---|---|---|---|---|---|
| | MVL-KNN | MVKNN | MVL-KNN | MVKNN | MVL-KNN | MVKNN | MVL-KNN | MVKNN |
| Dermatology | **87.43** | 85.79 | 0.88 | 0.88 | **0.87** | 0.86 | **0.88** | 0.87 |
| Multi-Feature | 63.65 | **95.45** | 0.95 | **0.96** | 0.94 | **0.95** | 0.95 | **0.96** |
| OPSL | **88.75** | 79.94 | **0.89** | 0.85 | **0.89** | 0.80 | **0.89** | 0.83 |
| Parkinson's | 56.08 | **78.97** | 0.56 | **0.81** | 0.56 | **0.79** | 0.56 | **0.80** |
| SPECTF | 70.41 | **77.90** | **0.69** | 0.66 | 0.70 | **0.78** | 0.70 | **0.72** |
| WISDM | 96.41 | **96.73** | 0.96 | **0.97** | 0.96 | **0.97** | 0.96 | **0.97** |
| Avg. | 77.12 | **85.80** | 0.82 | **0.86** | 0.82 | **0.86** | 0.80 | **0.86** |

Our method has advantages over the existing method [12] in three aspects. First, Liang et al. find $k$ nearest neighbors of a test sample for each view, then the intersection of nearest records from all views yields a reference set. Their algorithm tends to increase the error rate when the $k$ value is small and when the number of views is high. This is mainly because in these cases there are often no reference instances can be found due to the intersection between different views [12]. However, our method has not such a limitation about the number of neighbors and the number of views. Second, they use a single $k$ value for each view that cannot be sufficient to obtain high classification accuracy. In contrast, we use many different $k$ values ranging from 1 to $\sqrt{n}$ for each view and for each instance to improve classification results. Hence, our method decreases the sensitivity of the algorithm to changes in the model parameter. Third, we adapt the traditional KNN algorithm without any modification in its structure; however, they modified the standard KNN algorithm, which leads to implementation difficulties.

***MV-LSSVM and MVL-LS***: These two SVM-based multi-view learning methods were compared with MVKNN on the multi-feature digit dataset. Since 2 out of 6 views from the dataset were used in these studies, the same two views were also considered in our study to evaluate under the same circumstances. In other words, in this experiment, the dataset consists of 2000 digits and two views which are profile correlations (mfeat-fac) and Fourier coefficients (mfeat-fou). Similar to their studies, we applied the 5-fold cross validation technique. Table 11 shows the comparison results in terms of accuracy (%). It is clearly seen that MVKNN (87.5%) outperforms the existing SVM-based methods: MV-LSSVM (75.92%) and MVL-LS (83.87%).

**Table 11**. Comparison of MVKNN and SVM-based multi-view learning studies in terms of accuracy (%).

| Dataset | MV-LSSVM | MVL-LS | MVKNN |
|---|---|---|---|
| Multi-Feature Digit | 75.92 | 83.87 | **87.50** |

***MVDT***: The results obtained with the implementation of MVDT with various tree algorithms (C4.5, CART, TEIM, SCD, NBTree) were reported in [14]. Table 12 shows the performance comparison of the MVDT and MVKNN methods on the views of the multi-feature digit dataset in terms of accuracy. The results show that MVKNN has better accuracy than the rest in all views. When analyzed for each view separately, the best improvement (over 10%) was observed against MVDT-SCD. The best accuracy result (96.85%) was obtained by our MVKNN method in the mfeat-pix view. It is followed by the mfeat-fac view with an accuracy of 94.95%. The common characteristic of these two views is that the number of features is higher than the others (mfeat-fac - 216 features) and (mfeat-pix - 240 features). The lowest classification accuracy (71.05%) was obtained in the view with the lowest feature number (mfeat-mor - 6 features). Hence, it can be concluded that as the number of features increases, the accuracy value can increase.

**Table 12**. Comparison of MVKNN and MVDT with different tree algorithms in terms of accuracy (%).

| Views | MVDT-C4.5 | MVDT-CART | MVDT-TEIM | MVDT-SCD | MVDT-NBTree | MVKNN |
|---|---|---|---|---|---|---|
| mfeat-fac | 85.37 | 89.14 | 91.50 | 84.82 | 93.65 | **94.95** |
| mfeat-fou | 77.53 | 78.82 | 77.35 | 77.04 | 78.35 | **79.60** |
| mfeat-kar | 87.91 | 87.87 | 88.54 | 86.57 | 91.55 | **94.45** |
| mfeat-mor | 67.31 | 67.85 | 68.35 | 66.86 | 70.55 | **71.05** |
| mfeat-pix | 89.37 | 89.27 | 91.94 | 89.29 | 92.58 | **96.85** |
| mfeat-zer | 71.54 | 71.98 | 73.13 | 70.96 | 73.81 | **80.35** |
| Avg. | 79.84 | 80.82 | 81.80 | 79.26 | 83.42 | **86.21** |

## 5. Conclusion and feature work

Standard classification algorithms, such as KNN, are basically working on single-view data. However, many classification problems involve data with multiple views where the input feature space contains multiple feature vectors. In this case, the traditional algorithms like KNN simply concatenate all multiple views into a single view for learning. Nevertheless, such a simple view-concatenation strategy may generate undesirable classification results since each view has a specific characteristic. Therefore, MVL methods are needed to simultaneously explore and learn diverse information from multiple feature sets and improve learning performance by considering the diversity of different views.

To accomplish this goal, this article proposes an algorithm: Multi-view k-nearest neighbors (MVKNN) which has two main stages. In the first stage, the algorithm learns from each view data separately to construct a weak classifier for each view. In the second stage, it combines the classifiers trained in each view to build a strong multi-view model.

We carried out various experiments on twelve multi-view datasets to demonstrate the efficiency of the proposed MVKNN algorithm. The empirical results showed that a significant improvement was achieved by the MVKNN algorithm compared to the KNN algorithm in terms of accuracy. The results also indicated that MVKNN outperformed other well-known machine learning algorithms (SVM, DT, and NB) on average. Furthermore, MVKNN achieved higher accuracy compared to the state-of-the-art multi-view learning methods (MV-LSSVM, MVL-KNN, MVL-LS, and MVDT).

As future work, it should be valuable to expand this study in different ways. First, the proposed algorithm can be extended to handle weighted multi-view learning problems. A core assumption in MVKNN is that all views are basically considered as equally important and given the same weight in majority voting. However, one can expect that the view importance of the learning task can vary significantly. To capture the view importance, the weighted version of the MVKNN method can be implemented. Second, the proposed algorithm can be extended to handle semi-supervised multi-view learning problems. The supervised multi-view classification approaches present limitations when there are few labeled data. Because, in most classification problems, data instances are labeled by a user and this process could be expensive or time-consuming. To overcome this limitation, the MVKNN algorithm can be enhanced by incorporating unlabeled data into the learning process, in addition to the labeled data.

## References

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

[2] Yumurtaci M, Gökmen G, Kocaman Ç, Ergin S, Kilic O. Classification of short-circuit faults in high-voltage energy transmission line using energy of instantaneous active power components-based common vector approach. Turkish Journal of Electrical Engineering & Computer Sciences 2016; 24 (3): 1901-1915. doi:10.3906/elk-1312-131

[3] Madi M, Jarghon F, Fazea Y, Almomani O, Saaidah A. Comparative analysis of classification techniques for network fault management. Turkish Journal of Electrical Engineering & Computer Sciences 2020; 28 (3): 1442-1457. doi:10.3906/elk-1907-84

[4] Köse E, Hocaoglu AK. A new spectral estimation-based feature extraction method for vehicle classification in distributed sensor networks. Turkish Journal of Electrical Engineering & Computer Sciences 2019; 27 (2): 1120-1131. doi:10.3906/elk-1807-49

[5] Boyaci D, Erdoğan M, Yildiz F. Pixel-versus object-based classification of forest and agricultural areas from multiresolution satellite images. Turkish Journal of Electrical Engineering & Computer Sciences 2017; 25 (1): 365-375. doi:10.3906/elk-1504-261

[6] Zhuang F, Karypis G, Ning X, He Q, Shi Z. Multi-view learning via probabilistic latent semantic analysis. Information Sciences 2012; 199: 20-30. doi: 10.1016/j.ins.2012.02.058

[7] Sun S, Liu Y, Mao L. Multi-view learning for visual violence recognition with maximum entropy discrimination and deep features. Information Fusion 2019; 50: 43-53. doi: 10.1016/j.inffus.2018.10.004

[8] Hajmohammadi MS, Ibrahim R, Selamat A. Cross-lingual sentiment classification using multiple source languages in multi-view semi-supervised learning. Engineering Applications of Artificial Intelligence 2014; 36 (1): 195-203. doi: 10.1016/j.engappai.2014.07.020

[9] Wu B, Zhong E, Horner A, Yang Q. Music emotion recognition by multi-label multi-layer multi-instance multi-view learning. In: 22nd ACM International Conference on Multimedia; Orlando, FL, USA; 2014. pp. 117-126.

[10] Sun S. Multi-view Laplacian support vector machines. In: Tang J, King I, Chen L, Wang J (editors). Advanced Data Mining and Applications, Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2011, pp. 209-222.

[11] Houthuys L, Langone R, Suykens JA. Multi-view least squares support vector machines classification. Neurocomputing 2018; 282: 78-88. doi:10.1016/j.neucom.2017.12.029

[12] Liang Z, Zhang G, Li G, Fu W. An algorithm for acupuncture clinical assessment based on multi-view KNN. Journal of Computational Information Systems 2012; 8 (21): 9105-9112.

[13] Minh HQ, Bazzani L, Murino V. A unifying framework in vector-valued reproducing kernel hilbert spaces for manifold regularization and co-regularized multi-view learning. The Journal of Machine Learning Research 2016; 17 (1): 769-840. doi:10.5555/2946645.2946670

[14] Guan X, Liang J, Qian Y, Pang J. A multi-view OVA model based on decision tree for multi-classification tasks. Knowledge-Based Systems 2017; 138: 208-219. doi:10.1016/j.knosys.2017.10.004

[15] Wang H, Nie F, Huang H. Multi-view clustering and feature learning via structured sparsity. In: 30th International Conference on Machine Learning; Atlanta, Georgia, USA; 2013. pp. 352-360.

[16] Zhao J, Xie X, Xu X, Sun S. Multi-view learning overview: recent progress and new challenges. Information Fusion 2017; 38: 43-54. doi: 10.1016/j.inffus.2017.02.007

[17] Sun S. A survey of multi-view machine learning. Neural Computing and Applications 2013; 23 (7-8): 2031-2038. doi: 10.1007/s00521-013-1362-6

[18] Culp M, Michailidis G, Johnson K. On multi-view learning with additive models. The Annals of Applied Statistics 2009; 3 (1): 292-318. doi: 10.1214/08-AOAS202

[19] Xu C, Tao D, Xu C. A survey on multi-view learning. arXiv preprint, arXiv:1304.5634, 2013.

[20] Kumar V, Minz S. Multi-view ensemble learning: an optimal feature set partitioning for high-dimensional data classification. Knowledge and Information Systems 2016; 49 (1): 1-59.

[21] Wang S, Chen Z, Yan Q, Ji K, Peng L et al. Deep and broad URL feature mining for android malware detection. Information Sciences 2020; 513: 600-613. doi: 10.1016/j.ins.2019.11.008

[22] Sun S, Jin F. Robust co-training. International Journal of Pattern Recognition and Artificial Intelligence 2011; 25 (7): 1113-1126. doi: 10.1142/S0218001411008981

[23] Chang X, Yang Y, Wang H. Multi-view construction for clustering based on feature set partitioning. In: IEEE International Joint Conference on Neural Networks (IJCNN); Rio de Janeiro, Brazil; 2018. pp. 1-8.

[24] Yu S, Krishnapuram B, Rosales R, Rao RB. Bayesian co-training. Journal of Machine Learning Research 2011; 12 (79): 2649-2680.

[25] Peng J, Luo P, Guan Z, Fan J. Graph-regularized multi-view semantic subspace learning. International Journal of Machine Learning and Cybernetics 2019; 10 (5): 879-895. doi: 10.1007/s13042-017-0766-5

[26] Zhang H, Gönen M, Yang Z, Oja E. Understanding emotional impact of images using bayesian multiple kernel learning. Neurocomputing 2015; 165: 3-13. doi: 10.1016/j.neucom.2014.10.093

[27] Niu Y, Shang Y, Tian Y. Multi-view SVM classification with feature selection. Procedia Computer Science 2019; 162: 405-412. doi: 10.1016/j.procs.2019.12.004

[28] Xu Z, Sun S. An Algorithm on multi-view adaboost. In: Wong KW, Mendis BSU, Bouzerdoum A (editors). Neural Information Processing Theory and Algorithms, Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, pp. 355-362.

[29] Sun S, Zhang Q. Multiple-view multiple-learner semi-supervised learning. Neural Processing Letters 2011; 34 (3): 229. doi:10.1007/s11063-011-9195-8

[30] Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q et al. Top 10 algorithms in data mining. Knowledge and Information Systems 2008; 14 (1): 1-37. doi: 10.1007/s10115-007-0114-2

[31] Jiang Z, Bian Z, Wang S. Multi-view local linear KNN classification: theoretical and experimental studies on image classification. International Journal of Machine Learning and Cybernetics 2020; 11 (3): 525-543. doi: 10.1007/s13042-019-00992-9

[32] Srivastava SK, Singh SK. Multi-label classification of twitter data using modified ML-KNN. In: Kolhe M, Trivedi M, Tiwari S, Singh V (editors). Advances in Data and Information Sciences. Singapore: Springer, 2019, pp. 31-41.

[33] Xia Y, Peng Y, Zhang X, Bae H. DEMST-KNN: A novel classification framework to solve imbalanced multi-class problem. In: Silhavy R, Senkerik R, Kominkova Oplatkova Z, Prokopova Z, Silhavy P (editors). Artificial Intelligence Trends in Intelligent Systems. Cham, Germany: Springer, 2017, pp.291-301.

[34] Villar P, Montes R, Sánchez A M, Herrera F. Fuzzy-Citation-KNN: a fuzzy nearest neighbor approach for multi-instance classification. In: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE); Vancouver, BC, Canada; 2016. pp. 946-952.

[35] Gupta S, Rana S, Saha B, Phung D, Venkatesh S. A new transfer learning framework with application to model-agnostic multi-task learning. Knowledge and Information Systems 2016; 49 (3): 933-973. doi: 10.1007/s10115-016-0926-z

[36] Peng, P, Xu X, Wang X. Instance-based k-nearest neighbor algorithm for multi-instance multi-label learning. International Journal of Innovative Computing, Information and Control 2014; 10 (5): 1861-1871.

[37] Zhao S, Rui C, Zhang Y. MICkNN: multi-instance covering kNN algorithm. Tsinghua Science and Technology 2013; 18 (4): 360-368. doi: 10.1109/TST.2013.6574674

[38] Liu J, Wang C, Gao J, Han J. Multi-view clustering via joint nonnegative matrix factorization. In: Proceedings of the 2013 SIAM (Society for Industrial and Applied Mathematics) International Conference on Data Mining; Austin, TX, USA; 2013. pp. 252-260. doi: 10.1137/1.9781611972832.28

[39] Ahmad SR, Bakar AA, Yaakub MR, Yusop NMM. Statistical validation of ACO-KNN algorithm for sentiment analysis. Journal of Telecommunication, Electronic and Computer Engineering (JTEC) 2017; 9 (2-11): 165-170.

[40] Kramer O. K-nearest neighbors. In: Dimensionality reduction with unsupervised nearest neighbors. Berlin, Heidelberg, Germany: Springer, 2013, pp. 13-23.

[41] Gunarathna MHJP, Sakai K, Nakandakari T, Momii K, Kumari MKN. Machine learning approaches to develop pedotransfer functions for tropical Sri Lankan soils. Water 2019; 11 (9): 1940. doi: 10.3390/w11091940

[42] Park J, Lee DH. Parallelly running k-nearest neighbor classification over semantically secure encrypted data in outsourced environments. IEEE Access 2020; 8: 64617-64633. doi: 10.1109/ACCESS.2020.2984579

[43] Murphy K, van Ginneken B, Schilham AM, de Hoop BJ, Gietema HA et al. A large-scale evaluation of automatic pulmonary nodule detection in chest CT using local image features and k-nearest-neighbour classification. Medical image analysis 2009; 13 (5): 757-770. doi: 10.1016/j.media.2009.07.001

[44] Lall U, Sharma A. A nearest neighbor bootstrap for resampling hydrologic time series. Water Resources Research 1996; 32 (3): 679–693. doi: 10.1029/95WR02966

[45] Ruan L, Dias MPI, Wong E. Enhancing latency performance through intelligent bandwidth allocation decisions: a survey and comparative study of machine learning techniques. Journal of Optical Communications and Networking 2020; 12 (4): 20-32. doi:10.1364/JOCN.379715

[46] Hossin M, Sulaiman MN. A review on evaluation metrics for data classification evaluations. International Journal of Data Mining & Knowledge Management Process 2015; 5 (2): 1-11. doi: 10.5121/ijdkp.2015.5201

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

[48] Weiss NA, Introductory Statistics. 9th ed. Boston, MA, USA: Pearson Education, 2012.

[49] Yao Z, Song J, Liu Y, Zhang T, Wang J. Research on cross-version software defect prediction based on evolutionary information. IOP Conference Series: Materials Science and Engineering 2019; 563 (5): 1-7. doi: 10.1088/1757-899X/563/5/052092