An Efficient Iterative Approach to Explainable Feature Learning

Dino Vlahek, *Member, IEEE*, and Domen Mongus, *Member, IEEE*

***Abstract*— This article introduces a new iterative approach to explainable feature learning. During each iteration, new features are generated, first by applying arithmetic operations on the input set of features. These are then evaluated in terms of probability distribution agreements between values of samples belonging to different classes. Finally, a graph-based approach for feature selection is proposed, which allows for selecting high-quality and uncorrelated features to be used in feature generation during the next iteration. As shown by the results, the proposed method improved the accuracy of all tested classifiers, where the best accuracies were achieved using random forest. In addition, the method turned out to be insensitive to both of the input parameters, while superior performances in comparison to the state of the art were demonstrated on nine out of 15 test sets and achieving comparable results in the others. Finally, we demonstrate the explainability of the learned feature representation for knowledge discovery.**

***Index Terms*— Data classification, explainable artificial intelligence, feature learning, knowledge discovery.**

I. Introduction

FEATURE learning, or representation learning, describes a set of techniques that allow for defining augmented data representation for improved utilization of classification or regression models [1]. Today, feature learning replaces conventional feature engineering tasks in many applications, ranging from speech recognition [2]–[4] and computer vision [5], [6] to general signal processing [7], [8]. While feature engineering traditionally consists of user administered feature construction, feature evaluation, and feature selection steps [9], [10], feature learning follows these principles in an automated manner. In general, feature learning methods can be divided into unsupervised and supervised approaches [11]. While the former learn from unlabelled data, they rely on data transformations (i.e., feature constructions) and feature selections in order to extract low-dimensional representation that captures the underlying structure of the high-dimensional input data. In contrast, supervised approaches learn features from labeled data. This allows for computing an error term of the resulting model and, thus, performing feature validation prior to their selection [9]. Since we focus here on classification tasks, only the latter are discussed in the continuation.

Supervised feature learning methods can be divided roughly into feature selection, manifold learning, sparse dictionary learning, and deep learning. Feature selection is a process that extracts a subset of relevant features from the original feature space. On the one hand, this contributes to the computational efficiency of the learning process, while removing irrelevant or redundant features may also increase the accuracy of the resulting predictive model [9]. Nonetheless, the efficiencies of such approaches are limited, as they are unable to combine features or introduce new ones [12].

In contrast, manifold learning approaches perform dimensionality reduction by recombination of input features. The rationale behind these methods is that high-dimensional learning samples are often distributed close to some low-dimensional nonlinear manifold [13]. Dimensionality reduction is, thus, achieved by mapping samples onto it [14], [15]. As a consequence of changes in the distances between learning samples achieved in this way, significant distortions may be introduced to the data [16], while resulting classification models are difficult to interpret. In addition, manifold learning approaches can only reduce the dimensionality of the feature space but they cannot extend it. Accordingly, they are unable to increase its informativeness [15], [17].

On the other hand, sparse dictionary learning methods construct a representation of the input data that are based on a linear (or nonlinear) combination of an arbitrary number of basic elements, called atoms [18]. Dictionary learning is, thus, a multiobjective optimization problem, where the sparsity of representation is tried to be maximized in addition to minimization of the representation errors [19]. The classification of unknown samples can, thus, be achieved, based on reconstruction error introduced by class-specific dictionaries [20]. However, with the increasing number of classes, this can become computationally demanding. Alternatively, the classification error can be introduced as an additional optimization criterion during the sparse dictionary learning [19]. Thus, a shared dictionary is obtained, which is tuned for the given classifier. It is obvious that this increases the computational complexity significantly due to

Manuscript received 30 October 2020; revised 9 April 2021; accepted 18 August 2021. Date of publication 3 September 2021; date of current version 3 May 2023. This work was supported in part by the Slovenian Research Agency under Grant L7-2633 and Grant P2-0041, and in part by the projects "Wearable Integrated Smart Brace for Rehabilitation Monitoring and Diagnostic of Disorders in Muscular Functions—WIBRANT" and "Integration of Indoor and Outdoor Navigation—ION," which are cofinanced by the Republic of Slovenia, Ministry of Education, Science and Sport, and the European Union under the European Regional Development Fund (more info: www.eu-skladi.si/?set_language=en). *(Corresponding author: Dino Vlahek.)*

The authors are with the Laboratory for Geospatial Modelling, Multimedia and Artificial Intelligence, Faculty of Electrical Engineering and Computer Science, Institute of Computer Science, University of Maribor, 2000 Maribor, Slovenia (e-mail: dino.vlahek1@um.si).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3107049.

Digital Object Identifier 10.1109/TNNLS.2021.3107049

the nonconvex optimization problem and tuning of many parameters [20], [21]. In addition, while sparse representation allows for interpretation of a resulting model, it is challenging to extract useful knowledge in those cases when the dictionary contains a large number of atoms [21], [22].

Similar drawbacks are also noted when considering deep learning approaches. These are based on various architectures of artificial neural networks (ANNs) with multiple hidden layers of neurons (i.e., nodes) that allow for extracting higher level features progressively from the raw input [23], [24]. Neurons' activation functions can be modeled by both linear and nonlinear functions, thus optimizing feature representation within the decision function. By increasing the number of hidden layers (i.e., the ANN's depth), ANNs can approximate increasingly complex decision functions and achieve high classification (or prediction) accuracies. However, due to the presence of multiple local optima and a large number of so-called hyperparameters (i.e., dimensions) [24], [25], this also increases the complexity of the training procedure. Deep learning approaches are, therefore, difficult to tune, while they are also considered to be black-box function approximators that do not allow for knowledge discovery [1], [26]–[28].

The existing feature learning approaches are, thus, either uninterpretable, or they are limited in their accuracies due to the restricted capacities of feature recombination. Moreover, feature learning approaches that allow for the introduction of new features are computationally complex, as they require nonconvex optimizations with a large number of hyperparameters that need to be tuned. In order to address these challenges, a new method is proposed in this article that allows for the following:

1) achieving improved accuracy in comparison to the current state of the art;
2) computationally efficient learning due to the fast convergence of the model toward optimal representation;
3) efficient tuning due to its insensitivity to only two input parameters;
4) interpretation of the resulting model by feature construction tracking.

The rest of this article is organized as follows. An overview of the related work is given in Section II. Section III provides details of the proposed method. The validation of the proposed method is presented in Section IV, while Section V concludes this article.

II. Related Work

While feature learning can be viewed as an optimization problem where the space of all possible recombinations of features is searched in order to find an optimal representation for classification purposes [1], we follow here the traditional feature construction, evaluation, and selection steps. In the continuation of this section, the related methods are reviewed, accordingly, in addition to approaches that allow for the interpretation of the learned models.

A. Feature Construction

During feature construction, constructive operators are applied to a set of existing features in order to define new ones. In addition to dimensionality reduction approaches (such as the principal components analysis [29], [30] and nonnegative matrix factorization [18], [19], [31], [32]) that construct new features implicitly during feature projection or selection steps, a number of approaches focus on the extraction of higher level features that increase the informativeness of the feature space. ANNs, for example, create such features using hidden neurons within the intermediate layers. Here, input features are recombined by summing up their weighted values (i.e., convolution or dot product), transformed by the activation function, and followed by additional convolutions, such as pooling layers, fully connected layers, and a normalization layer in the case of deep learning. Depending on the structure of the hidden layers, the activation function used, and their tuning (i.e., learning) strategy, this principle applies to a number of ANN architectures, including autoencoders [33], [34], deep feedforward networks [35]–[38], recurrent neural networks [7], [39], and convolutional neural networks [28], [40].

While projection- and ANN-based methods suffer from loss of interpretability, constructing a feature in a traceable manner allows for deriving qualitative understanding between the input variables and the response and, thus, ensures their explainability [41]. Early approaches that satisfied this constrain focused on decision trees for determining the order of combining Boolean features by logical operators [42], [43]. In order to overcome the restriction to deal only with Boolean-valued features, a preprocessing step that transforms real-valued features to Boolean-valued ones, while combining them according to the information gain, was proposed in [44]. Still, the number of constructed features in these methods is usually fixed or predetermined by the user [45].

On the other hand, evolutionary algorithms allow for recombining features automatically with little to no user interaction. For instance, construction of a single high-level feature using arithmetic operations was proposed in [46], where four different metrics were used as fitness functions, including information gain, Gini-index, Chi-square, and their sum, while trigonometric operators based on intraclass variance were considered additionally in [47]. In the following, numerous attempts to construct a multidimensional feature space were proposed [48], [49], and the appropriateness of fitness functions for applying arbitrary classifiers was examined in [50]. Despite numerous studies demonstrating the benefits of an evolutionary algorithm for feature construction [47]–[50], they are obviously computationally inefficient, while they tend to produce only near-optimal results, as they rely on a number of input parameters (e.g., population size and probabilities of mutation and crossover rates) that need to be fine-tuned [51].

B. Feature Evaluation

Following feature construction, feature evaluation provides ranking of features in regard to their capacities to separate samples of different classes. A straightforward approach for feature evaluation relies on assessing the performances of classifiers built on each individual feature [12]. Nonetheless, the choice of a classifier affects the evaluation results highly,

while learning is often time-consuming, especially in those cases of a large number of features [12], [52], [53]. In order to address this issue, the ratios between distances among samples of different classes and those of the same class are used to approximate features' discriminative powers by discriminant analysis-based techniques. Examples of these include the Fisher criterion [54], the maximum margin criterion [55], and the Laplacian score [56]. Likewise, correlation-based approaches use Pearson and Spearman correlation coefficients between feature values and class labels, while techniques based on information gain rank the features according to the ratio between the entropy of class labels and conditional entropy of feature values [57]–[59]. In terms of accuracy, similar results are also obtained by using the computationally more efficient Gini impurity estimation [60]–[62]. While there are many variations of information gain- and Gini-based methods [61], [63]–[66], their major advantage is their ability to acknowledge nonlinear relationships between features and class labels [58], [67]. However, they favor features with a large number of distinct values, which can result in overfitting [57], [60], [62], [68].

C. Feature Selection

Feature selection is the process of selecting a subset of relevant features (i.e., variables or predictors) that maximizes the accuracy of the classification model. Strategies that are traditionally used to achieve this objective include filtering, wrapping, and embedded methods [12]. Filtering approaches are the simplest ones, as they threshold the features according to their evaluation. Although computationally efficient [12], [52], [69], [70], these early approaches tend to select redundant features, as the correlations between them are not considered [12], [71]. One way to overcome this issue is to base the ranking criterion on the entropy-based [65], [66], [72] and discriminant analysis-based metrics [54], [56]. However, inaccurate calculations of these approximations are introduced when the number of training samples is small, while they become computationally costly when using a large number of samples or datasets with a large number of features. More importantly, they only take into consideration pairwise feature dependencies, which results in the selection of suboptimal feature subsets [73].

Wrapper approaches, on the other hand, overcome these drawbacks by selecting such a subset of features that maximizes the performance of the targeted classifier [12]. This is essentially a multiobjective optimization problem of maximizing the classification performance while minimizing the number of selected features. Since multiobjective optimization is well-known, many optimization techniques exist [12], [74], including sequential search [75], [76], nature-inspired algorithms [77]–[79], and binary teaching-learning [80]. While these approaches do not require explicit feature evaluation, they still have the capacity to produce (near) optimal results. However, their performance depends on the targeted classification model. In addition, they are extremely computationally demanding [12], and thus, their usage is significantly limited.

Alternatively, in order to mitigate the time complexity of wrappers, the embedded methods apply feature selection during the training process. Thus, they select relevant features simultaneously while training the classifier using some form of penalization [12]. For example, decision trees achieve feature selection based on the estimation of mutual information [60], while support vector machines (SVMs) use regression analysis (such as least absolute shrinkage and selection operator LASSO with the L1 [81], or ridge regression with the L2 penalty [82]) in order to rank features during their training [83]. This reduces the computational demands of both methods significantly, as it allows for avoiding the repetitive execution of the learning procedure. However, SVMs are not interpretable, while they are also difficult to tune due to their sensitivity to a number of input parameters [12], [84].

D. Interpretability of Machine Learning

The interpretation of the decision-making principles behind the early machine learning approaches is rather elementary. Decision trees, for example, can be analyzed by traversing their nodes straightforwardly and evaluating local and global misclassification [85], while linear models can be interpreted by examining their parameters [86]. Nevertheless, due to their simplicity, these types of models do not represent sufficiently all the knowledge contained within the data [87] and, thus, often achieve low classification accuracies [88]. On the other hand, methods that may be better at extracting the knowledge and, consequently, achieve higher accuracies (like ANNs, random forests (RFs), and SVMs) are often considered to be black-box approaches due to their nonlinearities. Nevertheless, several attempts toward the interpretation of black-box approaches have been made recently [41], [89]–[92]. These are traditionally achieved either by learning interpretable models locally around a given sample [41] or by estimating the importance of each individual feature on the output classification [93], [94]. The latter is based on Shapely values that show how an individual feature contributes to the model's behavior by observing the difference between the probabilities of the expected and the actual classifications when that feature is ignored [93].

Although the described techniques are equally applicable across linear, nonlinear, and deep models, they are sensitive to high correlation among features [94]. Moreover, by observing only the input features, they do not consider the learned ones and are unable to support the interpretation of the dependences between them [92], [94], [95]. While the former has already been examined in [96], addressing the interpretability of codependencies between features turned out to be significantly more challenging. When considering ANNs and CNNs, one possible way to achieve this is by visualization of the decision-making process. This is usually done by generating heatmaps from the input samples, highlighting those that maximize the observed neurons' activations [89], [91], [97]–[100]. As this, in fact, allows for the interpretation of learned features, the visualized heatmaps are often contraintuitive [101], while the approach itself is limited to the CNNs [90]. Recently, an interpretable ANN was proposed, which identifies features

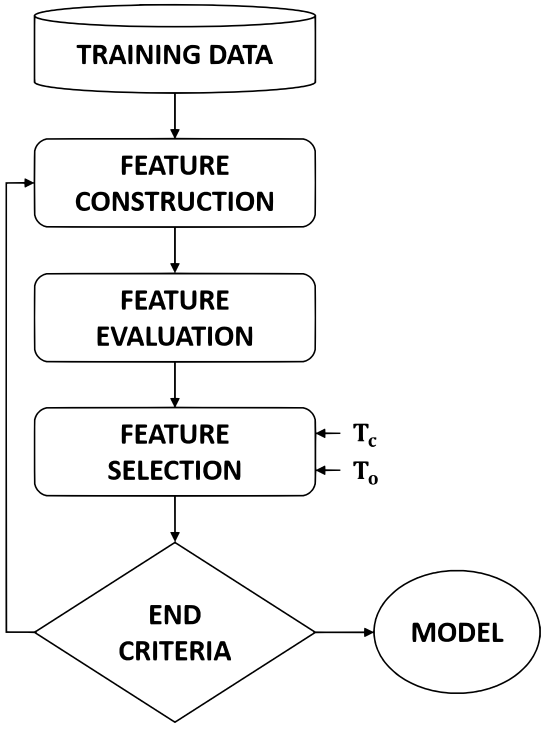

Fig. 1. Flowchart of the proposed method feature learning approach.

with high interclass variances from input data and combines them with those of low intraclass variances in order to improve the understanding of the model's predictions [102]. Although this allows for studying the principles of decision-making behind ANNs and examining the impacts of the input features on the final classification, the learned intermediate features may still be highly abstract and not meaningful for knowledge discovery.

III. Proposed Method

This section introduces a new iterative method for feature learning that allows for exploiting nonlinear codependencies between features in order to improve the classification performance of an arbitrary classifier while also providing a meaningful feature representation for knowledge discovery. As shown in Fig. 1, the proposed method relies on two input parameters, while each iteration consists of the following three steps:

1) feature construction that generates the new feature space $F^M \subseteq \mathbb{R}^M$ from the initial one $F^N \subseteq \mathbb{R}^N$;
2) feature evaluation that assesses the quality of the individual feature;
3) feature selection used to select the high-quality dissimilar features.

In the continuation, each step is discussed in detail.

A. Feature Construction

Within the mathematical framework as set out in this section, discrete functions and sets are denoted with capital letters, while variables and indexes are denoted with lowercase letters. Accordingly, an input feature space at the itth iteration is denoted as F_{it}^N and is defined by a set of features $F_{\text{it}}^N = \langle f_{\text{it}}^n \rangle$. A feature f_{it}^n, referred to by an index $n \in [1, N]$, is given as a mapping function $f_{\text{it}}^n : \mathbb{Z} \rightarrow \mathbb{R}$. On the other hand, an index $i \in [1, I]$ refers to a sample, i.e., a feature vector $\vec{x}_i$ defined as $\vec{x}_i = \langle f_{\text{it}}^n[i] \rangle$. Accordingly, a set of samples X is given by $X = \{\vec{x}_i\}$. A newly generated feature space at the itth iteration is denoted as F_{it}^M, where $m \in [1, M]$ denotes an index of a distinct new feature. Thus, a new feature space $F_{\text{it}}^M = \langle f_{\text{it}}^m \rangle$ is defined as a set of mapping functions $f_{\text{it}}^m : \mathbb{Z} \rightarrow \mathbb{R}$, each given by

$$f_{\text{it}}^m[i] = f_{\text{it}}^n[i] \oplus f_{\text{it}}^{n'}[i] \tag{1}$$

where $\oplus$ is an arbitrary operator applied on feature samples of features n and n', such that $n \neq n'$, respectively. Finally, a concatenated feature space is defined as

$$F_{\text{it}}^{M+N} = F_{\text{it}}^N \cup F_{\text{it}}^M. \tag{2}$$

For the purposes of this study, a set of used operators was defined as $\oplus \in \{+, -, \div, \times\}$. Note that, as division and subtraction are not commutative, they depend on the order of the operands. Thus, the number of newly generated features using summation and multiplication is equal to $N(N-1)$, while subtraction and division generate $2N(N-1)$ new features. The number of newly generated features during each iteration is, thus, defined as

$$M = 3N * (N-1). \tag{3}$$

Accordingly, the number of features in the concatenated feature space F_{it}^{M+N} is equal to $3N^2 - 2N$.

B. Feature Evaluation

The objective of this step is to evaluate each feature according to its suitability for classification [9], [10]. By considering classification as a distance metric learning problem, features with low intraclass [103] and high interclass variances [104] are generally considered to be of high quality [105]. Thus, it can be used instead of assessing feature quality by computationally intensive learning of classifiers. Nevertheless, when estimating intraclass and high interclass variances, samples of different classes may appear within the overlapping clusters, and therefore, straightforward estimation of variances does not ensure sufficient results [106]–[108]. In order to address this issue, we propose a new feature quality measure that assesses the probability distributions agreement between values of samples belonging to different classes.

Probability distributions agreement is a measure of the common area under two probability density curves [109]. Consequently, the low value of probability distributions agreement corresponds to a high separability between classes and improves the tradeoff between interclass and intraclass variances of clusters of learning samples. As shown by Sun and Wang [110], the overlap area also corresponds directly to the expected proportions of classification errors. In a discrete space, the probability distribution agreement can be estimated straightforwardly by the overlap between class histograms, as shown in Fig. 2.

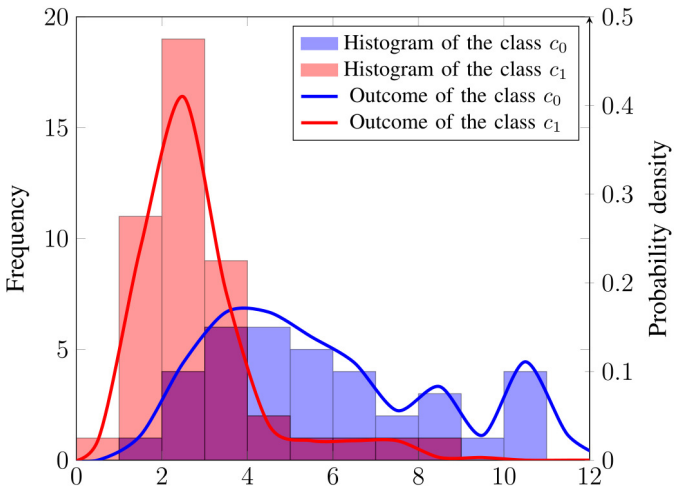

Fig. 2. Probability distributions agreement of classes c_0 and c_1.

Let a classification function $\mathbb{C}$, acting on a set of samples $\vec{x}_i = \langle f_{\text{it}}^n[i] \rangle$, be given by a mapping $\mathbb{C} : X \to [1, C]$. A set of samples $X_c \subseteq X$ of a given class c is then defined by

$$X_c = \{\vec{x}_i \in X; \mathbb{C}(\vec{x}_i) = c\}. \tag{4}$$

In order to estimate the probability distributions, per-class histograms of features are estimated by categorizing (binning) samples $\vec{x}_i \in X_c$ according to their corresponding feature values $f_{\text{it}}^n[i]$. A set of samples $X_c^{n,[a,b)}$, belonging to the category $[a, b)$ according the feature f_{it}^n, is defined as

$$X_c^{n,[a,b)} = \left\{\vec{x}_i \in X_c; a \leqslant f_{\text{it}}^n[i] < b\right\} \tag{5}$$

where $[a, b)$ denotes the binning range. Thus, probability $H_c^{n,[a,b)}$ of a category $[a, b)$ of a class c, estimated based on the histogram of a feature f_{it}^n, is defined as

$$H_c^{n,[a,b)} = \frac{\left|X_c^{n,[a,b)}\right|}{|X^n|}. \tag{6}$$

Such representation of the probability distribution of feature values of a class c allows for a straightforward estimation of the feature quality $\Delta(F^N)$. This is given by the mapping function $\Delta : F_{\text{it}}^N \to [0, 1]$ that estimates an overlap between the probability distributions of samples belonging to different classes. Formally,

$$\Delta\left(F_{\text{it}}^N\right) = \sum_{[a,b)} \left(\sum_c H_c^{n,[a,b)} - \underset{c}{\text{argmax}}\left(H_c^{n,[a,b)}\right) \right). \tag{7}$$

The result of (7) is an estimation of potentially incorrectly classified values of the nth feature. A graphic example of an overlap between two histograms and the corresponding probability functions of classes c_1 and c_2 is shown in Fig. 2. The red graph and the blue graph represent histograms and corresponding probability functions of classes c_0 and c_1, accordingly, while the violet area shows the overlap between them, which corresponds to possible misclassified samples.

C. Feature Selection

As an alternative to the existing feature selection approaches that are either computationally expensive or unable to deal with correlated features (as explained in Section II-C), we proposed in this section a new graph cut-based filtering technique that allows for extracting a subset of high-quality dissimilar features $F_{\text{it}+1}^N \subseteq F_{\text{it}}^{N+M}$ from the concatenated input feature space F_{it}^{N+M}. For this purpose, graph vertices are used to represent features, with associated weights that define their quality (as described in Section II-B), while graph edge weights define similarities between them. Here, an overlap and the correlation threshold, denoted as T_Δ and T_p accordingly, are used for graph definition. The former defines the necessary level of features' quality (i.e., maximal allowed class overlap) to be included into the output feature space, and the latter determines the minimal level of dissimilarity between them. Accordingly, an undirected graph used for feature selection is defined as $G = (F_{\text{it}}^N, E)$, where a set of vertices F^N is defined as $F_{\text{it}}^N = \{f_{\text{it}}^n \in F^{N+M}; \Delta(f_{\text{it}}^n) \leq T_\Delta\}$, while an unordered set of edges $E = \{e_{n,n'}; P(e_{n,n'}) \geq T_p\}$ is given by $e_{n,n'} = (f_{\text{it}}^n, f_{\text{it}}^{n'})$ for all $f_{\text{it}}^n, f_{\text{it}}^{n'} \in F^N$ such that $n \neq n'$. A vertex-weighting function is given by $\Delta(f_{\text{it}}^n)$, as defined in (7), and the edge-weighting function is, in our case, given by the absolute Pearson correlation coefficient $P : E \to [0, 1]$. Formally,

$$P(e_{n,n'}) = \left| \frac{\sum_{i=0}\left(f_{\text{it}}^n[i] - \mu_n\right)\left(f_{\text{it}}^{n'}[i] - \mu_{n'}\right)}{\sigma_n \sigma_{n'}} \right| \tag{8}$$

where μ_n denotes mean, while standard deviation σ_n of feature values is defined as $\sigma_n = ((1/I)\sum_{i=0}(f_{\text{it}}^n[i] - \mu_n))^{1/2}$. According to the theoretical framework set out in [111], we used the following definitions of elementary properties.

1) Vertices $f_{\text{it}}^n \in F_{\text{it}}^N$ and $f_{\text{it}}^{n'} \in F_{\text{it}}^N$ are adjacent in a graph G if there exists an edge $e_{n,n'} \in E$ linking them.
2) A path from f^{n_0} to f^{n_N} is an ordered sequence of adjacent vertices $\prod_{n_0,n_N} = \langle f_{\text{it}}^{n_0}, f_{\text{it}}^{n_1}, \ldots, f_{\text{it}}^{n_N} \rangle$.
3) A graph G is connected if, $\forall\ f_{\text{it}}^n, f_{\text{it}}^{n'} \in F_{\text{it}}^N$, there exists a path $\prod_{n,n'}$ between them.
4) A graph $G' = (F_{\text{it}}^{\prime N}, E')$ is subgraph of G if $F_{\text{it}}^{\prime N} \subseteq F_{\text{it}}^N$ and $E' \subseteq E$,
5) A neighborhood $Z(f_{\text{it}}^n)$ of a vertex f_{it}^n in graph G is the subset of vertices F_{it}^N, defined by all the adjacent vertices of f_{it}^n, namely $Z(f_{\text{it}}^n) = f_{\text{it}}^{n'} \in F_{\text{it}}^N; \exists e_{n,n'}$, where $n \neq n'$.
6) A subgraph $\text{CC}(G)_n$, addressed by a feature f_{it}^n, is a connected component of G if it is connected and it is maximal for this property.

Accordingly, we say that a set of vertices $CUT(F^N) \subseteq F_{\text{it}}^N$ is a vertex-cut if its removal separates graph G into at least two nonempty and pairwise disconnected connected components. It is obvious that $Z(f_{\text{it}}^n)$ is a graph-cut, as it separates a singleton $\{f_{\text{it}}^n\}$ (i.e., an individual vertex) from the rest of the graph, thus creating a subgraph $G' = (F_{\text{it}}^{\prime N}, E')$, whose vertex and edge sets are given formally by

$$\begin{aligned} F_{\text{it}}^{\prime N} &= F_{\text{it}}^N / \left(Z\left(f_{\text{it}}^n\right) \cup \left\{f_{\text{it}}^n\right\}\right) \\ E' &= \left\{e'_{n,n'} \in E;\ f_{\text{it}}^n, f_{\text{it}}^n \in F_{\text{it}}^{\prime N} \text{ and } n \neq n'\right\}. \end{aligned} \tag{9}$$

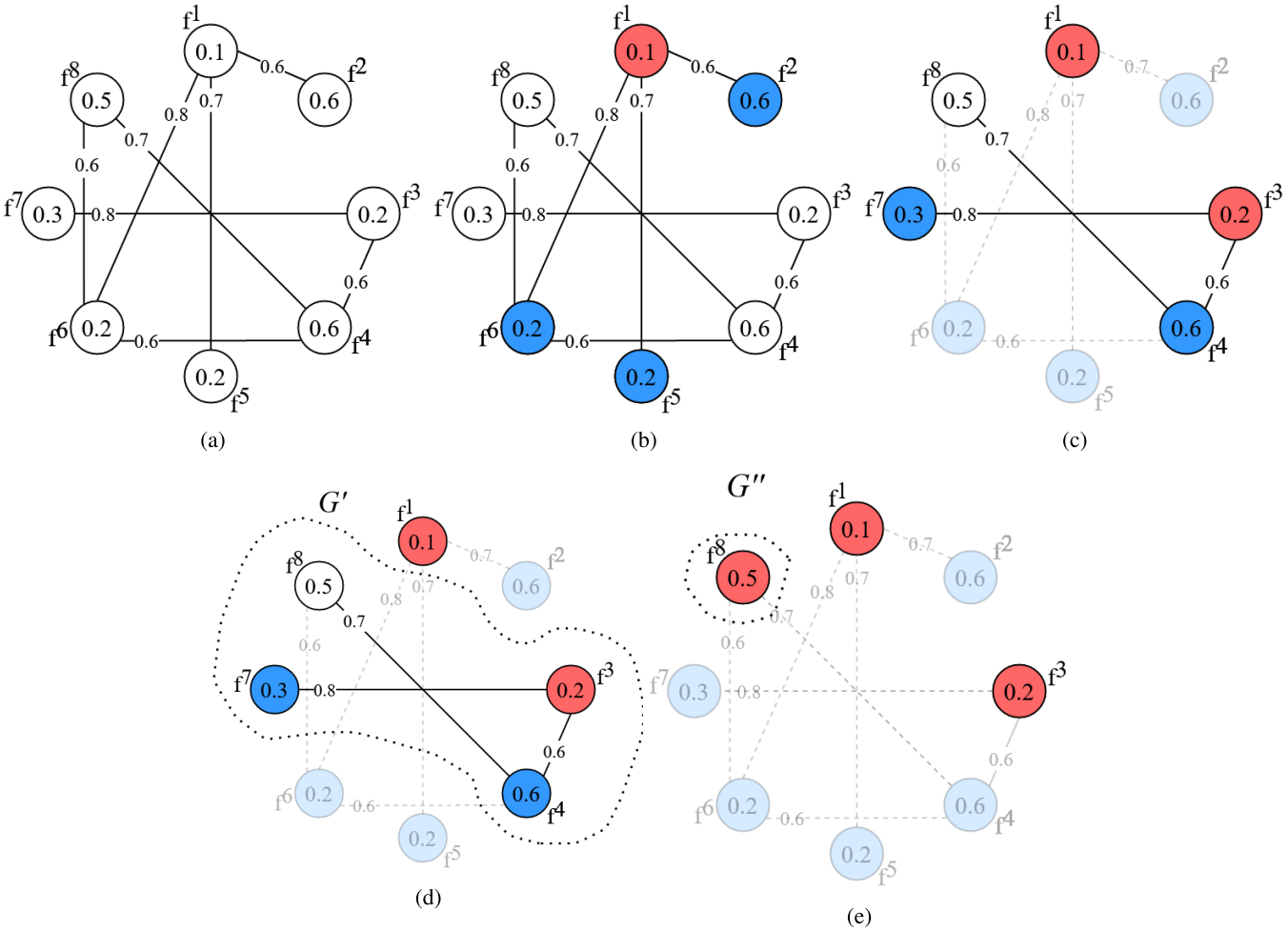

Fig. 3. Vertex cut-based feature selection on (a) Graph G, where (b) feature of the highest quality is selected (red vertex) and (c) its neighborhood (blue vertices) is removed. The same procedure is repeated on the remaining graphs (d) G' and (e) G'' in order obtain the output features (red vertices).

Consider the example in Fig. 3(a) that shows an undirected graph $G = \{F_{\text{it}}^N, E\}$, constructed over a set of features $F_{\text{it}}^{N+M} = \{f_{\text{it}}^1, f_{\text{it}}^2, \ldots, f_{\text{it}}^8\}$, with thresholds $T_\Delta = 0.6$ and $T_p = 0.8$ applied on the associated vertex- and edge-weighting functions Δ and P, accordingly. In order to ensure the preservation of the overall informativeness of selected features, a feature of the highest quality is selected $\hat{f}_{\text{it}}^n = \arg\min_{f_{\text{it}}^m \in G} \Delta(f_{\text{it}}^m)_{n_i}$ first by a vertex-cut of its neighborhood $Z(\hat{f}_{\text{it}}^n)$. Thus, all of its highly correlated adjacent features are removed from G [see Fig. 3(b), where $\hat{f}_{\text{it}}^n = f^1$ is marked by red and $Z(\hat{f}_{\text{it}}^n) = \{f^2, f^5, f^6\}$ is marked by blue vertices]. This results in G', as defined by (9), and a disconnected singleton $\{f^1\}$ [see Fig. 3(c), where G' is marked by white vertices]. As shown in Fig. 3(d), the same process is then repeated on G', separating the feature of the highest quality, namely, f^3, from the remaining graph G'' by removal of $Z(f^3) = \{f^4, f^7\}$. Finally, the output subset of high-quality dissimilar features, namely, $\{f^1, f^3, f^8\}$, is obtained by a vertex-cut of $Z(f^8) \subset G''$, as shown in Fig. 3(e).

As vertex-cut of graph G is a well-known problem, it is not discussed further here. Its efficient implementation is described in [112]–[114].

IV. Results and Discussion

A. Validation Procedure

The proposed method was implemented using C# on the Microsoft Windows 10 operating system, while all conducted experiments were done on a workstation with Intel Core i5 CPU and 16 GB of the main memory. In order to ensure the reproducibility of the experiments, the proposed method relied on Open CV version 2.4.13 implementation of classifiers with the following settings.

1) The K-nearest neighbor (KNN) classifier was assessed using default settings, where $K \in \{2, 3, \ldots, 8\}$ were tested.
2) The naive Bayes classifier (NBC) was used with the default settings.
3) ANN was of the back propagation type with symmetrical sigmoid activation function,
4) RF was of maximal depth from the range $\{2, 4, 8, 16, 20\}$, while the maximal number of iterations was from $\{5, 10, 15, 20, 25, 30\}$.
5) SVM used a radial basis function as the kernel function, with cost parameter $C \in \{2^{-4}, 2^{-3}, \ldots, 2^{11}\}$ and kernel parameter $\gamma \in \{2^{-11}, \ldots, 2^3, 2^4\}$.

TABLE I
DESCRIPTION OF TEST DATASETS

Dataset ID	Dataset name	#Features	#Classes	#Samples
Ds1	Fertility	9	2	100
Ds2	Haberman's Survival	3	2	306
Ds3	Statlog (Heart)	13	2	270
Ds4	Ionosphere	34	2	351
Ds5	Iris	4	3	150
Ds6	Liver Disorders	7	2	345
Ds7	Lung Cancer	56	3	32
Ds8	Parkinsons	22	2	195
Ds9	Seed	7	3	210
Ds10	Sonar	60	2	208
Ds11	User Knowledge Modeling	5	4	403
Ds12	Vertebral Column	6	2	130
Ds13	Wisconsin Breast Cancer Diagnostic	30	2	569
Ds14	Wine	13	3	178
Ds15	Zoo	16	7	101

In all cases, all combinations of parameter values were tested, and the highest measured results are reported in the continuation. All tests were conducted by tenfold cross validation [115], using average accuracy acc as the indicator of the method's efficiency. Based on the confusion matrix [59], accuracy was defined as

$$\text{acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}. \tag{10}$$

Accuracy assessment was conducted on 15 well-known benchmark datasets, available at the UCI machine learning [116] and KEEL dataset [117] repositories. Table I summarizes the characteristics of individual datasets, including their name, the number of features, the number of classes, and the number of contained samples. On this basis, the validation protocol consisted of the following steps.

1) Sensitivity analysis of the method, where the overall optimal parameters were identified and the best performing classifier was selected.
2) The comparison with the state of the art was performed next, in order to validate the efficiency of the proposed method.
3) The interpretation of the resulting classification model for knowledge discovery was assessed finally.

B. Sensitivity Analysis

In addition to the termination criteria (in our case defined by the convergence of features, achieved when no new features are introduced during the last iteration), the proposed method relies on two user-defined parameters that affect the learned feature representation, namely, the correlation T_p and the overlap T_Δ thresholds. In order to assess the method's sensitivity and estimate their optimal values, feature learning was achieved using all possible combinations of values from the sets $T_p \in \{0.0, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8\}$ and $T_\Delta \in \{0.0, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6\}$. An overview of the accuracies achieved by five classifiers (described in Section IV-A) on the obtained feature representations is shown in Fig. 4.

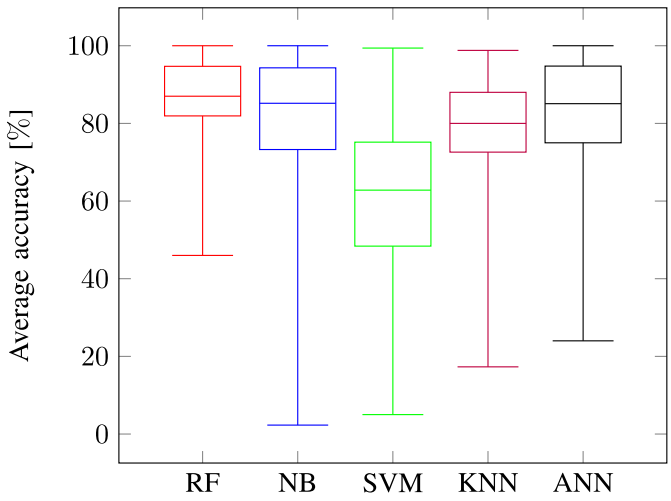

Fig. 4. Measured accuracies of tested classifier when applied on the learned feature representation using $T_p \in \{0.0, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8\}$ and $T_\Delta \in \{0.0, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6\}$, where each box plot displays the minimum and the maximum, the first and third quartiles, and the median accuracy.

Notably, RF achieved the best results, with the highest median accuracy of 81.23%. In addition, it demonstrated the highest robustness, as it achieved the lowest interquartile and the smallest overall range of the measured accuracies among all tested classifiers. Accordingly, the method's sensitivity was analyzed by closer inspection of its results. For this purpose, a so-called "one-factor-at-a-time" assessment [118] was conducted by averaging the measured accuracies when changing the values of one threshold while keeping the other one fixed. The obtained average accuracies and the associated standard deviations are reported in Table II.

As follows from the results, the averaged standard deviation of the classification accuracy in regard to T_p and T_Δ was lower than 2%, indicating the method's insensitivity to both input parameter values. Still, best performances were observed in the cases of $T_p = 0.7$ and $T_\Delta = 0.5$, as lower T_p and T_Δ values restricted the informativeness of the feature space by selecting only a small number of highly uncorrelated and high-quality features, respectively. On the other hand, high T_p values allowed for the inclusion of redundant features, while features with Δ above 0.5 decreased the accuracy by increasing the overall intraclass variance. Note that the original feature representation was obtained when choosing $T_p = 0.0$ or $T_\Delta = 0.0$. Thus, the proposed method allowed for improving overall average classification accuracy from 82.86 achieved on the original data representation to 87.66 achieved on the learned representation.

In addition, the values T_p and T_Δ affect the method's execution time directly, as it depends on the number of introduced features during each interaction. Namely, as the number of newly generated features is square-dependent on the number of input features [see (3)], execution times decrease significantly using strict thresholds, while loose thresholds may turn out to be computationally demanding. Nevertheless,

TABLE II

CLASSIFICATION ACCURACIES ACHIEVED BY RF ON THE LEARNED FEATURE REPRESENTATION WITH THE RESPECT TO THE INPUT VALUES (a) T_p AND (b) T_Δ

	Average classification accuracy [%]								
$T_p =$	0.0	0.5	0.55	0.6	0.65	0.7	0.75	0.8	σ_{T_p}
Ds1	87.63	89.49	90.29	91.16	88.55	91.03	90	90.74	1.17
Ds2	72.1	72.2	72.68	72.74	72.65	73.3	73.16	72.71	0.38
Ds3	91.9	94.65	94.95	95.45	94.54	93.97	94.23	94.57	0.99
Ds4	94.93	94.44	94.79	94.39	95.35	95.54	95.63	95.89	0.53
Ds5	68.29	71.14	71.37	71.63	71.47	71.18	71.35	71.8	1.06
Ds6	44.24	81.39	85.19	84.83	82.27	84.99	85.54	85.28	13.3
Ds7	85.62	83.88	82.85	84.63	84.57	84.71	84.79	84.6	0.75
Ds8	91.66	92.92	93.34	94.11	94.13	95	95.06	93.23	1.06
Ds9	77.39	83.96	82.45	85.38	84.66	83.59	83.37	83.48	2.29
Ds10	80.92	84.65	84.78	83.51	83.08	83.33	83.43	84.17	1.13
Ds11	86.94	85.76	85.93	85.92	91.63	92.41	93.62	92.07	3.21
Ds12	82.3	82.51	82.32	81.51	81.31	81.64	81.01	82.58	0.56
Ds13	95.05	97.13	97.28	97.01	95.91	97.57	96.2	97.2	0.81
Ds14	96.36	94.98	96.4	96.82	97.47	97.22	97.04	96.91	0.72
Ds15	89.07	85.08	87.55	87.29	88.57	89.44	88.11	88.9	1.3
Avg(acc)	82.96	86.27	86.81	87.09	87.07	**87.66**	87.50	87.60	1.45

(a)

	Average classification accuracy [%]							
$T_\Delta =$	0.0	0.35	0.4	0.45	0.5	0.55	0.6	σ_{T_Δ}
Ds1	87.63	89.92	90.28	90.3	90.24	90.24	90.09	0.9
Ds2	72.1	72.39	72.51	72.76	72.96	72.96	73.07	0.33
Ds3	91.9	94.54	94.73	94.69	94.64	94.76	94.66	0.97
Ds4	94.93	95.85	95.9	94.2	94.72	94.8	95.41	0.58
Ds5	68.29	68.29	71.65	71.13	71.6	71.38	71.35	1.42
Ds6	44.24	44.24	44.24	79.67	88.17	86.11	84.62	20.13
Ds7	85.62	84.18	84.48	84.26	84.16	84.18	84.47	0.48
Ds8	91.66	91.79	93.75	94.47	94.37	94.65	94.6	1.23
Ds9	77.39	82.9	83.98	84.49	84.31	84.31	84.08	2.37
Ds10	80.92	83.99	84.05	83.65	83.82	83.88	83.71	1.03
Ds11	86.94	86.94	86.94	88.79	88.98	89.15	90.74	1.36
Ds12	82.3	82.18	82.08	81.85	81.77	82.25	81.76	0.21
Ds13	95.05	96.9	97.02	96.83	96.79	97.08	96.67	0.65
Ds14	96.36	96.46	96.2	96.73	97.64	97.45	97.38	0.55
Ds15	87.07	87.07	87.23	88.58	88.18	87.33	88.23	0.59
Avg(acc)	82.96	83.84	84.33	86.82	**87.49**	87.36	87.37	1.84

(b)

TABLE III

AVERAGE EXECUTION TIMES OF THE PROPOSED METHOD WITH RESPECT TO THE INPUT VALUES (a) T_p AND (b) T_Δ

	Average execution times [s]						
$T_p =$	0.5	0.55	0.6	0.65	0.7	0.75	0.8
Ds1	7.93	7.98	7.92	7.8	7.74	7.76	7.75
Ds2	0.09	0.27	0.9	5.34	5.21	5.3	5.27
Ds3	40.7	40.2	40.5	40.4	40.5	40.2	40.1
Ds4	836.35	800.67	788.11	817.21	703.05	695.7	697.76
Ds4	0.04	0.01	0.01	0.12	0.36	1.53	1.93
Ds6	8.42	8.39	8.7	12.15	10.25	9.25	9.79
Ds7	784.31	681.97	634.96	669.64	689.75	664.14	706.3
Ds8	159.21	161.09	156.81	156.75	156.42	155.7	155.7
Ds9	0.21	1.06	0.08	0.44	0.62	4.16	4.95
Ds10	2828.88	2767.59	2814.23	2798.33	2799.13	2784.47	2727.37
Ds11	0.75	1.08	1.05	2.74	3.63	3.86	4.15
Ds12	9.97	10.03	9.89	9.89	9.8	9.84	9.91
Ds13	674.74	780.87	790.83	779.73	810.26	810.26	815.36
Ds14	18.45	20.66	24.14	25.53	24.89	27.03	27.35
Ds15	10.74	13.98	15.76	16.82	18.9	21.47	20.21

(a)

	Average execution times [s]					
$T_\Delta =$	0.35	0.4	0.45	0.5	0.55	0.6
Ds1	7.84	7.86	7.84	7.83	7.84	7.84
Ds2	3.25	3.17	3.12	3.23	3.17	3.24
Ds3	31.8	37.7	46.1	46.7	49.3	51.2
Ds4	562.8	798.62	795.46	807.53	802.28	809.47
Ds4	0.04	0.23	0.43	0.75	0.8	1.18
Ds6	0.82	7.32	12.44	12.86	11.88	12.05
Ds7	37.93	227.74	674.88	829.88	1066.75	1303.73
Ds8	156.26	158.12	157.03	156.75	158.38	157.77
Ds9	0.07	0.11	1.52	1.96	3.13	3.07
Ds10	1674.59	2253.21	3204.57	3183.69	3185.48	3199.96
Ds11	0.02	0.02	0.02	2.77	5.11	6.86
Ds12	9.85	9.89	9.94	9.92	9.92	9.9
Ds13	550.63	825.86	826.87	824.44	832.7	821.25
Ds14	9.96	17.7	25.03	28.43	30.52	32.4
Ds15	3.42	9.05	11.95	19.32	25.23	32.06

(b)

as shown in Table III, the average feature learning time was less than an hour in all the test cases. Furthermore, the execution times were more affected by threshold values when considering those datasets with a lower number of features and low computational demands (i.e., Ds1, Ds4, Ds9, Ds12, and Ds15), while execution times were doubled at most in those cases of computationally demanding datasets (e.g., Ds10). In general, however, T_Δ showed a stronger impact on the computational times than T_p. Finally, as the learned feature space was composed of relatively straightforwardly structured features (see examples in Fig. 5), the execution time of sample classification remained relatively unaffected.

C. Comparison With the State of the Art

In order to prove the proposed method's efficiency, RF was applied on the learned feature representation under the input thresholds $T_p = 0.7$ and $T_\Delta = 0.5$, while the measured accuracies were compared to six different state-of-the-art approaches, including the following:

1) floating-centroid method (FCM) [36];
2) nearest-neighbor partitioning (NNP) [37];
3) versatile elliptic basis function neural network (NNP) [35];
4) oblique elliptical basis function network (OEBFN) [38];
5) data augmentation (DA) [119];

TABLE IV

COMPARISON OF THE ACCURACIES ACHIEVED BY THE PROPOSED METHOD USING RF WITH THE CURRENT STATE OF THE ART (BEST RESULTS PER DATASET ARE HIGHLIGHTED)

	Classification accuracy [%]						
	RF	FCM [36]	NNP [37]	VEBF [35]	OEBFN [38]	DA [119]	DWS-MK
Ds1	**96.00**	83.64	79.09	N/A	N/A	N/A	88.00
Ds2	**76.77**	73.75	71.56	N/A	N/A	N/A	74.03
Ds3	86.29	N/A	N/A	85.19	**90.74**	N/A	N/A
Ds4	**95.15**	94.17	93.89	81.2	92.31	91.18	91.75
Ds5	97.78	97.33	**99.33**	96.67	97.78	94.27	97.33
Ds6	75.10	N/A	N/A	68.12	**76.81**	N/A	74.85
Ds7	**100.00**	N/A	N/A	76.67	66.67	N/A	N/A
Ds8	**90.02**	87	90.00	N/A	N/A	N/A	N/A
Ds9	**98.10**	95.24	96.19	N/A	N/A	89.38	93.33
Ds10	**87.36**	N/A	N/A	82.23	86.56	77.81	84.76
Ds11	95.28	**95.95**	95.24	N/A	N/A	N/A	N/A
Ds12	84.84	85.16	**85.48**	N/A	N/A	N/A	N/A
Ds13	**98.42**	94.9	95.6	N/A	N/A	97.89	96.68
Ds14	99.41	98.89	98.89	N/A	N/A	**100**	98.87
Ds15	**97.00**	93.33	94.67	N/A	N/A	N/A	N/A

6) depth-width-scaling multiple kernel learning (DWS-MKL) [120].

While FCM, NNP, NNP, and OEBFN are all fit for neural networks, DA and DWS-MKL learn features independently, and thus, different results may be obtained when applying different classifiers. Note, however, that, in the continuation, only the highest achieved accuracies are reported, regardless of the actual classifier being used.

As follows from the results shown in Table IV, the proposed method outperformed the compared methods in nine out of 15 test cases while achieving comparable results on the others. The only exception was Ds3, where OEBFN achieved over 4% improvements over the proposed method. As OEBFN is based on a neural network that was tuned individually for each of the given test datasets, one possible reason for this lies in the definition of the used classifier. Note that, in contrast, the proposed method was tested using only the default parameters of the applied classifiers (as reported in Section IV-A) in order to allow for straightforward repeatability of the reported results. However, by tuning the parameters of RF to Ds3, the classification accuracy improved from 86.29% to 88.89%, supporting this point. In addition, by tuning the parameters of the proposed method itself, improved feature representation was also achieved in test datasets Ds5, Ds6, Ds11, Ds12, and Ds14. As shown in Table V, the proposed method, thus, outperformed all the compared methods in these cases.

D. Interpretation of Learned Feature Representation

Finally, we discuss the interpretation of the resulting feature representation. Due to the extensive background research that allowed us to examine the correctness of the learned feature representation, the Wine Quality Dataset Red (WQDSR) was examined for this purpose [116]. It consisted of 1599 samples of red wines, classified into low- (c_0), medium- (c_1), and high-quality (c_2) classes in accordance with the measured physiochemical characteristics, as given in Table VI. While

TABLE V

IMPROVED FEATURE REPRESENTATION ON TEST SETS Ds5, Ds6, Ds11, Ds12, AND Ds14, ACHIEVED BY TUNING THE INPUT PARAMETERS T_p AND T_Δ

	T_p	T_Δ	Basic classifier	Accuracy [%]
Ds5	0.75	0.6	ANN	99.33
Ds6	0.8	0.45	RF	77.18
Ds11	0.75	0.4	RF	96.75
Ds12	0.7	0.4	RF	85.80
Ds14	0.75	0.5	RF	100

the classification accuracy on the original data representation was equal to 80.5%, in this case, 86.3% accuracy was measured under the same settings when using the learned feature representation. Accordingly, knowledge discovery was carried out by examining the achieved information enrichment. For this purpose, we examined the contributions of the learned features on the output classification using the so-called Shapley Additive Explanations (SHAP) framework [94]. Note that the proposed feature learning removed highly correlated features by itself, while the construction of the features was traceable; thus, such interpretation was not affected by any of its drawbacks, as reported in Section II.

Fig. 5 shows SHAP summary plots of learned features' contributions to low-, medium-, and high-quality classes of wines, which are ordered by their estimated qualities (as described in Section III-B). The X-axis of each plot shows the estimated Shapley values, defined by the difference between the expected probabilities of samples belonging to a specific class and the estimated probabilities of belonging to the same class when a given feature was ignored. The Y-axis gives a histogram of samples associated with the estimated Shapley values, while colors encode the samples' learned features' values, with a gradient ranging from blue (lowest feature value) to red (highest feature value).

When considering red wines, their density f_0^8 is, in general, considered as an indicator of quality, as dense wines contain more fixed compounds of salts, mineral compounds, organic acids (malic, tartaric, and lactic), phenolic substances (anthocyanin, flavonols, tannins, and hydroxybenzoic and hydroxycinnamic acids), and reducing sugars (arabinose, rhamnose, and xylose), which contributes to a better overall mouthfeel experience [121]. In our case, however, the measured levels of salt f_0^5, subtracted from the measured wine's density f_0^8, proved to be the most informative feature, with the quality assessment equal to $\Delta(f_0^8 - f_0^5) = 0.1645$. According to the related SHAP summary plot, low levels of density in combination with high saltiness (i.e., low $f_0^8 - f_0^5$ values) were characteristic for low-quality wines, while medium- and high-quality wines were recognized by higher densities and lower levels of salt (i.e., high $f_0^8 - f_0^5$ values). As salinity is generally understood as an unwanted characteristic of red wine [122], [123], while it still contributes to its density, the learned difference between both is arguably a better indicator of the wine's actual quality. Still, as indicated by the relatively low dispersion of SHAP values, the learned

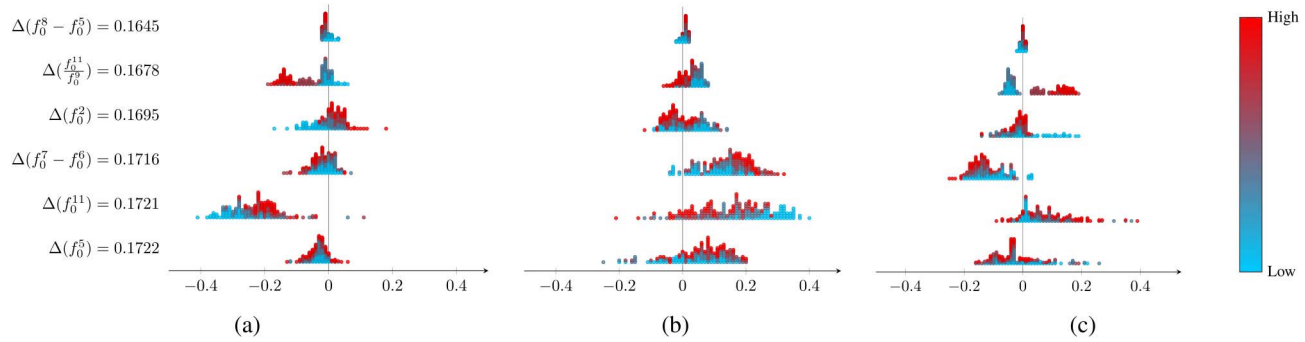

Fig. 5. SHAP summary plots for (a) low-quality wines c_0, (b) medium-quality wines c_1, and (c) high-quality wines c_2.

TABLE VI
INPUT FEATURES OF WQDSR

ID	Range	Description
f_0^1	$f_0^1 \in [0.12, 1.58]$	Concentrations of non-volatile acids
f_0^2	$f_0^2 \in [4.6, 15.9]$	Concentrations of acetic acid
f_0^3	$f_0^3 \in [0.0, 1.0]$	Concentrations of citric acid
f_0^4	$f_0^4 \in [0.9, 15.5]$	Concentrations of sugar remaining after fermentation stops
f_0^5	$f_0^5 \in [0.012, 0.611]$	Concentrations of salt in the wine
f_0^6	$f_0^6 \in [1.0, 72.0]$	Concentrations of free Sulphur dioxide SO_2 molecules
f_0^7	$f_0^7 \in [6.0, 289.0]$	Concentrations of total SO_2 molecules as free ones + ones in bound form
f_0^8	$f_0^8 \in [0.99007, 1.0037]$	The density of a wine in relation to the water
f_0^9	$f_0^9 \in [2.74, 4.01]$	The level of acidity on a pH scale
f_0^{10}	$f_0^{10} \in [0.33, 2]$	Concentration of sulphates in wine
f_0^{11}	$f_0^{11} \in [8.3, 15]$	Concentrations of alcohol ethanol

feature did not show a significant impact on the classification due to the fact that f_0^5 was itself included in the learned representation.

In contrast, concentration of alcohol f_0^{11} had a notably larger impact on the classification, with estimated $\Delta(f_0^{11}) = 0.1721$ as an individual feature, and $\Delta((f_0^{11}/f_0^9)) = 0.1678$ in its relation to the level of acidity f_0^9. On the one hand, f_0^{11} decreased the likelihood of low-quality classes while increasing the likelihood of medium- and high-quality classes. Numerous studies support this observation, as wines with higher alcohol content are associated with more complex flavor and aroma, as well as an increased sense of bitterness, sweetness, and hotness [124], [125]. On the other hand, wines with higher pH levels can be less stable, as they can turn red wines brown, increase their oxidative potential, affect their ability to age well, or cause premature aging [126]. It is, therefore, obvious that red wines with lower pH levels and higher alcohol concentrations are traditional of higher quality. While extensive studies of red wines have already confirmed the importance of this relation [125], [127], [128], the proposed method also managed to extract this knowledge from the learning data. As shown in Fig. 5(a)–(c), high values of (f_0^{11}/f_0^9) showed a highly positive impact on c_2 while decreasing the likelihood of classes c_1 and, even strongly, c_0. On the contrary to the pH levels, acetic acid contributes to the smell and taste of vinegar, while its high concentration typically indicates bacterial spoilage [129], [130]. Aligned with these studies, a large positive impact on the classification of low-quality wines was observed in our study at high levels of acetic acid f_0^2, while low values of f_0^2 showed a significantly positive impact on medium- and high-quality wines.

Finally, SO_2 protects wine from oxidation, as well as microbial contamination during aging, and thus, it is the most commonly used preservative additive in wine production [131]. In the learned representation, the number of bounded SO_2 estimated by $f_0^7 - f_0^6$ showed to be more informative than the input level of the free (f_0^6) and the total number of SO_2 molecules (f_0^7). With a measured overlap of $\Delta(f_0^7 - f_0^6) = 0.1716$, a considerable impact on the differentiation between medium- and high-quality wines was noted, where a higher concentration of bound SO_2 molecules had a substantial negative impact on the latter and positive on the former. This is also consistent with numerous studies in the field of Oenology [131], [132], where it has been shown that a high concentration of bound SO_2 indicates oxidation or microbial spoilage as a consequence of an improper aging of wine, decreasing wine quality and imparting unpleasant flavors and aromas.

V. CONCLUSION

A new iterative approach to explainable feature learning is introduced in this article. During each iteration, new features are generated by arithmetic combinations of the input ones, while their quality is assessed in terms of probability distribution agreements between values of samples belonging to

different classes. Finally, dissimilar features of high quality are selected using vertex-cuts on a graph with edge weights defined by the correlation between them. Due to the fast convergence of the model toward the local optimum, computationally efficient learning was achieved in this way. By being insensitive to both of the input parameters, namely, feature quality and correlation thresholds, the proposed method also provided significantly better performance than the compared methods in ten of 15 commonly used test sets while achieving comparable results in the others. Finally, we demonstrated the correctness of the learned data representation, thus proving the method's potentials for knowledge discovery. Nevertheless, it should be noted that only one dataset was examined as a proof of concept, while the method's meaningfulness on different datasets should be examined further.

In addition to the systematic analysis of its correctness for knowledge discovery, the method's potentials for image processing will be considered in our future work. While the latter requires the application of image processing operators, such as morphological and convolution filters, instead of arithmetic ones, different feature selection strategies will also be considered for this purpose.

REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[2] P. Sharma, V. Abrol, and A. K. Sao, "Deep-sparse-representation-based features for speech recognition," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 25, no. 11, pp. 2162–2175, Nov. 2017.

[3] H. Yu, Z.-H. Tan, Z. Ma, R. Martin, and J. Guo, "Spoofing detection in automatic speaker verification systems using DNN classifiers and dynamic acoustic features," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4633–4644, Oct. 2018.

[4] W. Pei, H. Dibeklioglu, D. M. J. Tax, and L. van der Maaten, "Multivariate time-series classification using the hidden-unit logistic model," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 4, pp. 920–931, Apr. 2018.

[5] Z. Li, Z. Lai, Y. Xu, J. Yang, and D. Zhang, "A locality-constrained and label embedding dictionary learning algorithm for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 278–293, Feb. 2017.

[6] W. Luo, J. Li, J. Yang, W. Xu, and J. Zhang, "Convolutional sparse autoencoders for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3289–3294, Jul. 2018.

[7] X. Li and X. Wu, "Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 4520–4524.

[8] S. Sakhavi, C. Guan, and S. Yan, "Learning temporal information for brain-computer interface using convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5619–5629, Nov. 2018.

[9] H. Liu and H. Motoda, *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Norwell, MA, USA: Kluwer, 1998.

[10] G. Dong and H. Liu, *Feature Engineering for Machine Learning and Data Analytics*. Boca Raton, FL, USA: CRC Press, 2018.

[11] G. Zhong, L.-N. Wang, X. Ling, and J. Dong, "An overview on data representation learning: From traditional feature learning to recent deep learning," *J. Finance Data Sci.*, vol. 2, no. 4, pp. 265–278, Dec. 2016.

[12] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.

[13] Y. Ma and Y. Fu, *Manifold Learning Theory and Applications*, 1st ed. Boca Raton, FL, USA: CRC Press, 2011.

[14] D. Lunga, S. Prasad, M. M. Crawford, and O. Ersoy, "Manifold-learning-based feature extraction for classification of hyperspectral data: A review of advances in manifold learning," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 55–66, Jan. 2014.

[15] H. Qiao, P. Zhang, D. Wang, and B. Zhang, "An explicit nonlinear mapping for manifold learning," *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 51–63, Feb. 2013.

[16] M. Aupetit, "Visualizing distortions and recovering topology in continuous projection techniques," *Neurocomputing*, vol. 70, nos. 7–9, pp. 1304–1330, Mar. 2007.

[17] S.-Q. Zhang, "Enhanced supervised locally linear embedding," *Pattern Recognit. Lett.*, vol. 30, no. 13, pp. 1208–1218, 2009.

[18] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proc. IEEE*, vol. 98, no. 6, pp. 1045–1057, Jun. 2010.

[19] Y. Suo, M. Dao, U. Srinivas, V. Monga, and T. D. Tran, "Structured dictionary learning for classification," 2014, *arXiv:1406.1943*. [Online]. Available: http://arxiv.org/abs/1406.1943

[20] W. Tang, A. Panahi, H. Krim, and L. Dai, "Analysis dictionary learning based classification: Structure for robustness," *IEEE Trans. Image Process.*, vol. 28, no. 12, pp. 6035–6046, Dec. 2019.

[21] M. J. Gangeh, A. K. Farahat, A. Ghodsi, and M. S. Kamel, "Supervised dictionary learning and sparse representation—A review," *CoRR*, vol. abs/1502.05928, pp. 1–60, Feb. 2015.

[22] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. R. Bach, "Supervised dictionary learning," in *Proc. 21st Int. Conf. Neural Inf. Process. Syst.*, 2009, pp. 1033–1040.

[23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[24] M. A. Nielsen, *Neural Networks and Deep Learning*. San Francisco, CA, USA: Determination Press, 2018.

[25] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 8, pp. 1553–1565, Aug. 2014.

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[27] L. Miralles-Pechuán, D. Rosso, F. Jiménez, and J. M. García, "A methodology based on deep learning for advert value calculation in CPM, CPC and CPA networks," *Soft Comput.*, vol. 21, no. 3, pp. 651–665, Feb. 2017.

[28] J. Gu *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, May 2018.

[29] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *J. Educ. Psychol.*, vol. 24, no. 6, pp. 417–441, 1933.

[30] H. Yu and J. Yang, "A direct LDA algorithm for high-dimensional data—With application to face recognition," *Pattern Recognit.*, vol. 34, no. 10, pp. 2067–2070, Oct. 2001.

[31] S. Sra and I. S. Dhillon, "Generalized nonnegative matrix approximations with Bregman divergences," in *Advances in Neural Information Processing Systems*, vol. 18, Y. Weiss, B. Schölkopf, and J. C. Platt, Eds. Cambridge, MA, USA: MIT Press, 2006, pp. 283–290.

[32] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[33] L. Le, A. Patterson, and M. White, "Supervised autoencoders: Improving generalization performance with unsupervised regularizers," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst. (NIPS)*. Red Hook, NY, USA: Curran Associates, 2018, pp. 107–117.

[34] Y. Sun, B. Xue, G. G. Yen, and M. Zhang, "A particle swarm optimization-based flexible convolutional autoencoder for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 8, pp. 2295–2309, Aug. 2019.

[35] S. Jaiyen, C. Lursinsap, and S. Phimoltares, "A very fast neural learning for classification using only new incoming datum," *IEEE Trans. Neural Netw.*, vol. 21, no. 3, pp. 381–392, Mar. 2010.

[36] L. Wang *et al.*, "Improvement of neural network classifier using floating centroids," *Knowl. Inf. Syst.*, vol. 31, no. 3, pp. 433–454, 2012.

[37] L. Wang, B. Yang, Y. Chen, X. Zhang, and J. Orchard, "Improving neural-network classifiers using nearest neighbor partitioning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2255–2267, Oct. 2017.

[38] H.-W. Peng, S.-J. Lee, and C.-H. Lee, "An oblique elliptical basis function network approach for supervised learning applications," *Appl. Soft Comput.*, vol. 60, pp. 552–563, Nov. 2017.

[39] A. Dong, Z. Du, and Z. Yan, "Round trip time prediction using recurrent neural networks with minimal gated unit," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 584–587, Apr. 2019.

[40] H. Han, Y. Li, and X. Zhu, "Convolutional neural network learning for generic data classification," *Inf. Sci.*, vol. 477, pp. 448–465, Mar. 2019.

[41] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1135–1144.

[42] G. Pagallo, "Learning DNF by decision trees," in *Proc. 11th Int. Joint Conf. Artif. Intell.*, vol. 1, 1989, pp. 639–644.

[43] C. J. Matheus and L. A. Rendell, "Constructive induction on decision trees," in *Proc. 11th Int. Joint Conf. Artif. Intell.*, vol. 1, 1989, pp. 645–650.

[44] Y.-J. Hu and D. Kibler, "Generation of attributes for learning algorithms," in *Proc. AAAI/IAAI*, vol. 1, Jan. 1996, pp. 806–811.

[45] S. Markovitch and D. Rosenstein, "Feature generation using general constructor functions," *Mach. Learn.*, vol. 49, pp. 59–98, Oct. 2002.

[46] M. Muharram and G. D. Smith, "Evolutionary constructive induction," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1518–1528, Nov. 2005.

[47] H. Guo and A. K. Nandi, "Breast cancer diagnosis using genetic programming generated feature," in *Proc. IEEE Workshop Mach. Learn. Signal Process.*, Sep. 2005, pp. 215–220.

[48] H. Vafaie and K. De Jong, "Genetic algorithms as a tool for restructuring feature space representations," in *Proc. 7th IEEE Int. Conf. Tools Artif. Intell.*, Nov. 1995, pp. 8–11.

[49] K. Krawiec, "Genetic programming-based construction of features for machine learning and knowledge discovery tasks," *Genet. Program. Evolvable Mach.*, vol. 3, pp. 329–343, Dec. 2002.

[50] M. Smith and L. Bull, "Improving the human readability of features constructed by genetic programming," in *Proc. Genet. Evol. Comput. (GECCO)*, Jan. 2007, pp. 1694–1701.

[51] M. Affenzeller, S. Winkler, S. Wagner, and A. Beham, *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*. Boca Raton, FL, USA: CRC Press, Jan. 2009.

[52] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *J. Mach. Learn. Res.*, vol. 3, pp. 1289–1305, Mar. 2003.

[53] S. Fakhraei, H. Soltanian-Zadeh, and F. Fotouhi, "Bias and stability of single variable classifiers for feature ranking and selection," *Expert Syst. Appl.*, vol. 41, no. 15, pp. 6945–6958, Nov. 2014.

[54] Q. Gu, Z. Li, and J. Han, "Generalized Fisher score for feature selection," in *Proc. 27th Conf. Uncertainty Artif. Intell. (UAI)*, 2011, pp. 266–273.

[55] H. R. Li, T. Jiang, and K. Zhang, "Efficient and robust feature extraction by maximum margin criterion," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 157–165, Jan. 2006.

[56] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *Proc. 18th Int. Conf. Neural Inf. Process. Syst.*, 2005, pp. 507–514.

[57] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, 2006.

[58] M. Verleysen, F. Rossi, and D. François, "Advances in feature selection with mutual information," in *Similarity-Based Clustering*. Berlin, Germany: Springer, 2009, pp. 52–69.

[59] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann, 2011.

[60] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*. *The Wadsworth and Brooks-Cole Statistics-Probability Series*. New York, NY, USA: Taylor & Francis, 1984.

[61] C. Strobl, A.-L. Boulesteix, and T. Augustin, "Unbiased split selection for classification trees based on the Gini index," *Comput. Statist. Data Anal.*, vol. 52, no. 1, pp. 483–501, 2007.

[62] L. E. Raileanu and K. Stoffel, "Theoretical comparison between the Gini index and information gain criteria," *Ann. Math. Artif. Intell.*, vol. 41, no. 1, pp. 77–93, May 2004.

[63] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 856–863.

[64] X. Wang and O. Gotoh, "A robust gene selection method for microarray-based cancer classification," *Cancer Informat.*, vol. 9, pp. 15–30, Feb. 2010.

[65] J. Dai and Q. Xu, "Attribute selection based on information gain ratio in fuzzy rough set theory with application to tumor classification," *Appl. Soft Comput.*, vol. 13, no. 1, pp. 211–221, 2013.

[66] D. R. Wijaya, R. Sarno, and E. Zulaika, "Information quality ratio as a novel metric for mother wavelet selection," *Chemometric Intell. Lab. Syst.*, vol. 160, pp. 59–71, Jan. 2017.

[67] O. Krakovska, G. Christie, A. Sixsmith, M. Ester, and S. Moreno, "Performance comparison of linear and non-linear feature selection methods for the analysis of large survey datasets," *PLoS ONE*, vol. 14, no. 3, pp. 1–17, Mar. 2019.

[68] I. Kononenko, "On biases in estimating multi-valued attributes," in *Proc. 14th Int. Joint Conf. Artif. Intell. (IJCAI)*, vol. 2. San Francisco, CA, USA: Morgan Kaufmann, 1995, pp. 1034–1040.

[69] D. A. Bell and H. Wang, "A formalism for relevance and its application in feature subset selection," *Mach. Learn.*, vol. 41, no. 2, pp. 175–195, Nov. 2000.

[70] J. Biesiada and W. Duch, "Feature selection for high-dimensional data—A Pearson redundancy based filter," in *Computer Recognition Systems 2*, vol. 45. Oct. 2007, pp. 242–249.

[71] H. Liu and H. Motoda, *Computational Methods of Feature Selection*. Boca Raton, FL, USA: CRC Press, 2007.

[72] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.

[73] J. Wang, J.-M. Wei, Z. Yang, and S.-Q. Wang, "Feature selection by maximizing independent classification information," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 828–841, Apr. 2017.

[74] L. Wang, N. Zhou, and F. Chu, "A general wrapper approach to selection of class-dependent features," *IEEE Trans. Neural Netw.*, vol. 19, no. 7, pp. 1267–1278, Jul. 2008.

[75] P. Somol, P. Pudil, J. Novovičová, and P. Paclík, "Adaptive floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 20, nos. 11–13, pp. 1157–1163, Nov. 1999.

[76] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 15, no. 11, pp. 1119–1125, 1994.

[77] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "A methodology for feature selection using multiobjective genetic algorithms for handwritten digit string recognition," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 17, no. 6, pp. 903–929, Sep. 2003.

[78] D. Jesenko, M. Mernik, B. Žalik, and D. Mongus, "Two-level evolutionary algorithm for discovering relations between nodes' features in a complex network," *Appl. Soft Comput.*, vol. 56, pp. 82–93, Jul. 2017.

[79] L.-Y. Chuang, H.-W. Chang, C.-J. Tu, and C.-H. Yang, "Improved binary PSO for feature selection using gene expression data," *Comput. Biol. Chem.*, vol. 32, no. 1, pp. 29–38, 2008.

[80] M. Allam and M. Nandhini, "Optimal feature selection using binary teaching learning based optimization algorithm," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 32, no. 10, pp. 1–13, Dec. 2020.

[81] P. Zhao and B. Yu, "On model selection consistency of lasso," *J. Mach. Learn. Res.*, vol. 7, pp. 2541–2563, Nov. 2006.

[82] P. Buteneers, K. Caluwaerts, J. Dambre, D. Verstraeten, and B. Schrauwen, "Optimized parameter search for large datasets of the regularization parameter and feature selection for ridge regression," *Neural Process. Lett.*, vol. 38, no. 3, pp. 403–416, Dec. 2013.

[83] S. Perkins, K. Lacker, and J. Theiler, "Grafting: Fast, incremental feature selection by gradient descent in function space," *J. Mach. Learn. Res.*, vol. 3, pp. 1333–1356, Mar. 2003.

[84] J. C. Ang, A. Mirzal, H. Haron, and H. N. A. Hamed, "Supervised, unsupervised, and semi-supervised feature selection: A review on gene selection," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 13, no. 5, pp. 971–989, Sep. 2016.

[85] M. Azad, I. Chikalov, and M. Moshkov, "Representation of knowledge by decision trees for decision tables with multiple decisions," *Procedia Comput. Sci.*, vol. 176, pp. 653–659, Jan. 2020.

[86] R. E. Chandler, "On the use of generalized linear models for interpreting climate variability," *Environmetrics*, vol. 16, no. 7, pp. 699–715, 2005.

[87] C. C. Aggarwal, *Data Mining: The Textbook*. Heidelberg, Germany: Springer, 2015.

[88] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics*, vol. 8, no. 8, p. 832, Jul. 2019.

[89] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5188–5196.

[90] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3319–3327.

[91] Q.-S. Zhang and S.-C. Zhu, "Visual interpretability for deep learning: A survey," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 27–39, Feb. 2018.
[92] P. V. Johnsen *et al.*, "A new method for exploring gene–gene and gene–environment interactions in GWAS with tree ensemble methods and SHAP values," *BMC Bioinf.*, vol. 22, pp. 1–29, May 2021.
[93] E. Štrumbelj and I. Kononenko, "Explaining prediction models and individual predictions with feature contributions," *Knowl. Inf. Syst.*, vol. 41, no. 3, pp. 647–665, Dec. 2014.
[94] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, vol. 30. Red Hook, NY, USA: Curran Associates, 2017, pp. 4765–4774.
[95] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, "Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods," in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, Feb. 2020, pp. 180–186.
[96] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. ICML*, 2017, pp. 3145–3153.
[97] M. Aubry and B. Russell, "Understanding deep features with computer-generated imagery," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 2875–2883.
[98] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision—ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham, Switzerland: Springer, 2014, pp. 818–833.
[99] P. E. Rauber, S. G. Fadel, A. X. Falcão, and A. C. Telea, "Visualizing the hidden activity of artificial neural networks," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 1, pp. 101–110, Jan. 2017.
[100] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digit. Signal Process.*, vol. 73, pp. 1–15, Feb. 2018.
[101] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2660–2673, Nov. 2017.
[102] D. Jung, J. Lee, J. Yi, and S. Yoon, "iCaps: An interpretable classifier via disentangled capsule networks," in *Computer Vision—ECCV 2020* (Lecture Notes in Computer Science), vol. 12364, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Glasgow, U.K.: Springer, Aug. 2020, pp. 314–330.
[103] A. Donner and J. J. Koval, "The estimation of intraclass correlation in the analysis of family data," *Biometrics*, vol. 36, no. 1, pp. 19–25, 1980.
[104] K. O. McGraw and S. Wong, "Forming inferences about some intraclass correlation coefficients," *Psychol. Methods*, vol. 1, no. 1, pp. 30–46, Mar. 1996.
[105] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.
[106] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.
[107] L. Wang, *Data Mining With Computational Intelligence*. Berlin, Germany: Springer-Verlag, 2009.
[108] R. J. Urbanowicz, M. Meeker, W. L. Cava, R. S. Olson, and J. H. Moore, "Relief-based feature selection: Introduction and review," *J. Biomed. Inform.*, vol. 85, pp. 189–203, Sep. 2018.
[109] H. F. Inman and E. L. Bradley, Jr., "The overlapping coefficient as a measure of agreement between probability distributions and point estimation of the overlap of two normal densities," *Commun. Statist., Theory Methods*, vol. 18, no. 10, pp. 3851–3874, Jan. 1989.
[110] H. Sun and S. Wang, "Measuring the component overlapping in the Gaussian mixture model," *Data Mining Knowl. Discovery*, vol. 23, no. 3, pp. 479–502, 2011.
[111] D. Mongus and B. Žalik, "An efficient approach to 3D single tree-crown delineation in LiDAR data," *ISPRS J. Photogramm. Remote Sens.*, vol. 108, pp. 219–233, Oct. 2015.
[112] D. Cornaz, F. Furini, M. Lacroix, E. Malaguti, A. R. Mahjoub, and S. Martin, "The vertex k-cut problem," *Discrete Optim.*, vol. 31, pp. 8–28, Feb. 2019.
[113] A. Berger, A. Grigoriev, and R. van der Zwaan, "Complexity and approximability of the k-way vertex cut," *Networks*, vol. 63, no. 2, pp. 170–178, Mar. 2014.
[114] M. D. Biha and M.-J. Meurs, "An exact algorithm for solving the vertex separator problem," *J. Global Optim.*, vol. 49, no. 3, pp. 425–434, Mar. 2011.
[115] E. Alpaydin, *Introduction to Machine Learning*. Cambridge, MA, USA: MIT Press, 2010.
[116] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, Irvine, CA, USA, Tech. Rep., 2017.
[117] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, and S. García, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, Jan. 2011.
[118] V. Czitrom, "One-factor-at-a-time versus designed experiments," *Amer. Statist.*, vol. 53, no. 2, pp. 126–131, May 1999.
[119] F. J. Moreno-Barea, J. M. Jerez, and L. Franco, "Improving classification accuracy using data augmentation on small data sets," *Expert Syst. Appl.*, vol. 161, Dec. 2020, Art. no. 113696.
[120] T. Wang, H. Su, and J. Li, "DWS-MKL: Depth-width-scaling multiple kernel learning for data classification," *Neurocomputing*, vol. 411, pp. 455–467, Oct. 2020.
[121] P. Ribéreau-gayon, D. Dubourdieu, B. Donéche, and A. Lonvaud-Funel, *Dry Extract and Minerals*. Hoboken, NJ, USA: Wiley, 2006, ch. 4, pp. 91–108.
[122] M. S. Coli, A. G. P. Rangel, E. S. Souza, M. F. Oliveira, and A. C. N. Chiaradia, "Chloride concentration in red wines: Influence of terroir and grape type," *Food Sci. Technol., Campinas*, vol. 35, no. 1, pp. 95–99, Mar. 2015.
[123] R. Walker *et al.*, "Salinity effects on vines and wines," *Bull. de l'OIV*, vol. 76, pp. 200–227, Jan. 2003.
[124] R. D. Mattes and D. DiMeglio, "Ethanol perception and ingestion," *Physiol. Behav.*, vol. 72, nos. 1–2, pp. 217–229, Jan. 2001.
[125] M. C. Goldner, M. C. Zamora, P. D. L. Lira, H. Gianninoto, and A. Bandoni, "Effect of ethanol level in the perception of aroma attributes and the detection of volatile compounds in red wine," *J. Sensory Stud.*, vol. 24, no. 2, pp. 243–257, Apr. 2009.
[126] J. Fischer, *The Evaluation of Wine: A Comprehensive Guide to the Art of Wine Tasting*, 1st ed. Bloomington, IN, USA: iUniverse, 2001.
[127] G. Pickering and G. J. Pickering, "The influence of ethanol and pH on the taste and mouth-feel sensations elicited by red wine," *J. Food Agricult. Environ.*, vol. 6, pp. 143–150, Jul. 2008.
[128] R. Gawel, P. A. Smith, S. Cicerale, and R. Keast, "The mouthfeel of white wine," *Crit. Rev. Food Sci. Nutrition*, vol. 58, no. 17, pp. 2939–2956, Nov. 2018.
[129] R. Boulton, V. Singleton, L. Bisson, and R. Kunkee, *Principles and Practices of Winemaking*, 3rd ed. New York, NY, USA: Springer, 1998.
[130] M. V. Moreno-Arribas and M. C. Polo, "Winemaking biochemistry and microbiology: Current knowledge and future trends," *Crit. Rev. Food Sci. Nutrition*, vol. 45, no. 4, pp. 265–286, Jun. 2005.
[131] T. M. Monro *et al.*, "Sensing free sulfur dioxide in wine," *Sensors*, vol. 12, no. 8, pp. 10759–10773, Aug. 2012.
[132] R. F. Guerrero and E. Cantos-Villar, "Demonstrating the efficiency of sulphur dioxide replacements in wine: A parameter review," *Trends Food Sci. Technol.*, vol. 42, no. 1, pp. 27–43, Mar. 2015.

Dino Vlahek (Member, IEEE) is currently pursuing the Ph.D. degree in computer science with the Faculty of Electrical Engineering and Computer Science, Institute of Computer Science, University of Maribor, Maribor, Slovenia.

He is currently a Researcher with the Laboratory for Geospatial Modelling, Multimedia, and Artificial Intelligence, University of Maribor. His research interests are feature learning, data analytics, and model interpretation.

Domen Mongus (Member, IEEE) is currently an Associate Professor with the Faculty of Electrical Engineering and Computer Science, University of Maribor, Maribor, Slovenia. His research interests include environmental intelligence, data fusion, feature learning, and remote sensing data processing.

Dr. Mongus was a Long-Time Member of the Executive Committee of the European Umbrella Organization for Geographic Information and the Vice-President of the Program Board of Strategic Research and Innovation Partnership in Smart Cities. He is also a member of the Executive Committee of Geographical Information Systems International Group (GISIG). For his extensive research work in the respected fields, he was, among others, named Young Scientist of Danube Region and received the highest institutional academic award for exceptional contributions to the scientific and pedagogical reputation and excellence of the University of Maribor.