Iterative RELIEF for Feature Weighting: Algorithms, Theories and Applications

Yijun Sun[†,‡]
[†]Interdisciplinary Center for Biotechnology Research
[‡]Department of Electrical and Computer Engineering
University of Florida
Gainesville, FL 32610-3622

Abstract: RELIEF is considered one of the most successful algorithms for assessing the quality of features. In this paper, we propose a set of new feature weighting algorithms that perform significantly better than RELIEF, without introducing a large increase in computational complexity. Our work starts from a mathematical interpretation of the seemingly heuristic RELIEF algorithm as an online method solving a convex optimization problem with a margin-based objective function. This interpretation explains the success of RELIEF in real applications, and enables us to identify and address its following weaknesses. RELIEF makes an implicit assumption that the nearest neighbors found in the original feature space are the ones in the weighted space, and RELIEF lacks a mechanism to deal with outlier data. We propose an iterative RELIEF (I-RELIEF) algorithm to alleviate the deficiencies of RELIEF by exploring the framework of the Expectation-Maximization algorithm. We extend I-RELIEF to multiclass settings by using a new multiclass margin definition. To reduce computational costs, an online learning algorithm is also developed. Convergence analysis of the proposed algorithms is presented. The results of large-scale experiments on the UCI and microarray datasets are reported, which demonstrate the effectiveness of the proposed algorithms, and verify the presented theoretical results.

Keywords: Feature weighting, feature selection, RELIEF, iterative algorithm, DNA microarray, classification.

IEEE Transactions on Pattern Analysis and Machine Intelligence
Submitted in September 2006, accepted with minor revision in November 2006

The preliminary results of the work appeared in [1].

Please address all correspondence to: Dr. Yijun Sun, Interdisciplinary Center for Biotechnology Research, University of Florida, P.O. Box 103622, Gainesville, FL 32610-3622, USA. E-mail: sun@dsp.ufl.edu.

I. INTRODUCTION

Feature selection is one of the fundamental problems in machine learning. Not only can its proper design reduce system complexity and processing time, but it can also enhance system performance in many cases. It becomes even more critical to the success of a machine learning algorithm in problems involving a large amount of irrelevant features. One example is DNA microarray data [2], [3], where the number of features (genes) is typically on the order of thousands or even tens of thousands, while the number of the useful genes is believed to be in the range of tens and hundreds. Due to the high costs of collecting a large number of patient data, one usually only has tens or at most hundreds of samples for training. With limited training samples, selecting useful features for these kinds of problems poses a serious challenge to the existing feature selection algorithms.

The research on feature selection is very active in the past decade [4], [5], [6], [7], [8]. The existing feature selection algorithms can be generally categorized as *wrapper* or *filter* methods based on criterion functions used in searching for informative features [9]. In wrapper methods, the performance of a learning algorithm is used to evaluate the goodness of selected feature subsets, whereas in filter methods criterion functions evaluate feature subsets by their information content, typically interclass distance or information-theoretic measures, rather than optimizing the performance of any specified learning algorithm directly. In most cases, filter methods are computationally much more efficient but perform worse than wrapper methods. Given a criterion function, feature selection is reduced to a search problem. An exhaustive search is optimal but only works when the number of features is not too large, as it quickly becomes computationally infeasible with the increase of problem size. Some heuristic combinatorial searches, such as forward and/or backward selection [5], are usually employed. These algorithms have shown some successes in practical applications. However, none of them can provide any guarantee of optimality. For more detailed discussions, interested readers can refer to [7] and the references therein.

The computational issue of combinatorial search can to some extent be alleviated by using a feature weighting strategy. Allowing feature weights to take real-valued numbers instead of binary ones enables the employment of some well-established optimization techniques, and thus allows for efficient algorithmic implementation. Among the existing feature weighting algorithms, the RELIEF algorithm [10] is considered one of the most successful ones due to its simplicity and effectiveness [11]. The pseudo-code of the algorithm is presented in Fig. 1. The key idea of RELIEF is to iteratively estimate feature weights according to their ability to discriminate between neighboring patterns. In each iteration, a pattern $\mathbf{x}$ is randomly selected, and then two nearest neighbors of $\mathbf{x}$ are found, one from the same class (termed the *nearest hit* or NH) and the other from a different class (termed the *nearest miss* or NM). The weight of the i-th feature is then updated: $w_i = w_i + |\mathbf{x}^{(i)} - \text{NM}^{(i)}(\mathbf{x})| - |\mathbf{x}^{(i)} - \text{NH}^{(i)}(\mathbf{x})|$. RELIEF was extended to handle noisy and missing data in [12], wherein a probabilistic interpretation for RELIEF was also provided, stating that the learned weights approximate the following probability:

$$w_i = P(\text{different value of } i\text{-th feature} \mid \text{NM}) - P(\text{different value of } i\text{-th feature} \mid \text{NH}) \ . \tag{1}$$

It is, however, not easy to understand this interpretation since both probabilities in Eq. (1) are not well defined. To further examine the problem, in Section II we present a novel interpretation for RELIEF from the optimization perspective. We prove that RELIEF is an online algorithm that solves a convex optimization problem with a margin-based objective function. The margin is defined based on a 1-NN

RELIEF Algorithm

(1) **Initialization**: given $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$, set $w_i = 0, \ \ 1 \leq i \leq I$, number of iterations T;
(2) **for** $t = 1 : T$
  (3) Randomly select a pattern $\mathbf{x}$ from $\mathcal{D}$;
  (4) Find the nearest hit $\mathrm{NH}(\mathbf{x})$ and miss $\mathrm{NM}(\mathbf{x})$ of $\mathbf{x}$;
  (5) **for** $i = 1 : I$
    (6) Compute: $w_i = w_i + |\mathbf{x}^{(i)} - \mathrm{NM}^{(i)}(\mathbf{x})| - |\mathbf{x}^{(i)} - \mathrm{NH}^{(i)}(\mathbf{x})|$;
  (7) **end**
(8) **end**

Fig. 1. Pseudo-code of RELIEF [10]

(one-nearest-neighbor) classifier [13]. Therefore, compared with filter methods, RELIEF usually performs better due to the performance feedback of the nonlinear classifier in search for informative features; compared with conventional wrapper methods, by optimizing a convex problem, RELIEF avoids *any* exhaustive or heuristic combinatorial search and thus can be implemented very efficiently. These two merits make RELIEF particularly suitable for large-scale problems such as microarray data analysis.

The new interpretation of RELIEF enables us to identify and address some weaknesses of the algorithm. One major drawback of RELIEF is that the nearest neighbors are defined in the original feature space, which are highly unlikely to be the ones in the weighted space. Moreover, RELIEF lacks a mechanism to deal with outlier data. In the presence of a large number of irrelevant features and mislabeling, the solution quality of RELIEF can be severely degraded. To mitigate these problems, in Section III, we propose a new feature weighting algorithm, referred to as I-RELIEF, by following the principle of the Expectation-Maximization (EM) algorithm [14]. I-RELIEF treats the nearest neighbors and identity of a pattern as hidden random variables, and iteratively estimates feature weights until convergence. We provide a convergence theorem for I-RELIEF, which shows that under certain conditions I-RELIEF converges to a unique solution irrespective of initial starting points. We also extend I-RELIEF to multiclass problems. In Section IV, by using the fact that RELIEF optimizes a margin-based objective function, we propose a new multiclass RELIEF algorithm using a new multiclass margin definition. We also consider online learning for I-RELIEF. The new proposed I-RELIEF algorithms are based on batch learning. In the case where there exist a large amount of training samples, online learning is computationally much more attractive. We develop an online I-RELIEF algorithm in Section V, wherein a convergence theorem is also provided. To verify the effectiveness of the newly proposed algorithms and confirm the theoretical results established in this paper, we conduct a large-scale experiment in Section VII on nine UCI datasets and six microarray datasets. We finally conclude the paper in Section VIII.

A. Main Contributions

The main contributions of the paper are twofold:

- First, in algorithmic aspects, starting from a new interpretation of RELIEF, we propose a set of

feature weighting algorithms. The effectiveness of those algorithms, in terms of solution quality and computational efficiency, is experimentally demonstrated on a wide variety of datasets. Considering the increased demand for analyzing data with large feature dimensionality in some emerging domains such as bioinformatics, we expect widespread usage of these algorithms in these applications.

- Second, in theoretical aspects, this paper may provide a new direction of feature selection research in addition to providing some new algorithms. Feature selection plays a critical role in machine learning. Yet, as opposed to classifier design, it still to date lacks rigorous theoretical treatment. This is largely due to the difficulty in defining an objective function that can be easily optimized by some well-established optimization techniques. It is particularly true for wrapper methods that use a nonlinear classifier to evaluate the goodness of selected feature subsets. The crisp partition of a feature set and the nonlinearity of a classification function make the resulting objective function non-convex and even non-differentiable. For this reason, most feature selection algorithms rely on heuristic search. The I-RELIEF algorithms has a clearly defined objective function and can be solved through numerical analysis instead of combinatorial search, and thus presents a promising direction for a more rigorous treatment of feature selection problems.

Before moving on to the main body of the paper, we make a brief discussion on feature redundance. It is reported that redundant features can deteriorate classification performance [9]. Therefore, for classification purposes, removing redundant features is necessary. However, in some applications such as DNA microarray, some researchers have pointed out that the identification of a small gene subset with good predictive power may not be sufficient to provide significant insight into the understanding and modeling of the relationship between genes and certain diseases [15]. Redundant (or co-regulated) genes may provide biologists with some useful side information. Moreover, in the presence of a huge number of irrelevant features, removing both irrelevant and redundant features simultaneously could make the resulting algorithm complicated. Therefore, we suggest performing the task in two stages: the first stage tries to recover all of the relevant features, and the second stage, examining a much smaller feature subset, performs some computationally more expensive algorithms such as wrapper methods to remove redundant features if the ultimate goal is for classification. Designing an accurate feature weighting algorithm that can be used in the first stage is the focus of this paper.

II. Optimization Approach to RELIEF Algorithm

In this section, we provide a mathematical interpretation for the seemingly heuristic RELIEF algorithm. Let $\mathcal{D}=\{(\mathbf{x}_n, y_n)\}_{n=1}^N \in \mathcal{X}\times\mathcal{Y}$ denote a training dataset, where $\mathcal{X} \in \mathcal{R}^I$ is the pattern space, I is the feature dimensionality and $\mathcal{Y}=\{\pm 1\}$ is the label space. Following the margin definition in [16], we define the margin for a pattern $\mathbf{x}_n$ as $\rho_n = d(\mathbf{x}_n - \mathrm{NM}(\mathbf{x}_n)) - d(\mathbf{x}_n - \mathrm{NH}(\mathbf{x}_n))$, where $\mathrm{NM}(\mathbf{x}_n)$ and $\mathrm{NH}(\mathbf{x}_n)$ are the nearest miss and hit of pattern $\mathbf{x}_n$, respectively, and $d(\cdot)$ is a distance function. For the moment, we define $d(\mathbf{x}) = \sum_i |x_i|$, which can be extended to other distance functions. Note that $\rho_n > 0$ if and only if $\mathbf{x}_n$ is correctly classified by a 1-NN classifier. One natural idea is to scale each feature such that the averaged margin in a weighted feature space is maximized:

$$\begin{aligned} \max_{\mathbf{w}} \quad & \sum_{n=1}^{N} \rho_n(\mathbf{w}) = \max_{\mathbf{w}} \sum_{n=1}^{N} \left(\sum_{i=1}^{I} w_i |\mathbf{x}_n^{(i)} - \mathrm{NM}^{(i)}(\mathbf{x}_n)| - \sum_{i=1}^{I} w_i |\mathbf{x}_n^{(i)} - \mathrm{NH}^{(i)}(\mathbf{x}_n)| \right) , \\ \text{s.t.} \quad & \|\mathbf{w}\|_2^2 = 1, \mathbf{w} \geqslant 0 , \end{aligned} \tag{2}$$

where $\rho_n(\mathbf{w})$ is the margin of $\mathbf{x}_n$ computed with respect to $\mathbf{w}$. The constraint $\|\mathbf{w}\|_2^2 = 1$ prevents the maximization from increasing without bound, and $\mathbf{w} \geqslant 0$ ensures that the weight vector induces a distance measure. By defining $\mathbf{z} = \sum_{n=1}^{N}(|\mathbf{x}_n - \mathrm{NM}(\mathbf{x}_n)| - |\mathbf{x}_n - \mathrm{NH}(\mathbf{x}_n)|)$, where $|\cdot|$ is the point-wise absolute operator, Eq. (2) can be simplified as:

$$\begin{array}{ll} \max\limits_{\mathbf{w}} & \mathbf{w}^{\mathrm{T}}\mathbf{z} \ , \\ \text{s.t.} & \|\mathbf{w}\|_2^2 = 1, \mathbf{w} \geqslant 0 \ . \end{array} \tag{3}$$

The Lagrangian of Eq. (3) is:

$$L = -\mathbf{w}^{\mathrm{T}}\mathbf{z} + \lambda(\|\mathbf{w}\|_2^2 - 1) + \sum_{i=1}^{I}\zeta_i(-w_i) \ , \tag{4}$$

where λ and $\boldsymbol{\zeta} \geqslant 0$ are the Lagrangian multipliers. Taking the derivative of L with respect to $\mathbf{w}$ and setting it to zero yields:

$$\frac{\partial L}{\partial \mathbf{w}} = -\mathbf{z} + 2\lambda\mathbf{w} - \boldsymbol{\zeta} = 0 \quad \Rightarrow \quad \mathbf{w} = \frac{1}{2\lambda}(\mathbf{z} + \boldsymbol{\zeta}) \tag{5}$$

Below, we derive a closed-form solution for $\mathbf{w}$. In the derivation, we make an assumption that there exists an $i, 1 \leq i \leq I$, such that $z_i > 0$. This assumption is very weak since if it does not hold (i.e., $\mathbf{z} \leq \mathbf{0}$), it simply says that on average the distance between a pattern and its nearest miss is larger than the distance of the pattern from its nearest hit, which is very rare in real applications. In this case, machine learning algorithms that make decisions based on neighborhood information (i.e., RBF, and SVM with RBF kernel) will perform poorly. With the above assumption, we prove that $\lambda > 0$ by contradiction. Suppose $\lambda < 0$. Since there exists $z_i > 0$, then $z_i + \zeta_i > 0$. We would have $w_i < 0$, which contradicts the constraint $\mathbf{w} \geq 0$. By using the Karush-Kuhn-Tucker (KKT) condition [17], namely $\sum_i \zeta_i w_i = 0$, it is easy to verify the following three cases:

- Case 1: $z_i = 0 \Rightarrow \zeta_i = 0$ and $w_i = 0$;
- Case 2: $z_i > 0 \Rightarrow z_i + \zeta_i > 0 \Rightarrow w_i > 0 \Rightarrow \zeta_i = 0$;
- Case 3: $z_i < 0 \Rightarrow \zeta_i > 0 \Rightarrow w_i = 0 \Rightarrow z_i = -\zeta_i$.

It immediately follows that the optimal solution of $\mathbf{w}$ can be calculated in a closed form:

$$w_i = \begin{cases} 0 & \text{if } z_i \leq 0 \\ \frac{1}{2\lambda} z_i & \text{if } z_i > 0 \end{cases}$$

and

$$\mathbf{w}^* = \mathbf{w}/\|\mathbf{w}\|_2 = (\mathbf{z})^+/\|(\mathbf{z})^+\|_2 \ , \tag{6}$$

where $(\mathbf{z})^+ = [\max(z_1, 0), \cdots, \max(z_I, 0)]^T$.

By comparing Eq. (6) with the weight update rule of RELIEF in Fig. 1, we conclude that RELIEF is an online algorithm that solves the optimization scheme of Eq. (2). This is true except when $w_i^* = 0$ for $z_i \leq 0$, which usually corresponds to irrelevant features discarded in RELIEF.

Our proof is inspired by the previous work [8] where RELIEF maximizing the averaged margin was empirically observed, but no mathematical proof was provided. From our analysis, we find that RELIEF is a feature weighting algorithm that utilizes the performance of a highly nonlinear classifier in search for useful features, yet results in a simple convex problem with a closed-form solution. This clearly explains the simplicity and effectiveness of RELIEF.

Other distance functions can also be used. If the squared Euclidean distance is used, the algorithm will be the RELIEF algorithm defined in [8]; if Euclidean distance is used, the resulting algorithm is Simba [8]. One major problem with Simba is its implementation. First, Simba returns many local maxima, which is mitigated in Simba by restarting the algorithm from several starting points. Hence, the acquisition of the global maximum is not guaranteed through its invocation. Second, Simba is a constrained nonlinear optimization problem which cannot be easily solved through conventional optimization techniques. Therefore, Simba first performs gradient ascent over the objective function while ignoring the constraints, and then projects the solution onto the constraints at the last step. It is unclear whether Simba converges.

III. Iterative RELIEF Algorithm

A. Algorithm

The new interpretation of RELIEF as an online implementation of a convex optimization problem provides some insights into its success in practical applications. More importantly, it enables us to identify some weaknesses of the algorithm and to propose some solutions to fix them. Two major drawbacks of RELIEF become clear from the objective function defined in Eq. (3). First, the nearest neighbors are defined in the original feature space, which may not be true in the weighted feature space. Second, the objective function optimized by RELIEF is actually the average margin. In the presence of outliers, some margins can take large negative values. In a highly noisy data case with a large amount of irrelevant features and/or mislabelling, the aforementioned two issues can become so severe that the performance of RELIEF may be greatly deteriorated.

A heuristic algorithm, called RELIEF-F [12], has been proposed to address the first problem. RELIEF-F averages K, instead of just one, nearest neighbors when computing the sample margins. Empirical studies have shown that RELIEF-F can achieve significant performance improvement over the original RELIEF. As for the second problem, to our knowledge, no such algorithm exists. One possible solution is to adopt the large margin concept that is used in SVM [18] and AdaBoost [19], where slack variables are defined to account for the very negative margins and a soft margin is maximized [20]. In order to solve the induced optimization problem efficiently, a good linear or quadratic programming solver is needed. Moveover, one introduces a very critical parameter that controls the trade-off between training performance and a penalty term. Using the large margin concept for feature selection seems to be an interesting direction, which we will pursue in our future work. In this paper, however, we will confine ourselves to the RELIEF framework. Below, we propose an analytic solution to address the aforementioned two issues simultaneously.

We first define two sets $\mathcal{M}_n = \{i : 1 \leq i \leq N, y_i \neq y_n\}$ and $\mathcal{H}_n = \{i : 1 \leq i \leq N, y_i = y_n, i \neq n\}$, associated with each pattern $\mathbf{x}_n$. Suppose now that for each pattern $\mathbf{x}_n$, its nearest hit and miss are known, the indices of which are saved in the set $\mathcal{S}_n = \{(s_{n1}, s_{n2})\}$, where $s_{n1} \in \mathcal{M}_n$ and $s_{n2} \in \mathcal{H}_n$. For example, $s_{n1} = 1$ and $s_{n2} = 2$ mean that the nearest miss and hit of $\mathbf{x}_n$ are $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively. We also denote $\mathbf{o} = [o_1, \cdots, o_N]^{\mathrm{T}}$ as a set of binary parameters, such that $o_n = 0$ if $\mathbf{x}_n$ is an outlier, or $o_n = 1$ otherwise. Here, we use the term "outliers" to refer to mislabeled samples or samples highly corrupted by noise. The objective function we want to optimize may be formulated as:

$$C(\mathbf{w}) = \frac{1}{N} \sum_{\{n=1, o_n=1\}}^{N} \left(\|\mathbf{x}_n - \mathbf{x}_{s_{n1}}\|_{\mathbf{w}} - \|\mathbf{x}_n - \mathbf{x}_{s_{n2}}\|_{\mathbf{w}} \right) , \tag{7}$$

where $\|\mathbf{x}\|_{\mathbf{w}} = \sum_i w_i|x_i|$. This objective function can be easily optimized by using the conclusion drawn in Section II. Of course, we do not know the set $\mathcal{S} = \{\mathcal{S}_n\}_{n=1}^N$ and the vector $\mathbf{o}$. However, if we assume the elements of $\{\mathcal{S}_n\}_{n=1}^N$ and $\mathbf{o}$ are random variables, we can proceed by deriving the probability distributions of the unobserved data. We first make a guess on the weight $\mathbf{w}$. By using the pairwise distances that have been computed when searching for the nearest hits and misses, the probability of the i-th data point being the nearest miss of $\mathbf{x}_n$ can be defined as:

$$P_m(i|\mathbf{x}_n, \mathbf{w}) = \frac{f(\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}})}{\sum_{j\in\mathcal{M}_n} f(\|\mathbf{x}_n - \mathbf{x}_j\|_{\mathbf{w}})}, \forall i \in \mathcal{M}_n \ . \tag{8}$$

Similarly, the probability of the i-th data point being the nearest hit of $\mathbf{x}_n$ is:

$$P_h(i|\mathbf{x}_n, \mathbf{w}) = \frac{f(\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}})}{\sum_{j\in\mathcal{H}_n} f(\|\mathbf{x}_n - \mathbf{x}_j\|_{\mathbf{w}})}, \forall i \in \mathcal{H}_n \ , \tag{9}$$

and the probability of $\mathbf{x}_n$ being an outlier can be defined as:

$$P_o(o_n = 0|\mathcal{D}, \mathbf{w}) = \frac{\sum_{i\in\mathcal{M}_n} f(\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}})}{\sum_{\mathbf{x}_i\in\mathcal{D}\setminus\mathbf{x}_n} f(\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}})} \ , \tag{10}$$

where $f(\cdot)$ is a kernel function. One commonly used example is $f(d) = \exp(-d/\sigma)$, where the kernel width σ is a user defined parameter. Throughout the paper, the exponential kernel is used. Other kernel functions can also be used, and the descriptions of their properties can be found in [21].

Now we are ready to derive the following iterative algorithm. Although we adopt the idea of the EM algorithm [14] that treats the unobserved data as random variables, it should be noted that the following method is not an EM algorithm since the objective function in Eq. (7) is not a likelihood. For brevity of notation, we define $\alpha_{i,n} = P_m(i|\mathbf{x}_n, \mathbf{w}^{(t)}), \beta_{i,n} = P_h(i|\mathbf{x}_n, \mathbf{w}^{(t)}), \gamma_n = 1 - P_o(o_n = 0|\mathcal{D}, \mathbf{w}^{(t)}), \mathcal{W} = \{\mathbf{w} : \|\mathbf{w}\|_2^2 = 1, \mathbf{w} \geq 0\}, \mathbf{m}_{i,n} = |\mathbf{x}_n - \mathbf{x}_i|$ if $i \in \mathcal{M}_n$, and $\mathbf{h}_{i,n} = |\mathbf{x}_n - \mathbf{x}_i|$ if $i \in \mathcal{H}_n$.

Step 1: After the t-th iteration, the Q function is calculated as[1]:

$$\begin{aligned} Q(\mathbf{w}|\mathbf{w}^{(t)}) &= \mathrm{E}_{\{\mathcal{S},\mathbf{o}\}}[C(\mathbf{w})] \ , \\ &\approx \frac{1}{N}\sum_{n=1}^{N} \gamma_n \left(\sum_{i\in\mathcal{M}_n} \alpha_{i,n}\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}} - \sum_{i\in\mathcal{H}_n} \beta_{i,n}\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}} \right) \ . \end{aligned} \tag{11}$$

[1]More rigorously, even though $\mathbf{x}_n$ may not be an outlier, there may be outliers present in $\mathcal{M}_n$ and $\mathcal{H}_n$ that should be considered in computing the Q function. However, mislabeling in $\mathbf{o}$ has a much larger impact on calculating margins than that in $\mathcal{M}_n$ and $\mathcal{H}_n$, which is explained as follows: consider the term computed in Eq. (11): $\tilde{\rho_n} = \sum_{i\in\mathcal{M}_n} \alpha_{i,n}\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}} - \sum_{i\in\mathcal{H}_n} \beta_{i,n}\|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}}$. Suppose $\mathbf{x}_n$ is an outlier. Then the value of the margin is completely wrong; now suppose $\mathbf{x}_n$ is not an outlier, but one element in $\mathcal{M}_n$ is. Since the first term in $\tilde{\rho_n}$ is computed by averaging over the entire set of $\mathcal{M}_n$, there only has a little impact on the margin value. Since I-RELIEF is a data pre-processing algorithm and performs very well on a wide variety of datasets (c.f. Sec. VII), accounting for mislabeling in $\mathcal{M}_n$ and $\mathcal{H}_n$ may not improve the performance significantly, but will complicate the algorithm implementation and the subsequent analyses substantially.

I-RELIEF Algorithm

(1) **Initialization**: given $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$, set $\mathbf{w}^{(0)} = \mathbf{1}/I$, number of iterations T, kernel width σ, stop criterion θ;

(2) **for** $t = 1 : T$

(3) Calculate pairwise distances with respect to $\mathbf{w}^{(t-1)}$;

(4) Calculate P_m, P_h and P_o as Eqs. (8), (9) and (10);

(5) Update weights as Eq. (12);

(6) **If** $\|\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)}\| < \theta$, **break**, **end**

(7) **end**

Fig. 2. Pseudo-code of I-RELIEF

Step 2: The re-estimation of $\mathbf{w}$ in the $(t+1)$-th iteration is:

$$\begin{aligned}
\mathbf{w}^{(t+1)} &= \arg\max_{\mathbf{w}\in\mathcal{W}} Q(\mathbf{w}|\mathbf{w}^{(t)}) \ , \\
&\approx \arg\max_{\mathbf{w}\in\mathcal{W}} \frac{1}{N}\sum_{n=1}^{N}\gamma_n\left(\sum_{i\in\mathcal{M}_n}\alpha_{i,n}\|\mathbf{x}_n-\mathbf{x}_i\|_{\mathbf{w}} - \sum_{i\in\mathcal{H}_n}\beta_{i,n}\|\mathbf{x}_n-\mathbf{x}_i\|_{\mathbf{w}}\right) \ , \\
&= \arg\max_{\mathbf{w}\in\mathcal{W}} \frac{1}{N}\sum_{n=1}^{N}\gamma_n\left(\sum_{i\in\mathcal{M}_n}\alpha_{i,n}\sum_j w_j m_{i,n}^j - \sum_{i\in\mathcal{H}_n}\beta_{i,n}\sum_j w_j h_{i,n}^j\right) \ , \\
&= \arg\max_{\mathbf{w}\in\mathcal{W}} \frac{1}{N}\sum_{n=1}^{N}\gamma_n\left(\sum_j w_j \underbrace{\sum_{i\in\mathcal{M}_n}\alpha_{i,n}m_{i,n}^j}_{\bar{m}_n^j} - \sum_j w_j \underbrace{\sum_{i\in\mathcal{H}_n}\beta_{i,n}h_{i,n}^j}_{\bar{h}_n^j}\right) \ , \\
&= \arg\max_{\mathbf{w}\in\mathcal{W}} \mathbf{w}^T \underbrace{\frac{1}{N}\sum_{n=1}^{N}\gamma_n(\bar{\mathbf{m}}_n - \bar{\mathbf{h}}_n)}_{\boldsymbol{\nu}} = (\boldsymbol{\nu})^+/\|(\boldsymbol{\nu})^+\|_2 \ .
\end{aligned} \tag{12}$$

The above two steps iterate alternatingly until convergence, i.e. $\|\mathbf{w}^{(t+1)} - \mathbf{w}^{(t)}\| < \theta$.

We name the above algorithm as iterative RELIEF, or I-RELIEF for short. The pseudo-code of I-RELIEF is presented in Fig. 2. Since Eqs. (8), (9) and (10) return us with reasonable estimates of P_m, P_h and P_o, respectively, and the re-estimation of $\mathbf{w}$ is a convex optimization problem, we expect a good convergence behavior and reasonable performance from I-RELIEF. In the following subsection, we provide a convergence analysis for I-RELIEF.

B. Convergence Analysis

We begin by studying the asymptotic behavior of I-RELIEF. If $\sigma \to +\infty$, then

$$\lim_{\sigma\to+\infty} P_m(i|\mathbf{x}_n, \mathbf{w}) = \frac{1}{|\mathcal{M}_n|} \tag{13}$$

for every $\mathbf{w} \in \mathcal{W}$ since $\lim_{\sigma \to +\infty} f(d) = 1$. On the other hand, for a given $\mathbf{w}$, if $\sigma \to 0$, by assuming that for every $n, d_{in} \triangleq \|\mathbf{x}_n - \mathbf{x}_i\|_{\mathbf{w}} \neq d_{jn}$ if $i \neq j$, we have

$$\begin{aligned} \lim_{\sigma \to 0} P_m(i|\mathbf{x}_n, \mathbf{w}) &= \lim_{\sigma \to 0} \exp(-d_{in}/\sigma) / \sum_{j \in \mathcal{M}_n} \exp(-d_{jn}/\sigma) \\ &= \lim_{\sigma \to 0} 1 / \sum_{j \in \mathcal{M}_n} \exp(-(d_{jn} - d_{in})/\sigma) \\ &= \begin{cases} 1 & \text{if } d_{in} = \min\limits_{j \in \mathcal{M}_n} d_{jn} \\ 0 & \text{if } d_{in} > \min\limits_{j \in \mathcal{M}_n} d_{jn} \end{cases}. \end{aligned} \tag{14}$$

$P_h(i|\mathbf{x}_n, \mathbf{w})$ and $P_o(n|\mathbf{w})$ can be computed similarly. From the above analysis, we observe that if $\sigma \to 0$, I-RELIEF is equivalent to iterating the original RELIEF (NM = NH = 1) provided that outlier removal is not considered. In our experiments, we rarely observe that the resulting algorithm converges. On the other hand, if $\sigma \to +\infty$, I-RELIEF converges in one step because the term $\boldsymbol{\nu}$ in Eq. (12) is a constant vector for any initial feature weights. This suggests that the convergence behavior and convergence rates of I-RELIEF are fully controlled by the choice of the kernel width. In the following, we present a proof by using the well-known Banach fixed point theorem. We first state the theorem without proof. For detailed proofs, we refer the reader to [22].

Definition 1. *Let $\mathcal{U}$ be a subset of a norm space $\mathcal{Z}$, and $\|\cdot\|$ is a norm defined in $\mathcal{Z}$. An operator $T : \mathcal{U} \to \mathcal{Z}$ is called a contraction operator if there exists a constant $q \in [0, 1)$ such that*

$$\|T(x) - T(y)\| \leq q\|x - y\| \tag{15}$$

for every $x, y \in \mathcal{U}$. q is called the contraction number of T.

Definition 2. *An element of a norm space $\mathcal{Z}$ is called a fixed point of $T : \mathcal{U} \to \mathcal{Z}$ if $T(x) = x$.*

Theorem 1 (Fixed Point Theorem)**.** *Let T be a contraction operator mapping a complete subset $\mathcal{U}$ of a norm space $\mathcal{Z}$ into itself. Then the sequence generated as*

$$x^{(t+1)} = T(x^{(t)}), \quad t = 0, 1, 2, \cdots \tag{16}$$

with arbitrary $x^{(0)} \in \mathcal{U}$ converges to the unique fixed point x^* of T. Moreover, the following estimation error bounds hold:

$$\begin{aligned} &\|x^{(t)} - x^*\| \leq \tfrac{q^t}{1-q}\|x^{(1)} - x^{(0)}\|, \\ \textit{and} \quad &\|x^{(t)} - x^*\| \leq \tfrac{q}{1-q}\|x^{(t)} - x^{(t-1)}\|. \end{aligned} \tag{17}$$

In order to apply the fixed point theorem to prove the convergence of I-RELIEF, the gist is to identify the contraction operator in I-RELIEF and check if all conditions in Theorem 1 are met. To this end, let $\mathcal{P} = \{\mathbf{p} : \mathbf{p} = [P_m, P_h, P_o]\}$ and we specify the two steps of I-RELIEF in a functional form as $A1 : \mathcal{W} \to \mathcal{P}, A1(\mathbf{w}) = \mathbf{p}; A2 : \mathcal{P} \to \mathcal{W}, A2(\mathbf{p}) = \mathbf{w}$. By indicating the functional composition by a circle $(\circ)$, I-RELIEF can be written as $\mathbf{w}^{(t)} = (A2 \circ A1)(\mathbf{w}^{(t-1)}) \triangleq T(\mathbf{w}^{(t-1)})$, where $T : \mathcal{W} \to \mathcal{W}$. Since $\mathcal{W}$ is a closed subset of a norm space $\mathcal{R}^I$ and complete, T is an operator mapping a complete subset $\mathcal{W}$ into itself. However, it is difficult to directly verify that T is a contraction operator satisfying the condition in Eq. (15). Noting that for $\sigma \to +\infty$, I-RELIEF converges with one step, we have $\lim_{\sigma \to +\infty} \|T(\mathbf{w}_1, \sigma) - T(\mathbf{w}_2, \sigma)\| = 0$, for every $\mathbf{w}_1, \mathbf{w}_2 \in \mathcal{W}$. Therefore, in the limit, T is a contraction operator with contraction constant

$q = 0$, that is, $\lim_{\sigma \to +\infty} q(\sigma) = 0$. Therefore, for every $\varepsilon > 0$, there exists a $\bar{\sigma}$ such that $q(\sigma) \leq \varepsilon$ whenever $\sigma > \bar{\sigma}$. By setting $\varepsilon < 1$, the resulting operator T is a contraction operator. Combining the above arguments, we establish the convergence result for I-RELIEF, which is summarized in the following theorem.

Theorem 2. *Let $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N \in \mathcal{X} \times \mathcal{Y}, \mathcal{W} = \{\mathbf{w} : \|\mathbf{w}\|_2 = 1, \mathbf{w} \geq 0\}$ and I-RELIEF be defined in Fig. 2. There exists a $\bar{\sigma}$ such that*

$$\lim_{t \to +\infty} \|\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)}\| = 0 \tag{18}$$

whenever $\sigma > \bar{\sigma}$. Moreover, for a fixed $\sigma > \bar{\sigma}$, I-RELIEF converges to a unique solution for any initial feature weights $\mathbf{w}^{(0)} \in \mathcal{W}$.

Theorem 2 ensures the convergence of I-RELIEF but does not tell us how large the kernel width should be. In our experiment (c.f. Section VII-C), we find that by using a relative large σ value, say $\sigma > 0.5$, the convergence is guaranteed. Also, the error bound in Ineq. (17) tells us that the smaller the contraction number q, the tighter the error bound and hence the faster the convergence rate. Since it is difficult to explicitly express q as a function of σ, it is difficult to prove that q monotonically decreases with σ. However, in general, a larger kernel width yields a faster convergence, which is experimentally confirmed in Section VII-C. It is also worthwhile to emphasize that unlike other machine learning algorithms such as neural networks [23], the convergence and the solution of I-RELIEF are not affected by the initial value if the kernel width is fixed. We also experimentally find that setting the initial feature weights all to be $1/I$ can only lead to slight but negligible improvement of the convergence rate compared to a randomly generated initial value.

C. Relation to RELIEF-F

RELIEF-F is a special case of I-RELIEF. To see this, we first set $P_o = 0$ since RELIEF-F has no mechanism to handle outliers. Then we set $P_m(i|\mathbf{x}_n, \mathbf{w}^{(0)})$ and $P_h(i|\mathbf{x}_n, \mathbf{w}^{(0)})$ to be $1/K$ if $\mathbf{x}_i$ is one of the K nearest misses or hits of $\mathbf{x}_n$ and 0 otherwise, respectively. Moreover, RELIEF-F only runs for one iteration. That is, in the context of I-RELIEF, RELIEF-F first makes a guess on the weights as all equal to $1/I$, and then calculates P_m and P_h as above, and finally re-estimates $\mathbf{w}$ and stops. Obviously, from our analysis, the estimate of P_m and P_h is quite rough. Kononenko [12] suggested that K be chosen as 10, which is used as a rule in practice in comparison between RELIEF-F and other algorithms. In our experiments, however, we observe that the performance of RELIEF-F is highly influenced by the value of K. Therefore, we estimate K through cross-validation using training data in our experiments.

IV. Extension to Multiclass RELIEF

In this section, we consider feature weighting for multiclass problems. Some algorithms originally designed for binary problems can be naturally extended to multiclass settings, while for others the extension is not straightforward. In particular, for wrapper methods, the extension largely depends on the capability of a classifier to handle multiclass problems [7]. In many cases, a multiclass problem is first decomposed into several binary ones [24], [25], which further increases the computational burden of a wrapper method. RELIEF, due to the use of a nearest neighbor classifier as shown previously, does not suffer from such a problem.

The RELIEF algorithm was originally designed to handle binary problems. Later, RELIEF-F was proposed in [12] to deal with multiclass problems by modifying the weight update rule (line 6 of Fig. 1) as:

$$w_i = w_i + \sum_{\{c \in \mathcal{Y}, c \neq y(\mathbf{x})\}} \frac{P(c)}{1 - P(c)} |\mathbf{x}^{(i)} - \mathrm{NM}_c^{(i)}(\mathbf{x})| - |\mathbf{x}^{(i)} - \mathrm{NH}^{(i)}(\mathbf{x})| \ , \tag{19}$$

where $\mathcal{Y} = \{1, \cdots, C\}$ is the label space, $\mathrm{NM}_c(\mathbf{x})$ is the nearest miss of $\mathbf{x}$ from class c, and $P(c)$ is the *a priori* probability of class c. By using the conclusions drawn in Section II, it can be shown that RELIEF-F is equivalent to defining a sample margin as:

$$\rho = \sum_{\{c \in \mathcal{Y}, c \neq y(\mathbf{x})\}} \frac{P(c)}{1 - P(c)} d(\mathbf{x} - \mathrm{NM}_c(\mathbf{x})) - d(\mathbf{x} - \mathrm{NH}(\mathbf{x})) \ . \tag{20}$$

Note that a positive sample margin does not necessarily imply a correct classification in 1-NN. With the identification of the margin of RELIEF-F, the extension of RELIEF-F to the iterative version, which we call I-RELIEF-1, can be easily done by plugging the margin of Eq. (20) into Eq. (2). We skip the detailed derivations here due to the space limitation.

From the commonly used margin definition for multiclass problems [25], however, it is more natural to define a margin as:

$$\begin{aligned} \rho &= \min_{\{c \in \mathcal{Y}, c \neq y(\mathbf{x})\}} d(\mathbf{x} - \mathrm{NM}_c(\mathbf{x})) - d(\mathbf{x} - \mathrm{NH}(\mathbf{x})) \ , \\ &= \min_{\{\mathbf{x}_i \in \mathcal{D} \setminus \mathcal{D}_{y(\mathbf{x})}\}} d(\mathbf{x} - \mathbf{x}_i) - d(\mathbf{x} - \mathrm{NH}(\mathbf{x})) \ , \end{aligned} \tag{21}$$

where $\mathcal{D}_c$ is a subset of $\mathcal{D}$ containing only the patterns from class c. Compared to the definition in Eq. (20), this definition regains the property that a positive sample margin corresponds to a correct classification. The derivation of the iterative version of multiclass RELIEF using the new margin definition, which we call I-RELIEF-2, is straightforward.

V. Online Learning

The iterative RELIEF algorithms are based on a batch learning model. That is, feature weights are updated after seeing all of the training data. In case the amount of training data is enormous, or we do not have the luxury of seeing all of the data when starting training, online learning is computationally much more attractive than batch learning. In this section, we derive an online algorithm for I-RELIEF. Convergence analysis is also presented.

A. Online Learning

Recall in I-RELIEF, the update of feature weights requires us to calculate the following term (Eq. (12)):

$$\boldsymbol{\nu} = \frac{1}{N} \sum_{n=1}^{N} \gamma_n (\bar{\mathbf{m}}_n - \bar{\mathbf{h}}_n) \tag{22}$$

Instead of treating each observation equally, we introduce a time-dependent forgetting factor $\xi^{(t)}$ ($0 \leq \xi^{(t)} \leq 1$) to emphasize the most recently observed data [26] by assuming that with more observations, we should have a more accurate estimation of feature weights:

$$\boldsymbol{\nu}^{(T)} = \vartheta^{(T)} \sum_{t=1}^{T} (\prod_{s=t}^{T} \xi^{(s)}) \gamma^{(t)} (\bar{\mathbf{m}}^{(t)} - \bar{\mathbf{h}}^{(t)}) \ , \tag{23}$$

where $\vartheta^{(T)} = (\sum_{t=1}^{T}(\prod_{s=t}^{T}\xi^{(s)}))^{-1}$ is a normalization factor. Eq. (23) can be further manipulated:

$$\begin{aligned}
\boldsymbol{\nu}^{(T)} &= \vartheta^{(T)}\sum_{t=1}^{T}(\prod_{s=t}^{T}\xi^{(s)})\underbrace{\gamma^{(t)}(\bar{\mathbf{m}}^{(t)}-\bar{\mathbf{h}}^{(t)})}_{\boldsymbol{\pi}^{(t)}}\,, \\
&= \vartheta^{(T)}\xi^{(T)}\left(\sum_{t=1}^{T-1}(\prod_{s=t}^{T-1}\xi^{(s)})\boldsymbol{\pi}^{(t)}+\boldsymbol{\pi}^{(T)}\right)\,, \\
&= \frac{\vartheta^{(T)}\xi^{(T)}}{\vartheta^{(T-1)}}\boldsymbol{\nu}^{(T-1)}+\vartheta^{(T)}\xi^{(T)}\boldsymbol{\pi}^{(T)}\,, \\
&= \boldsymbol{\nu}^{(T-1)}+\vartheta^{(T)}\left(\xi^{(T)}\boldsymbol{\pi}^{(T)}+(\tfrac{\xi^{(T)}}{\vartheta^{(T-1)}}-\tfrac{1}{\vartheta^{(T)}})\boldsymbol{\nu}^{(T-1)}\right)\,.
\end{aligned} \tag{24}$$

Unraveling $\vartheta^{(T)}$, we have:

$$\begin{aligned}
\vartheta^{(T)} &= (\sum_{t=1}^{T}(\prod_{s=t}^{T}\xi^{(s)}))^{-1}\,, \\
&= (\xi^{(T)}+\xi^{(T)}\sum_{t=1}^{T-1}(\prod_{s=t}^{T-1}\xi^{(s)}))^{-1}\,, \\
&= (\xi^{(T)}+\xi^{(T)}/\vartheta^{(T-1)})^{-1} = \tfrac{\vartheta^{(T-1)}}{\xi^{(T)}(\vartheta^{(T-1)}+1)}\,.
\end{aligned} \tag{25}$$

Substituting Eq. (25) into Eq. (24) yields:

$$\begin{aligned}
\boldsymbol{\nu}^{(T)} &= \boldsymbol{\nu}^{(T-1)}+\vartheta^{(T)}\xi^{(T)}\left(\boldsymbol{\pi}^{(T)}-\boldsymbol{\nu}^{(T-1)}\right)\,, \\
&= \boldsymbol{\nu}^{(T-1)}+\tfrac{\vartheta^{(T-1)}}{(\vartheta^{(T-1)}+1)}\left(\boldsymbol{\pi}^{(T)}-\boldsymbol{\nu}^{(T-1)}\right)\,.
\end{aligned} \tag{26}$$

By defining $\eta^{(T)} = \vartheta^{(T-1)}/(\vartheta^{(T-1)}+1)$ as a learning rate, Eq. (26) can be rewritten as:

$$\begin{aligned}
\boldsymbol{\nu}^{(T)} &= \boldsymbol{\nu}^{(T-1)}+\eta^{(T)}\left(\boldsymbol{\pi}^{(T)}-\boldsymbol{\nu}^{(T-1)}\right)\,, \\
&= (1-\eta^{(T)})\boldsymbol{\nu}^{(T-1)}+\eta^{(T)}\boldsymbol{\pi}^{(T)}\,.
\end{aligned} \tag{27}$$

It states that the current estimate can be simply computed as a linear combination of the previous estimate and the current observation.

After we obtain $\boldsymbol{\nu}^{(T)}$, the feature weight $\mathbf{w}^{(T)}$ can be calculated as:

$$\mathbf{w}^{(T)} = (\boldsymbol{\nu}^{(T)})^{+}/\|(\boldsymbol{\nu}^{(T)})^{+}\|_2\,, \tag{28}$$

where $(\nu_i)^{+} = \max(\nu_i, 0)$. The pseudo-code of online I-RELIEF is presented in Fig. 3.

B. Convergence Analysis

In this subsection, we establish the convergence of online I-RELIEF. We also prove that under certain conditions, online I-RELIEF does converge to I-RELIEF (based on batch learning) when $t \to +\infty$.

Theorem 3. *The online I-RELIEF algorithm converges when the learning rate is appropriately selected.*

Proof. The proof can be easily done by recognizing that Eq. (27) has the same form as the Robbins-Moron stochastic approximation algorithm [27]. The conditions on the learning rate $\eta^{(t)}$:

$$\lim_{t\to+\infty}\eta^{(t)} = 0\,, \sum_{t=1}^{\infty}\eta^{(t)} = \infty, \text{and } \sum_{t=1}^{\infty}(\eta^{(t)})^2 < +\infty\,, \tag{29}$$

ensure the convergence of online I-RELIEF. ■

Online Version of I-RELIEF

(1) **Initialization**: given $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$, set $\mathbf{w}^{(0)} = \mathbf{1}/I$ and $\boldsymbol{\nu}^{(0)} = \mathbf{0}$, number of iterations T, kernel width σ;

(2) **for** $t = 1 : T$

(3) Randomly select a pattern $\mathbf{x}$ from $\mathcal{D}$;

(4) Calculate pairwise distances with respect to $\mathbf{w}^{(t-1)}$;

(5) Calculate P_m, P_h and P_o as Eqs. (8), (9) and (10);

(6) Calculate $\boldsymbol{\pi}^{(t)}$ as Eq. (24);

(7) Calculate $\boldsymbol{\nu}^{(t)} = (1 - \eta^{(t)})\boldsymbol{\nu}^{(t-1)} + \eta^{(t)}\boldsymbol{\pi}^{(t)}$;

(8) Update weights $\mathbf{w}^{(t)} = (\boldsymbol{\nu}^{(t)})^+ / \|(\boldsymbol{\nu}^{(t)})^+\|_2$;

(9) **end**

(10) Output $\mathbf{w}^{(T)}$.

Fig. 3. Pseudo-code of online I-RELIEF

Eq. (27) presents a generic form for online RELIEF. However, in real applications, it is difficult to specify a learning rate for a given dataset. The choice of learning rates is a large remaining issue in stochastic approximation [27]. Therefore, in the following, we will only focus on a commonly used learning rate $\eta^{(t)} = 1/at$ with $a \in (0, 1]$. It is easy to show that this learning rate satisfies the conditions in Eq. (29). Also note that if $a = 1$, Eq. (27) can be simplified as: $\boldsymbol{\nu}^{(T)} = \boldsymbol{\nu}^{(T-1)} + \frac{1}{T}\left(\boldsymbol{\pi}^{(T)} - \boldsymbol{\nu}^{(T-1)}\right) = \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{\pi}^{(t)}$.

Theorem 3 ensures the convergence of online I-RELIEF but does not tell us whether online I-RELIEF converges to I-RELIEF. We prove that it is true in the following theorem. To eliminate the randomness, instead of randomly selecting patterns from $\mathcal{D}$, we group the data into blocks, each block containing the entire training dataset and denoted as $\mathcal{B}^{(m)} = \mathcal{D}$. Online I-RELIEF successively performs online learning over $\mathcal{B}^{(m)}, m = 1, 2, \cdots$. The order of samples within each block is fixed.

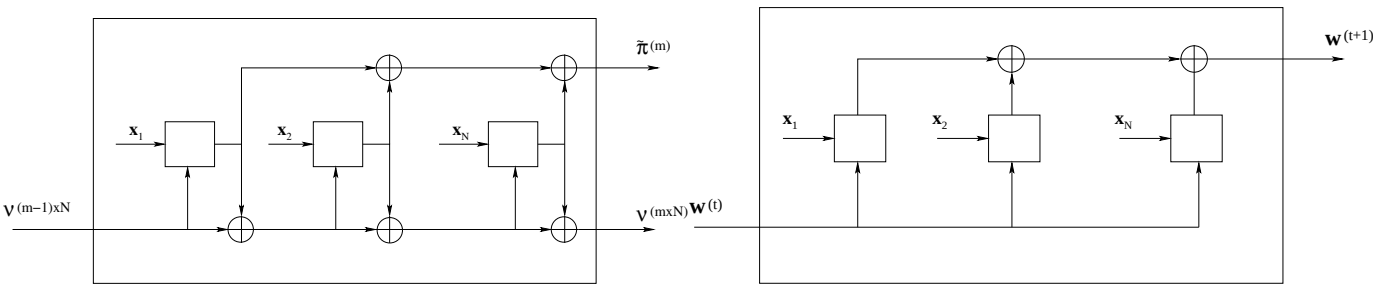
Theorem 4. *Let the learning rate $\eta^{(t)} = 1/at$ with $a \in (0, 1]$. If both algorithms converge, I-RELIEF and online I-RELIEF converge to the same solution, regardless of the learning rate (different a in $\eta^{(t)} = 1/at$) in online I-RELIEF.*

To prove the theorem, we need the following lemma.

Lemma 1. *Let $\{a_j\}$ be a bounded sequence, i.e. for every j, $M_1 \leq a_j \leq M_2$. If $\lim_{j\to+\infty} a_j = a^*$, then $\lim_{n\to+\infty} \frac{1}{n}\sum_{i=1}^{n} a_i = a^*$.*

Proof. The proof is provided in the Appendix.

Proof of Theorem 4. For simplicity, we first consider $\eta^{(t)} = 1/t$. For the m-th block of data, denote



(a) $\widetilde{T}$ (b) T

Fig. 4. Two constructed operators: (a) $\widetilde{T}$, and (b) T. Each square block denotes a search engine that, given a pattern and a feature weight vector, returns the nearest neighbors of the pattern. The only difference between $\widetilde{T}$ and T is that the former performs online learning while the latter does not.

$\tilde{\boldsymbol{\pi}}^{(m)} = \frac{1}{N}\sum_{t=(m-1)\times N+1}^{m\times N} \boldsymbol{\pi}^{(t)}$. After running over M blocks of data, we have:

$$\boldsymbol{\nu}^{(M\times N)} = \frac{1}{M\times N}\sum_{t=1}^{M\times N} \boldsymbol{\pi}^{(t)} = \frac{1}{M}\sum_{m=1}^{M} \tilde{\boldsymbol{\pi}}^{(m)} \ . \tag{30}$$

From Theorem 3, we know that online I-RELIEF converges, i.e. $\lim_{t\to+\infty} \boldsymbol{\nu}^{(t)} = \boldsymbol{\nu}^*$. It follows that $\lim_{m\to+\infty} \tilde{\boldsymbol{\pi}}^{(m)} = \tilde{\boldsymbol{\pi}}^*$. Since the data have compact support, the sequence $\{\tilde{\boldsymbol{\pi}}^{(m)}\}$ is bounded. Using Lemma 1, we have:

$$\lim_{M\to+\infty} \boldsymbol{\nu}^{(M\times N)} = \lim_{M\to+\infty} \frac{1}{M}\sum_{m=1}^{M} \tilde{\boldsymbol{\pi}}^{(m)} = \tilde{\boldsymbol{\pi}}^* = \boldsymbol{\nu}^* \ , \tag{31}$$

where the last equality is due to the fact that a convergent sequence cannot have two limits.

We prove the convergence of online I-RELIEF to I-RELIEF by using the property that for a contraction operator, the fixed point is unique. Recall that if the kernel width is appropriately selected, $T : \mathcal{W} \to \mathcal{W}$ is a contraction operator for I-RELIEF, i.e. $T(\mathbf{w}^*) = \mathbf{w}^*$. We then construct an operator $\widetilde{T} : \mathcal{W} \to \mathcal{W}$ for online I-RELIEF, which, in the m-th iteration, uses $\tilde{\mathbf{w}}^{(m-1)} = (\boldsymbol{\nu}^{((m-1)\times N)})^+ / \|(\boldsymbol{\nu}^{((m-1)\times N)})^+\|_2$ as input, and then computes $\boldsymbol{\nu}^{(m\times N)}$ by performing online learning on $\mathcal{B}^{(m)}$, and finally returns $\tilde{\mathbf{w}}^{(m)} = (\boldsymbol{\nu}^{(m\times N)})^+ / \|(\boldsymbol{\nu}^{(m\times N)})^+\|_2$. It follows from Eq. (31) that as $m \to +\infty$, we have $\widetilde{T}(\tilde{\mathbf{w}}^*) = \tilde{\mathbf{w}}^*$, where $\tilde{\mathbf{w}}^* = (\boldsymbol{\nu}^*)^+ / \|(\boldsymbol{\nu}^*)^+\|_2$. Therefore, $\tilde{\mathbf{w}}^*$ is the fixed point of $\widetilde{T}$. The only difference between $\widetilde{T}$ and T is that the former performs online learning whereas the latter does not. The two operators are plotted in Fig. 4. Since $\{\boldsymbol{\nu}^{(t)}\}$ is convergent, it is also a Cauchy sequence. That is, as $m \to +\infty$, the difference between every pair of $\boldsymbol{\nu}$ within one block goes to zero with respect to some norms. The operator $\widetilde{T}$, therefore, is identical to T in the limit. It follows that $\tilde{\mathbf{w}}^* = \mathbf{w}^*$, since otherwise there would be two fixed points for a contraction operator, which contradicts Theorem 1.

The above arguments can also be applied to the case where $a \in (0, 1)$. The only difference is that $\lim_{m\to+\infty} \tilde{\boldsymbol{\pi}}^{(m)} = (\tilde{\boldsymbol{\pi}}^*)/a$ where $\tilde{\boldsymbol{\pi}}^*$ is defined for $a = 1$. However, this will not change the final solution since the feature weights are calculated as $(\boldsymbol{\nu}^*)^+ / \|(\boldsymbol{\nu}^*)^+\|_2$. ■

VI. Computational Complexity

One major advantage of RELIEF and its variations over other algorithms is their computational efficiency. The complexity of RELIEF, I-RELIEF and online I-RELIEF are $\mathcal{O}(TNI)$, $\mathcal{O}(TN^2I)$ and $\mathcal{O}(TNI)$,

respectively, where T is the total number of iterations, I is the feature dimensionality and N is the number of data points. If RELIEF runs over the entire dataset, i.e. $T = N$, then the complexity is $\mathcal{O}(N^2 I)$. In the following section, we empirically show that online I-RELIEF can attain similar solutions to I-RELIEF after one pass of the training data. Therefore, the computational complexity of online I-RELIEF is of the same order as that of RELIEF.

VII. Experiments

A. Experimental Setup

We conduct large-scale experiments to demonstrate the effectiveness of the proposed algorithms and to study their behavior. Since in most practical applications one typically does not know the true feature set, it is necessary to conduct experiments in a controlled manner. We perform experiments on two test-beds. The first test-bed is composed of 9 datasets: *twonorm*, *waveform*, *ringnorm*, *diabetics*, *flare-solar*, *thyroid*, *heart*, *segmentation* and *yeast*, all publicly available at the UCI Machine Learning Repository [28]. The data information is summarized in Table I. We add 50 independently Gaussian distributed irrelevant features to each pattern, representing different levels of signal-to-noise ratios[2]. In real applications, it is also possible that some patterns are mislabeled. To evaluate the robustness of each algorithm against mislabeling, we introduce noise to training data but keep testing data intact. The level of noise represents a percentage of randomly selected training data for which its class labels are changed. The second test-bed contains six microarray datasets: *9-tumors* [29], *Brain-tumor2* [30], *Leukemia-1* [2], *prostate-tumors* [31], *DLBCL* [32] and *SRBCT* [33]. Except for *prostate-tumors* and *DLBCL*, the remaining four datasets are multiclass problems (from 3 to 9 classes). One characteristic of microarray data, different from most classification problems we encounter, is the extremely large feature dimensionality (from 2308 to 10509) compared to the small sample sizes (from 60 to 102). The data information is presented in Table I. For more detailed information on these data, interested readers can refer to [2], [29], [30], [31], [32], [33]. For all of the datasets, except for a simple scaling of each feature value to be between 0 and 1 as required in RELIEF, no other preprocessing is performed. The scaling operation was justified in [10] as to clamp each feature weight into the interval between -1 and 1. From our analysis, we note that scaling can also improve the performance of both RELIEF and I-RELIEF by noting that KNN usually uses this simple heuristic to improve its performance.

We use two metrics to evaluate the performance of the feature weighting algorithms. In most applications, feature weighting is performed for selecting a small feature subset to defy the curse of dimensionality. Therefore, a natural choice for a performance metric is classification errors. The classification-error metric, however, may not be able to fully characterize algorithmic performance. In some cases, we find experimentally that including a few irrelevant features may not change classification errors significantly. Indeed, improving classification performance sometimes is not the only purpose for performing feature weighting. In applications where the acquisition of data is quite expensive, including some useless features is highly undesirable. For microarray data, including irrelevant genes may complicate subsequent research. This consideration is the main motivation for us to add 50 useless features to original feature sets in the UCI datasets. We treat feature selection as a target recognition problem. Though features in original feature sets may be weakly relevant or even useless, it is reasonable to assume that the original features contain

[2]The signal-to-noise ratio refers to the ratio between the number of original features and that of the artificially added useless ones.

TABLE I

DATA SUMMARY OF 9 UCI AND 6 MICROARRAY DATASETS

Dataset	Train	Test	Feature	Class
twonorm	400	7000	20	2
waveform	400	4600	21	2
ringnorm	400	7000	20	2
diabetics	468	300	8	2
heart	170	100	13	2
flare-solar	666	400	9	2
thyroid	140	75	5	2
segmentation	210	2100	19	7
yeast	500	984	8	10
9-tumors	60	/	5726	9
Brain-tumor2	60	/	10367	4
Leukemia-1	72	/	5327	3
Prostate-tumors	83	/	2308	4
SRBCT	102	/	10509	2
DLBCL	77	/	5469	2

at least the same or more information than the useless ones that are added artificially. By changing a threshold, we can plot a receiver operating characteristic (ROC) curve [13] that gives us a direct view on the capabilities of each algorithm to identify useful features and at the same time rule out useless ones. However, as the classification-error metric, the ROC metric is not exclusive. Some algorithms are down-biased and tend to assign zero weights to not only useless features but also some presumably useful features in original feature sets (c.f. Fig. 8), resulting in a small area under a ROC curve. Since we do not know the true identities of features in original feature sets, in this case, we need to check classification errors to see if the studied algorithm does select all of the useful features.

B. Experiments on UCI Datasets

We first perform experiments on the UCI datasets. For binary problems, we compare I-RELIEF with RELIEF-F and Simba. For multiclass problems, we compare RELIEF-F with I-RELIEF-1 and I-RELIEF-2.

To make the experiment computationally feasible, we use KNN to estimate classification errors for each feature weighting algorithm. KNN is certainly not an optimal classifier for each dataset. However, the focus of the paper is not on the optimal classification but on feature weighting. KNN provides us with a platform where we can compare different algorithms fairly with a reasonable computational cost. The number of the nearest neighbors K is estimated through a stratified 10-fold cross validation using training data. We do not spend extra effort on re-estimating K when only a subset of features are used in training and testing, rather opting to use the one estimated in the original feature space. Though the value of K is surely not optimal, we find that it is fair for each algorithm.

The kernel width σ is the only free parameter in I-RELIEF. We show in Section VII-C that σ is not a critical parameter. Nevertheless, we estimate it through 10-fold cross validation in the experiment. One problem associated with the estimation with cross validation using classification errors as criterion is that it requires us to specify the optimal number of features used in KNN. To overcome this difficulty, the following heuristic method is used: for a given candidate of σ, feature weights are estimated, and then

KNN is performed in the induced weighted feature space [34]. The optimal σ is then chosen as the one with the smallest classification error. Likewise, we find the number of NH and NM in RELIEF-F through cross validation, rather than presetting it to 10 as suggested in [12]. The code of Simba used in the study is downloaded from [8]. As we have seen in Section II, there are some local maxima in Simba's objective function. Simba tries to overcome this problem by performing a gradient ascent from serval different starting points. We set the number of stating points to be 5, which is the default value of Simba. Also, we set the number of passes of the training data to be 5, the default value of which is 1. In summary, we try to make the comparison among different algorithms as fairly as possible.

To eliminate statistical variations, each algorithm is run 20 times for each dataset. In each run, a dataset is randomly partitioned into training and testing, and 50 irrelevant features are added. The averaged testing errors of KNN as a function of the number of the top ranked features are plotted in Fig. 5. (In the notation 50/10, the first number refers to the number of irrelevant features and the second one the percentage of mislabeled samples.) The ROC curves of the algorithms are plotted in Fig. 6. As a reference, the classification errors of KNN on the clean data (without irrelevant features and mislabeling) and noisy data are reported in Table II. From these experimental results, we arrive at the following observations.

(1) The performance of KNN is degraded significantly in the presence of a large amount of irrelevant features, as reported in the literature, while mislabeling has less influence on the performance of KNN than irrelevant features.

(2) From Fig. 5, we can see that with respect to classification errors, in nearly all of datasets, I-RELIEF performs the best, RELIEF-F the second and Simba the worst. For a more rigorous comparison between I-RELIEF and RELIEF-F, a Student's paired two-tailed t-test is also performed. The p-value of the t-test reported in each row in Table II represents the probability that two sets of compared samples come from distributions with equal means. The smaller the p-value, the more significant the difference of the two average values is. The optimal number of features used in KNN is estimated through 10-fold cross validation using training data. At the 0.03 p-value level, I-RELIEF wins on nine cases (*ringnorm* ($50/10$), *twonorm* ($50/10$), *thyroid* ($50/0$), *waveform*, *f-solar* and *yeast*), and ties with RELIEF-F on the remaining nine cases. As we argued before, the classification-error metric may not fully characterize algorithmic performance. Therefore, we check the ROC curves plotted in Fig. 6. In almost all datasets, I-RELIEF has the largest area under a ROC curve, RELIEF-F the second, and Simba the smallest. For five (*ringnorm* (50/0), *heart* (50/10), *thyroid* (50/10) and *diabetics*) out of nine cases that have no significant differences in classification errors, it is clear from Fig. 6 that I-RELIEF performs much better than RELIEF-F with respect to the ROC metric.

To further demonstrate the performance of each algorithm, we particularly focus on two datasets: *ringnorm* and *waveform*. We plot the learned feature weights of one realization in Figs. 7 and 8. For ease of comparison, the maximum value of each feature weight vector is normalized to be 1. Without mislabeling, the weights learned in RELIEF-F are similar to those of I-RELIEF but the former have larger weights on the useless features than the latter. It is interesting to note that, for both datasets, Simba assigns zero weights to not only useless features but also to some presumably useful ones. In these cases, we need to go back to the classification-error metric. Particularly, for *waveform* (50/0), we observe that the testing error of Simba becomes flat after the tenth features since, except for these 10 features, the weights of the remaining features are all zero. This implies that Simba in effect does not identify all of the useful

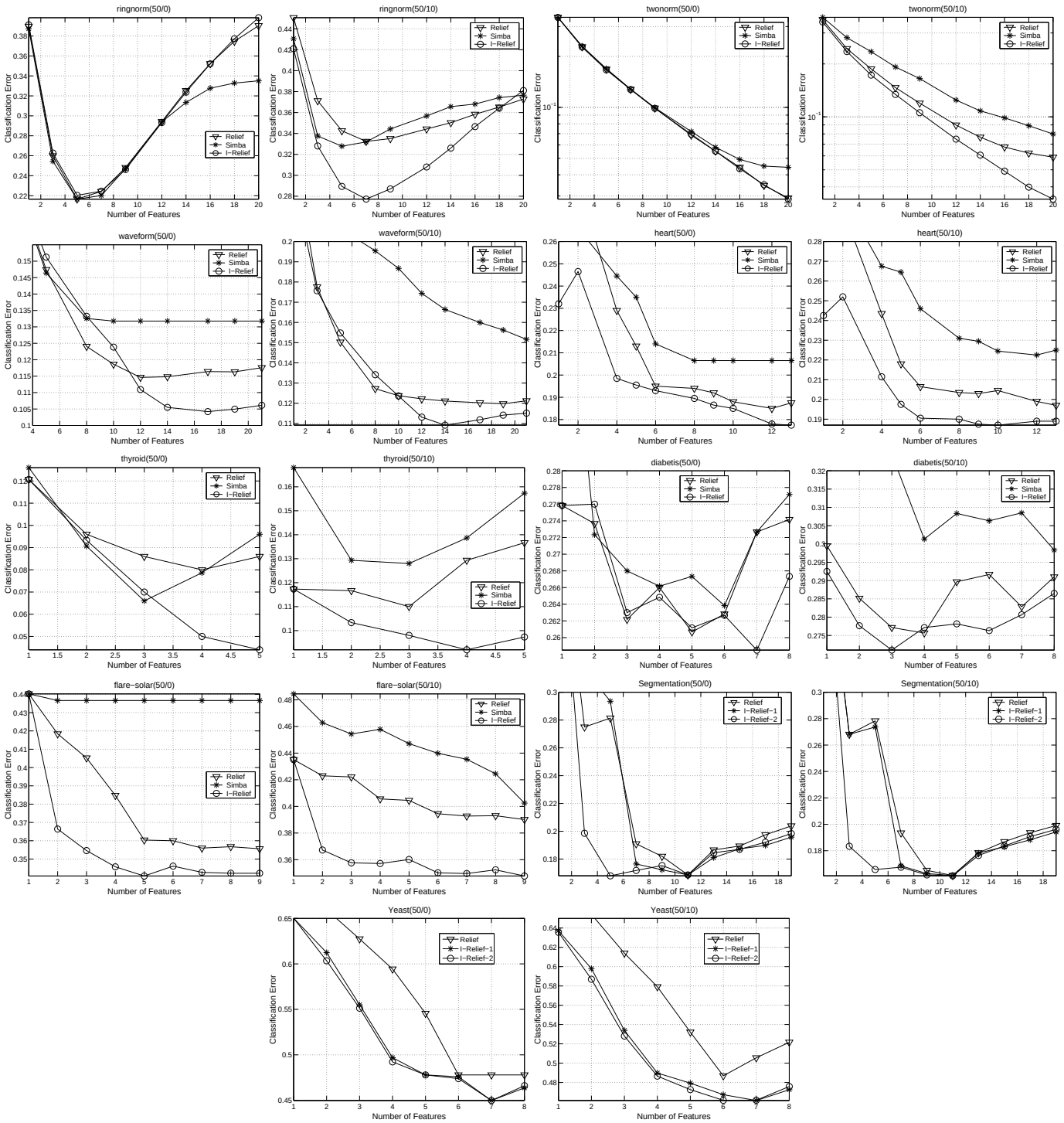


Fig. 5. Comparison of three algorithms using the classification error metric on 9 UCI datasets.

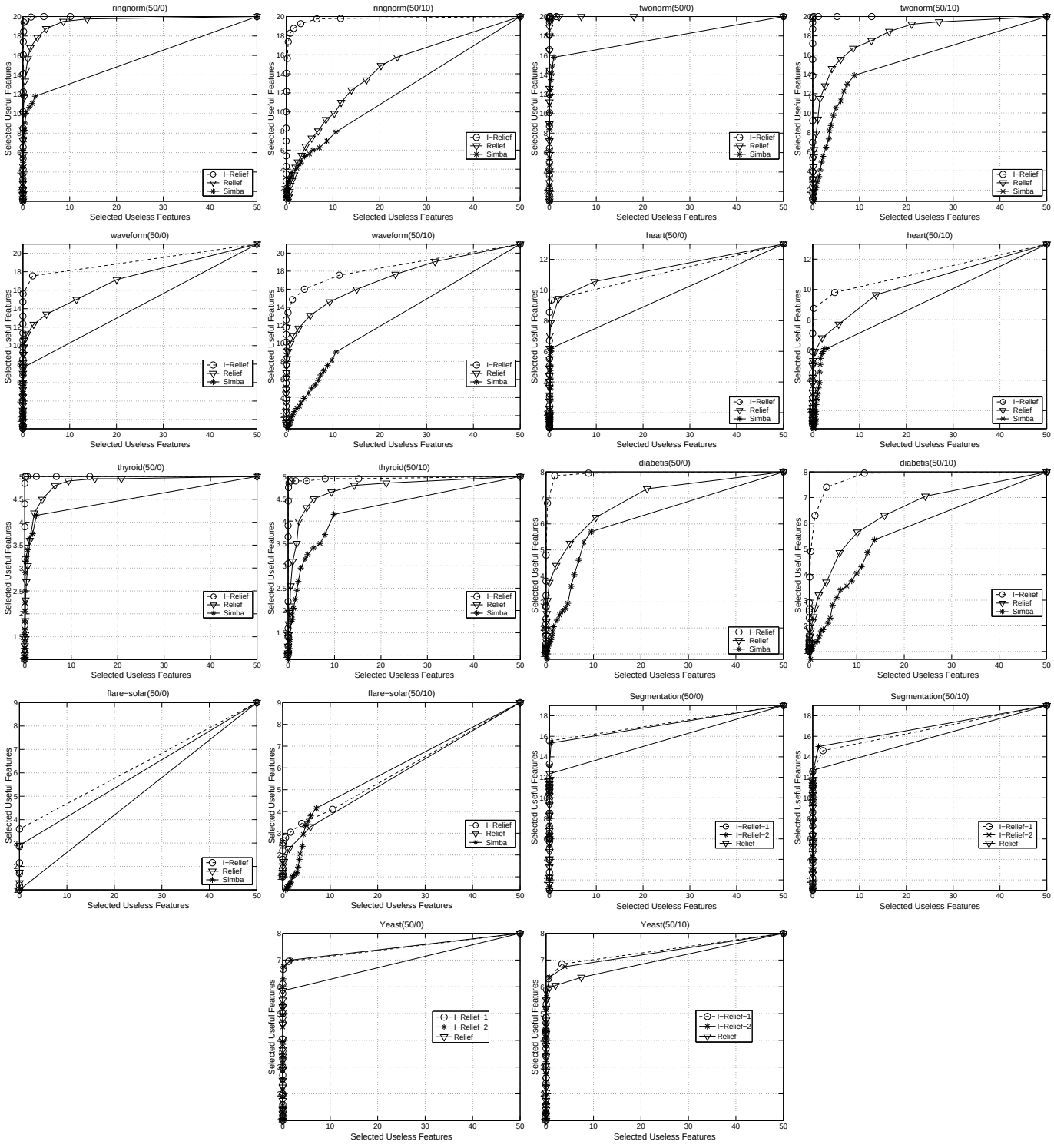


Fig. 6. Comparison of three algorithms using the ROC metric on 9 UCI datasets.

TABLE II

THE TESTING ERRORS AND STANDARD DEVIATIONS (%) ON 9 UCI DATASETS. THE LAST COLUMN SUMMARIZES THE P-VALUES OF STUDENT'S T-TESTS COMPARING THE RESULTS OF RELIEF AND I-RELIEF FOR EACH DATASET. THE LAST ROW (W/L/T) SUMMARIZES WIN/LOSS/TIE IN COMPARING RELIEF AND I-RELIEF BASED ON A SIGNIFICANCE LEVEL 0.03.

Dataset	KNN (clean data)	Mislabelling	KNN (noisy data)	I-RELIEF	RELIEF	P-value (I-RELIEF/RELIEF)
Ringnorm	39.2(1.3)	0%	45.1(1.2)	22.0(1.2)	21.7(1.1)	0.47
		10%	44.2(1.1)	**28.1(1.5)**	**34.0(4.5)**	**0.00**
Twonorm	3.1(0.2)	0%	4.8(0.6)	3.1(0.7)	3.2(0.5)	0.96
		10%	6.4(0.7)	**3.7(0.7)**	**6.2(1.3)**	**0.00**
Waveform	12.6(0.7)	0%	14.2(1.7)	**10.5(1.1)**	**11.2(1.1)**	**0.03**
		10%	14.7(1.6)	**11.2(1.2)**	**12.2(1.3)**	**0.00**
Heart	17.9(3.1)	0%	19.2(2.9)	18.8(3.4)	19.6(3.2)	0.45
		10%	19.8(3.9)	19.5(3.9)	20.9(3.6)	0.26
Thyroid	4.4(2.4)	0%	24.1(3.8)	**5.8(3.2)**	**8.7(4.3)**	**0.02**
		10%	26.0(4.1)	9.8(3.8)	11.3(3.6)	0.20
Diabetics	28.9(1.5)	0%	31.6(2.6)	27.1(1.7)	27.0(2.1)	0.80
		10%	31.7(2.5)	27.2(2.2)	27.3(1.8)	0.77
F-solar	34.8(2.4)	0%	34.5(2.6)	**34.5(3.3)**	**37.1(3.8)**	**0.03**
		10%	36.1(1.7)	**35.1(2.1)**	**38.7(3.7)**	**0.00**
Segment	12.5(1.4)	0%	27.9(1.7)	17.0(1.4)	17.7(1.7)	0.17
		10%	29.2(1.8)	17.3(1.4)	17.4(1.2)	0.92
Yeast	43.8(0.9)	0%	65.8(1.5)	**45.0(1.8)**	**47.8(1.4)**	**0.00**
		10%	66.2(1.4)	**45.5(1.7)**	**49.0(1.9)**	**0.00**
						W/T/L=9/9/0

features. With 10% mislabeling, the weight quality of both RELIEF-F and Simba degrades significantly, whereas I-RELIEF performs similarly as before. For example, for *waveform* (50/10), Simba mistakenly identifies an irrelevant feature as the top feature. These observations imply that both Simba and RELIEF-F are not robust against label noise.

We finally focus on the dataset *diabetics*, for which the statistical test shows that there is no significant difference in classification errors between RELIEF-F and I-RELIEF (p-value ≈ 0.8). However, from Figs. 6 and 9, I-RELIEF is the clear winner in terms of solution quality. One possible explanation is that the dataset *diabetics* contains some weakly relevant features, and these features contribute insignificantly to classification performance though having more information than random noise. This example demonstrates the effectiveness of I-RELIEF in ranking features according to their relevance. Similar results are also observed on the datasets *ringnorm* (50/0), *heart* (50/10) and *thyroid* (50/10). From this experiment, we conclude that when comparing feature selection and weighting algorithms, using classification errors as the only performance metric may not be enough.

C. Choice of Kernel Width

The kernel width σ in I-RELIEF can be estimated through cross validation on training data. It is well-known that the cross-validation method may result in an estimate with a large variance. Fortunately, this problem does not pose a serious concern. In this subsection, we show that σ is not a critical parameter. In Fig. 10, we plot the feature weights learned on *twonorm* and *waveform* using different σ values. We observe that for relatively large σ values, the resulting feature weights do not have much difference. This indicates that the performance of I-RELIEF is not sensitive to the choice of σ values, which makes model selection easy in practical applications.

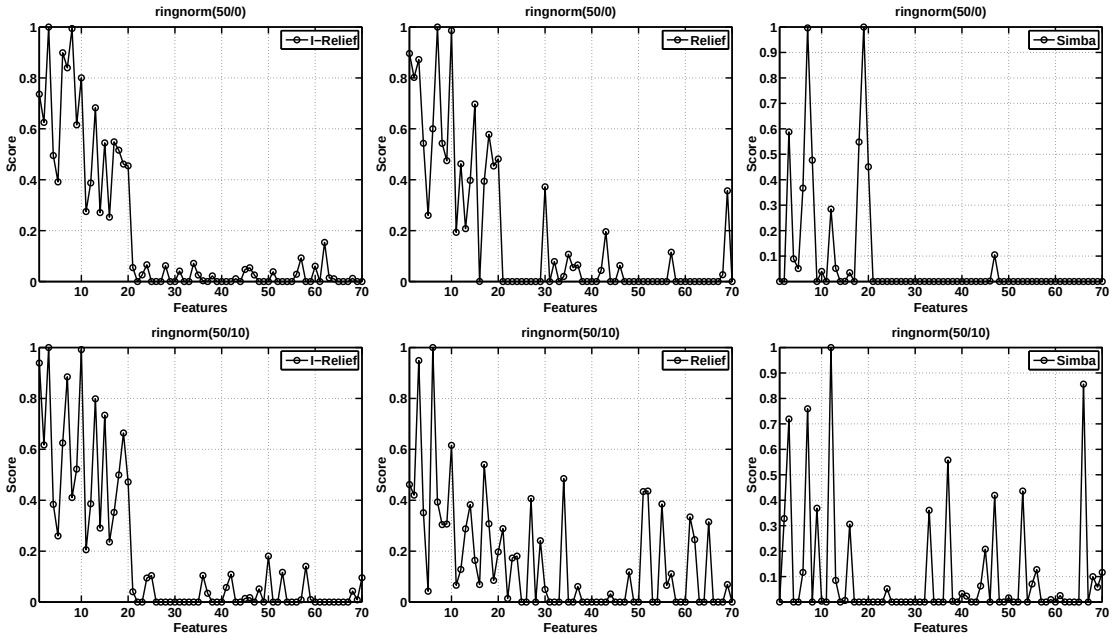


Fig. 7. Feature weights learned in three algorithms on *ringnorm* dataset. The first 20 features are presumably useful.

We also conduct some experiments to confirm the convergence results established in Section III-B. Plotted in Fig. 11(a) are the convergence rates of I-RELIEF with different σ values on *waveform* dataset. We observe that the algorithm diverges when $\sigma = 0.05$ but converges in all other cases. Moreover, with the increase of σ values, the convergence becomes faster. In Fig. 11(b), we plot the convergence rates of I-RELIEF with different initial values for a fixed kernel width. The line with stars is for the uniformly distributed initial value, and the line with circles for randomly generated initial values, both averaged from 10 runs. This experimental result confirms that I-RELIEF converges from *any* starting point, and using the uniform initial value does improve convergence but the improvement is negligible.

D. Online Learning

In this subsection, we perform some experiments to verify the convergence properties of online I-RELIEF established in Section V. The feature weights learned in I-RELIEF are used as a target vector. The stopping criterion θ (line 1 in Fig. 2) is set to be 10^{-5} to ensure that the target vector is a good approximation of the true solution (c.f. Eq.(17)). The convergence results with different learning rates (different a in $\eta^{(t)} = 1/at$), averaged from 20 runs, are plotted in Fig. 12. We only present the results of *waveform* and *ringnorm* since the results for other datasets are almost identical. From the figure, we first observe that online I-RELIEF, regardless of the learning rates, converges to I-RELIEF, which confirms the theoretical findings in Theorem 4. We also find that after 400 iterations (both datasets have 400 training samples), the feature weights are already very close to the target vectors. In Fig. 13, we plot the target vector and the feature weights of online I-RELIEF (after 400 iterations). For comparison, the feature weights of RELIEF-F for *ringnorm* are also plotted.

The choice of learning rates is an open issue in stochastic approximation. From Fig. 12, we find that appropriate selection of a learning rate does accelerate convergence. However, the impact on the final solution is quite small. From this experiment, we may conclude that in practical applications we can

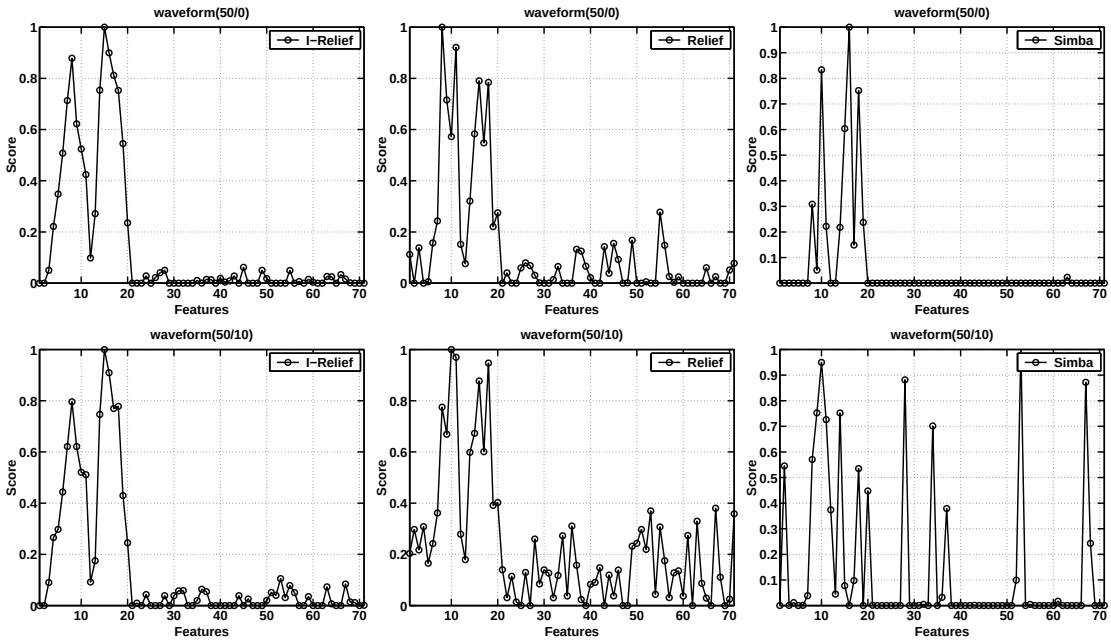


Fig. 8. Feature weights learned in three algorithms on *waveform* dataset. The first 21 features are presumably useful.

simply set the learning rate $\eta^{(t)} = 1/t$ without sacrificing much in performance.

E. Experiments on Microarray

We finally compare RELIEF to I-RELIEF with two margin definitions on six microarray datasets. Due to the limited sample numbers, the leave-one-out method is used to evaluate the performance of each algorithm.

The classification errors of KNN as a function of the 500 top ranked features are plotted in Fig. 14. Since *Prostate-Tumor* and *DLBCL* are binary problems, I-RELIEF-1 is equivalent to I-RELIEF-2. From the figure, we observe that except for *DLBCL*, in which I-RELIEF performs similar to RELIEF, for the remaining five datasets, I-RELIEF-2 is the clear winner compared to RELIEF and I-RELIEF-1. For *9-Tumors* and *Brain-Tumor2*, I-RELIEF-2 outperforms RELIEF over all ranges by $5\% - 20\%$. For *Leukemia-1* and *SRBCT*, though the performance of three algorithms all converge after 100 genes, it is clear that I-RELIEF is much more accurate than RELIEF in ranking genes. For example, in *SRBCT*, with 5 genes, the error for I-RELIEF-2 is about 7% compared to about 17% for RELIEF. Also note that in *SRBCT*, I-RELIEF-2 almost reaches a zero error with about 15 genes, compared to about 60 genes in RELIEF (The x-axis is log-scaled.). For comparison, we also report the classification errors of KNN using all genes. We can see that gene selection can significantly improve the performance of KNN .

We note that the numbers of genes found by I-RELIEF that correspond to the minimum classification errors are all less than 200. With these small gene sets, oncologists may be able to work on them directly to infer molecular mechanisms underlying disease causes. Also, if for classification purposes, some computationally expensive methods such as wrapper methods, can be used to further filter out some redundant genes. By using some sophisticated classification algorithms such as SVM, much improvement on classification performance is expected. One important issue in using feature weighting algorithms for microarray data analysis is to determine where to cut in a ranked gene list. It may be difficult to derive an

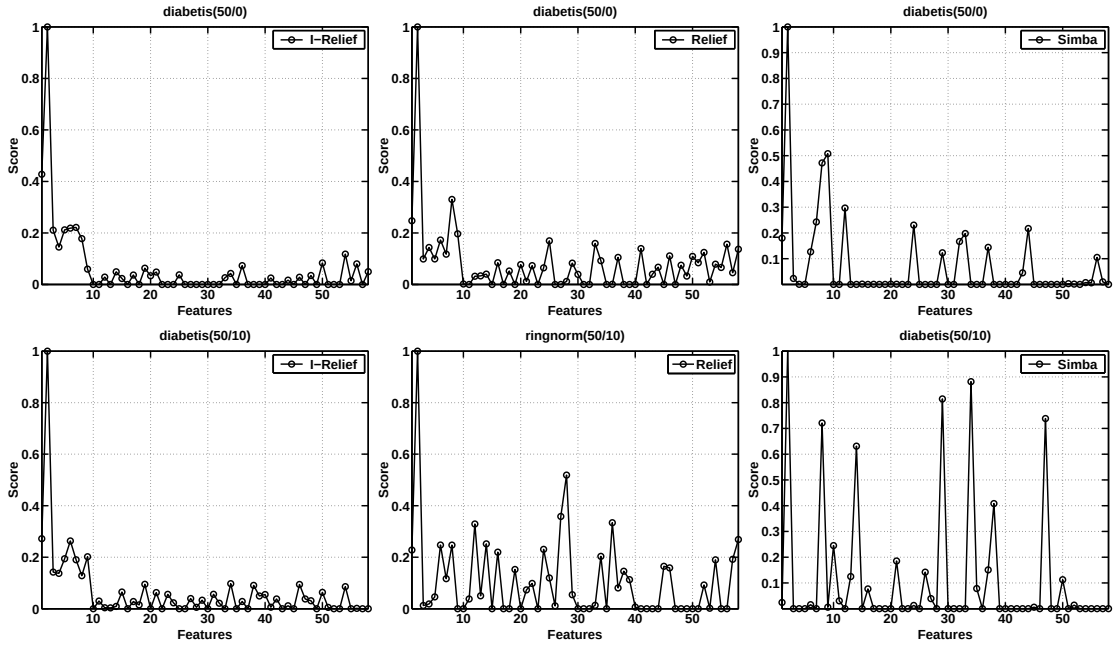


Fig. 9. Feature weights learned in three algorithms on *diabetics* dataset. The first 8 features are presumably useful.

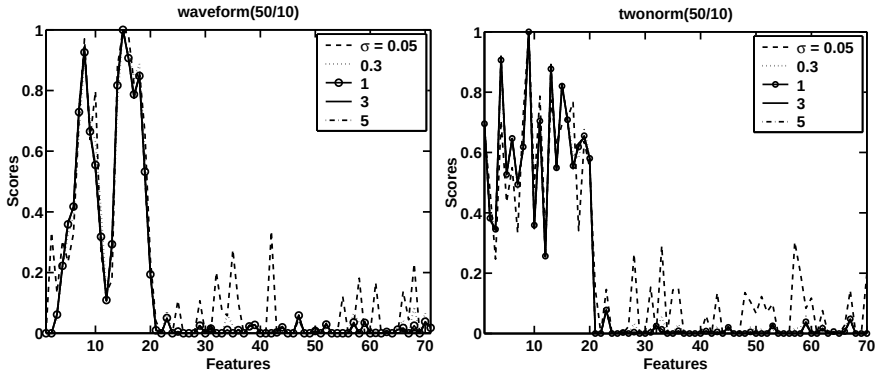
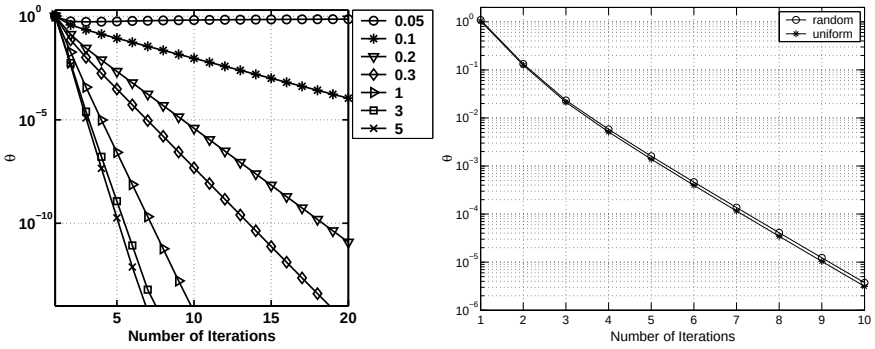


Fig. 10. Feature weights learned using different σ values on *twonorm* and *waveform* datasets.



(a) (b)

Fig. 11. The convergence rates of I-RELIEF using (a) different σ, and (b) different initial values on *waveform* dataset. The y-axis $\theta = \|\mathbf{w}^{(t+1)} - \mathbf{w}^{(t)}\|$.

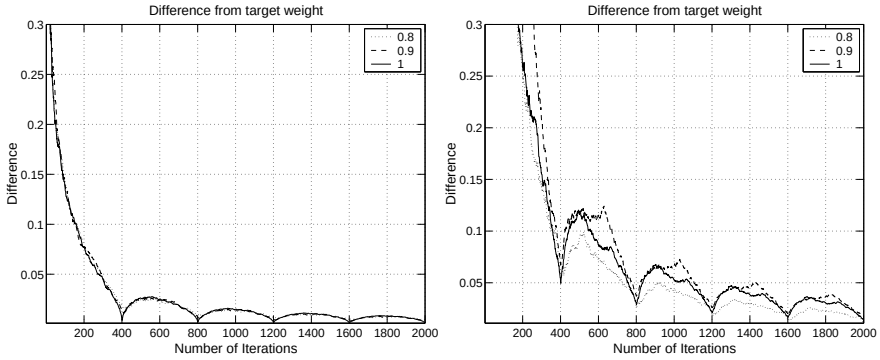


Fig. 12. Convergence rates of online I-RELIEF using difference learning rates on (a) *waveform*, and (b) *ringnorm* datasets.

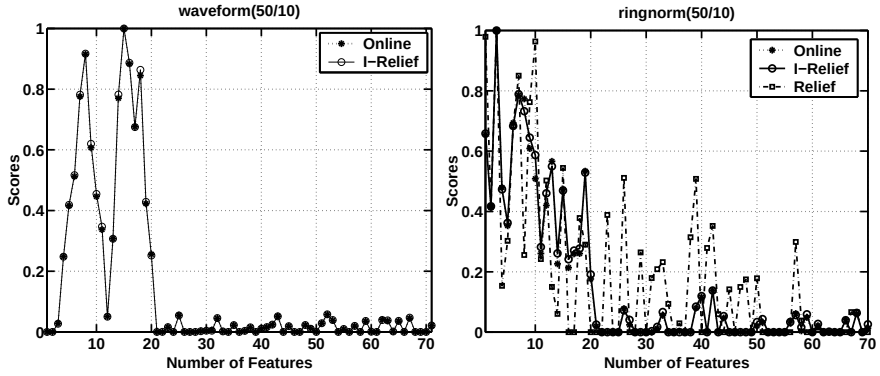


Fig. 13. Feature weights learned in I-RELIEF and online I-RELIEF (after 400 iterations) on *waveform* and *ringnorm* datasets.

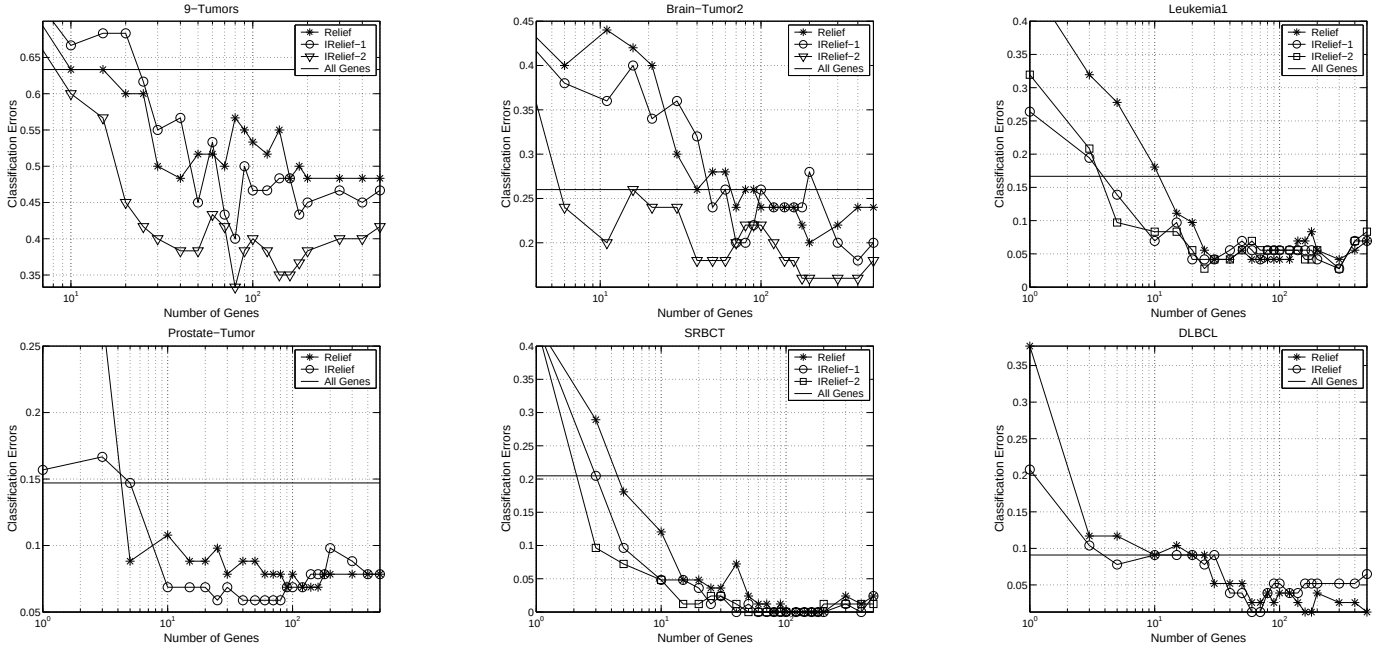


Fig. 14. Classification errors of three feature weighting algorithms on six microarray datasets.

analytical solution. One commonly used method is through cross validation that first estimates classification parameters, including cut-off thresholds, based on training data, and then uses the estimated parameters to classify the held-out testing samples (also called the two-loop protocol). Detailed descriptions of the experimental protocols is beyond the scope of the paper. Interested readers may refer to [35].

VIII. Conclusion

In this paper, we have proposed a set of new feature weighting algorithms, all stemming from a simple yet informative interpretation of RELIEF. We have proven that RELIEF implements an online algorithm that solves a convex optimization problem with a margin based objective function. This new interpretation enables us to identify some weaknesses of RELIEF and to propose some solutions to fix them. Following the principle of the EM algorithm that treats unobserved data as random variables, we have proposed an iterative RELIEF (I-RELIEF) algorithm. Using a new multiclass margin definition, we have proposed a new multiclass RELIEF algorithm. In order to speed up the learning process of I-RELIEF, we have also developed an online I-RELIEF algorithm. We have established some convergence theorems for the proposed I-RELIEF algorithms.

Large-scale experiments have been conducted on nine UCI and six microarray datasets to demonstrate the effectiveness of the proposed algorithms and to verify the theoretical results established in this paper. We have experimentally shown that: (1) I-RELIEF performs significantly better than the RELIEF-F and Simba algorithms in terms of the capability to recover useful features and rule out useless ones, and is robust against mislabeling noise; (2) I-RELIEF has a nice convergence property. If the parameter is properly selected, I-RELIEF converges to a unique solution regardless of the initial starting points. Moreover, the performance of I-RELIEF is not sensitive to the choice of the tuning parameter, which makes model selection easy in practical applications; (3) with online learning, I-RELIEF can be implemented with the same computational cost as RELIEF.

We outline several directions we will pursue in the future:

1) Although we have shown that the kernel width is not a critical parameter in I-RELIEF, it would be highly desirable to estimate the kernel width during the learning process, making I-RELIEF a parameter-free algorithm.
2) The correct identification of relevant genes lays solid foundation for subsequent research (e.g. breast cancer prognosis). The effectiveness of I-RELIEF in ranking features according to their relevance has been clearly demonstrated in the UCI datasets but only partially in the microarray datasets by using the classification-error metric. We are currently working closely with oncologists to determine the biological significance of the top ranked genes identified by our algorithms. Also, encouraged by the superior performance of I-RELIEF over RELIEF-F, we intend to develop a classification system for microarray data by combining I-RELIEF with some sophisticated classifiers such as SVM.

IX. Acknowledgement

We gratefully thank four anonymous reviewers for their valuable comments to improve the paper quality.

X. Appendix – Proof of Lemma1

Lemma 1. *Let $\{a_j\}$ be a bounded sequence, i.e. for every j, $M_1 \leq a_j \leq M_2$. If $\lim_{j\to+\infty} a_j = a^*$, then $\lim_{n\to+\infty} \frac{1}{n}\sum_{i=1}^{n} a_i = a^*$.*

We need to prove that for every $\epsilon > 0$, there exists a positive integer $N(\epsilon)$ such that $|\frac{1}{n}\sum_{i=1}^{n} a_i - a^*| < \epsilon$ for every $n > N(\epsilon)$. We first consider $\frac{1}{n}\sum_{i=1}^{n} a_i - a^* < \epsilon$. Since $\{a_j\}$ is a convergent sequence, for every ϵ, we can find an $\epsilon_1 \in (0, \epsilon)$ and a positive integer $N(\epsilon_1)$ such that $|a_j - a^*| < \epsilon_1$ for every $j > N(\epsilon_1)$.

Since $M_1 \leq a_j \leq M_2$ for every j, we have

$$\begin{aligned} & \frac{1}{n}\sum_{i=1}^{n} a_i - a^* \ , \\ = \ & \frac{1}{n}\left(\sum_{i=1}^{N(\epsilon_1)} (a_i - a^*) + \sum_{i=N(\epsilon_1)+1}^{n} (a_i - a^*)\right) \ , \\ < \ & \frac{1}{n}\left(\sum_{i=1}^{N(\epsilon_1)} (M_2 - a^*) + \sum_{i=N(\epsilon_1)+1}^{n} \epsilon_1\right) \ , \\ = \ & \tfrac{1}{n} N(\epsilon_1)(M_2 - a^* - \epsilon_1) + \epsilon_1 \ . \end{aligned} \tag{32}$$

It immediately follows that if $n > \max\{N(\epsilon_1)(M_2 - a^* - \epsilon_1)/(\epsilon - \epsilon_1), N(\epsilon_1)\}$, $\frac{1}{n}\sum_{i=1}^{n} a_i - a^* < \epsilon$ holds. Similarly, we can prove that if $n > \max\{N(\epsilon_1)(a^* - M_1 - \epsilon_1)/(\epsilon - \epsilon_1), N(\epsilon_1)\}$, $\frac{1}{n}\sum_{i=1}^{n} a_i - a^* > -\epsilon$ holds. Therefore, for every ϵ, there exists $N(\epsilon) = \max\{N(\epsilon_1)(M_2 - a^* - \epsilon_1)/(\epsilon - \epsilon_1), N(\epsilon_1)(a^* - M_1 - \epsilon_1)/(\epsilon - \epsilon_1), N(\epsilon_1)\}$ such that $|\frac{1}{n}\sum_{i=1}^{n} a_i - a^*| < \epsilon$ for every $n > N(\epsilon)$, which proves the lemma. ■

References

[1] Y. Sun and J. Li, "Iterative RELIEF for feature weighting," in *Proc. 23rd International Conference on Machine Learning*. ACM Press, 2006, pp. 913–920.

[2] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, October 1999.

[3] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. Mesirov, T. Poggio, W. Gerald, M. Loda, E. Lander, and T. Golub, "Multiclass cancer diagnosis using tumor gene expression signatures," *Proc. Natl. Acad. Sci. USA*, vol. 98, no. 26, pp. 15 149–15 154, December 2001.

[4] A. Jain and D. Zongker, "Feature selection: evaluation, application, and small sample performance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, February 1997.

[5] P. Pudil and J. Novovicova, "Novel methods for subset selection with respect to problem knowledge," *IEEE Intelligent Systems*, vol. 13, no. 2, pp. 66 – 74, March 1998.

[6] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik., "Feature selection for SVMs," in *Advances in Neural Information Processing Systems*, 2001, pp. 668–674.

[7] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157 – 1182, 2003.

[8] R. Gilad-Bachrach, A. Navot, and N. Tishby, "Margin based feature selection - theory and algorithms," in *Proc. 21st International Conference on Machine Learning*. ACM Press, 2004, pp. 43–50.

[9] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[10] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proc. 9th International Conference on Machine Learning*. Morgan Kaufmann, 1992, pp. 249 – 256.

[11] T. G. Dietterich, "Machine learning research: Four current directions," *AI Magazine*, vol. 18, no. 4, pp. 97–136, 1997.

[12] I. Kononenko, "Estimating attributes: Analysis and extensions of RELIEF," in *Proc. European Conference on Machine Learning*, 1994, pp. 171–182.

[13] R. Duda, P. Hart, and D. Stork, *Pattern Classification*. New York: J. Wiley, 2000.

[14] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B*, vol. 39, no. 1, pp. 1–38, 1977.

[15] T. Jenssen and E. Hovig, "Gene-expression profiling in breast cancer," *Lancet*, vol. 365, pp. 634–635, 2005.

[16] K. Crammer, R. Gilad-Bachrach, A. Navot, and N. Tishby, "Margin analysis of the LVQ algorithm," in *Advances in Neural Information Processing Systems*, 2002, pp. 462–469.

[17] E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*. New York: John Wiley and Sons Inc., 2001.

[18] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[19] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[20] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor, "Linear programming boosting via column generation," *Machine Learning*, vol. 46, pp. 225–254, 2002.

[21] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning," *Artificial Intelligence Review*, vol. 11, no. 15, pp. 11–73, 1997.

[22] R. Kress, *Numerical Analysis*. New York: Springer-Verlag, 1998.

[23] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press, 1995.

[24] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *Journal of Artificial Intelligence Research*, vol. 2, pp. 263–286, 1995.

[25] Y. Sun, S. Todorovic, J. Li, and D. Wu, "Unifying error-correcting and output-code AdaBoost through the margin concept," in *Proc. 22nd International Conference on Machine Learning*. ACM Press, 2005, pp. 872 – 879.

[26] M. Sato and S. Ishii, "On-line EM algorithm for the normalized Gaussian network," *Neural Computation*, vol. 12, no. 2, pp. 407–432, Feb. 2000.

[27] H. Kushner and G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*, 2nd ed. New York: Springer-Verlag, 2003.

[28] C. Blake and C. Merz, "UCI repository of machine learning databases," 1998.

[29] J. Staunton, D. Slonim, H. Coller, P. Tamayo, M. Angelo, J. Park, U. Scherf, J. Lee, W. Reinhold, J. Weinstein, J. Mesirov, E. Lander, and T. Golub, "Chemosensitivity prediction by transcriptional profiling," *Proc. Natl. Acad. Sci. USA*, vol. 98, no. 19, pp. 10 787–10 792, September 2001.

[30] C. Nutt, D. Mani, R. Betensky, P. Tamayo, J. Cairncross, C. Ladd, U. Pohl, C. Hartmann, M. McLaughlin, T. Batchelor, P. Black, A. von Deimling, S. Pomeroy, T. Golub, and D. N. Louis, "Gene expression-based classification of malignant gliomas correlates better with survival than histological classification," *Cancer Research*, vol. 63, no. 7, pp. 1602–1607, April 2003.

[31] D. Singh, P. Febbo, K. Ross, D. Jackson, J. Manola, C. Ladd, P. Tamayo, A. Renshaw, A. D'Amico, J. Richie, E. Lander, M. Loda, P. Kantoff, T. Golub, and W. Sellers, "Gene expression correlates of clinical prostate cancer behavior," *Cancer Cell*, vol. 1, no. 2, pp. 203–209, March 2002.

[32] M. Shipp, K. Ross, P. Tamayo, A. Weng, J. Kutok, R. Aguiar, M. Gaasenbeek, M. Angelo, M. Reich, G. Pinkus, T. Ray, M. Koval, K. Last, A. Norton, T. Lister, J. Mesirov, D. Neuberg, E. Lander, J. Aster, and T. Golub, "Diffuse large b-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning," *Nature Medicine*, vol. 8, no. 1, pp. 68–74, 2002.

[33] J. Khan, J. Wei, M. Ringner, L. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. Antonescu, C. Peterson, and P. Meltzer, "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks," *Nature Medicine*, vol. 7, no. 6, pp. 673–679, 2001.

[34] D. Wettschereck, D. W. Aha, and T. Mohri, "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 273–314, 1997.

[35] L. Wessels, M. Reinders, A. Hart, C. Veenman, H. Dai, Y. He, and L. van't Veer, "A protocol for building and evaluating predictors of disease state based on microarray data," *Bioinformatics*, vol. 21, pp. 3755–3762, 2005.