Abstract The nearest neighbor rule is one of the most widely used models for classification and selecting a compact set of prototype instances is an important problem for its applications. Many existing approaches on the prototype selection problem rely on instance-based analyses and local criteria on the class distribution, which are intractable for numerical optimization techniques. In this paper, we explore a parametric approach, which learns the parameters for prioritizing individual prototypes so as to minimize an approximated violation of the nearest neighbor rule over the training set. We show that our approach reduces the problem to large-margin learning and demonstrate its advantage by empirical comparisons using public benchmark data.

Keywords Nearest neighbor classifier, Prototype selection, Soft maximum, Large-margin learning

1 Introduction

The classification model based on the nearest neighbor rule and a set of prototype instances is one of the most widely used models for classification and a typical example of non-parametric, lazy learning algorithms [33]. Prototype selection is a pre-processing to reduce the number of prototypes and an critical problem for its applications due to the practical cost of storing and measuring distances for a large number of prototypes [6].

The goal of prototype selection is to obtain a compact set of prototypes that can maintain or possibly improve the accuracy of the nearest neighbor rule. Many existing techniques for prototype selection have been motivated by local or instance-based criteria, e.g., removing prototypes far from the decision boundary of two classes to eliminate redundancy, or conversely, removing those near the boundary for generalization. Improvements by hybrid methods of such techniques have been reported [30]. Due to their instance-based definitions, however, it is generally difficult to learn optimal values of the parameters for these techniques. Alternatively, critical parameters are usually selected empirically by validation with wrapper methods [26].

In this paper, we revisit the prototype selection problem to explore a parametric approach that is based on the behavior of the nearest neighbor model over the training set and a discriminative learning principle. There are two key intuitions. First, we define a parametric extension of the nearest neighbor rule, in which the choice of the nearest neighbor is adjusted by the numerical parameter assigned to each prototype. The parameter values can also

S. Ando
School of Management,
Tokyo University of Science,
500 Shimokiyoku, Kuki,
Saitama, 3468512 Japan E-mail: ando@rs.tus.ac.jp
indicate instances that are irrelevant for the predictions of the training set. Secondly, we introduce a differentiable approximation of the condition that the nearest neighbor rule predicts the class of a training instance correctly, and subsequently derive a constrained optimization problem for minimizing the violation of the condition with regards to the numerical parameters of the prototypes. We further show that this problem reduces to a large-margin learning using a sparse representation of the relations among the neighboring instances.

The proposed method is also applicable in domains where only pairwise distances among the instances are available, while many techniques for prototype selection rely on the analyses of the feature vector space. These analyses are infeasible for non-vector input data and variables with complex dependency among them. The nearest neighbor algorithms are often used for such data, e.g., in time series classification, the Euclidean distance is often problematic due to the sequential structure of the time series [18], and a dissimilarity function based on non-linear warping is known to be highly effective [10, 34].

In summary, the contribution of our work is a prototype selection approach which (a) enables an optimization algorithm which directly addresses the violation of the nearest neighbor rule and (b) expands the application to the distance space input data. The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 describes the proposed method. Section 5 shows the results of our empirical study. Section 6 presents our conclusion.

2 Related Work

2.1 Prototype Selection

Prototype selection [15], also referred to as data reduction, [32], instance selection [24, 16, 17], and template reduction [11, 27] is a pre-processing for the nearest neighbor classification. Its goal is to select a compact subset of the training set as prototypes, in order to improve the efficiency of the algorithm while maintaining or possibly reducing the generalization error.

A typical operation in prototype selection methods, referred to as editing, eliminates or relocates noisy prototypes that cause misclassifications near the borders of different classes [32, 6], and have the effects of smoothing and generalizing the decision boundaries. Another typical operation is condensation, which discards prototypes far from the class borders in order to reduce the redundancy of the prototype set without affecting the decision boundary [2]. Combinations of these techniques, such as feature analysis [28] and clustering [25, 30], are used to analyze boundaries and interior instances in hybrid methods [23, 29].

The implementations of these techniques are generally categorized as wrapper and filtering methods [26]. In the wrapper methods, different sets of prototypes are evaluated and selected based on the performances of the nearest neighbor classifier. The filtering methods uses score functions that are less expensive to compute than the performance measures and select prototypes based on their individual scores. The wrapper methods is a general, flexible framework for empirically selecting a best model among a finite number of settings, but do not provide a natural framework for learning an optimal model. For the filter methods, strong or complex dependency among prototypes are problematic, as it is difficult to represent such dependency sufficiently by individual scores.

Heuristic combinatorial search such as evolutionary algorithms, a memetic algorithm, an artificial immune model, and tabu search, have been explored to find compact subsets of the training instances [7, 14, 13, 35]. Theoretical approaches for prototype selection shown the upper-bound of generalization error based on VC dimensions, structural risk minimization, and Bayesian analysis [9, 19, 12]. The looseness of the bounds and the scalability of the algorithms are practical issues in these approaches.
2.2 Discriminative Learning and Nearest Neighbor Model

Recently, discriminative approaches within which the nearest neighbor model are embedded have been studied in the context of metric learning. Large-margin Nearest Neighbor Classification \cite{31} addresses the problem of distance metric learning as a constrained optimization problem to minimize the violation of the nearest neighbor rule with regards to the affine transformation. The violation is formalized as the margin from the delineating hyperplane and the problem is solved by semi-definite programming. Max-margin Multiclass Nearest Neighbors \cite{21} also learns the metric space based on the minimization of its entropy.

3 Proposed Method

3.1 Notations

Let $X = \{x_i\}_{i=1}^n$ denote the set of labeled instances and $Y = \{y_i\}_{i=1}^n$ their class values. Instances are not necessarily represented as vectors, as the nearest neighbor prediction requires only (dis)similarities from the prototypes to a new instance.

In order to evaluate a candidate prototype set in the training phase, one must avoid trivial cases where the target instance being classified is also one of the prototypes set, as the prototypes and the training instances originally come from the same set of instances, $X$. In the following sections, we employ a setup similar to the leave-one-out cross validation, such that the target instance is always removed from the prototype set. That is, the prototypes for predicting the class of $x_i$ will be a subset of the remainder of the labeled instances, denoted by $Z_i = X \setminus \{x_i\}$. In this paper, we will make the distinction to refer to an instance as a prototype only when it is used as one of the reference objects for the nearest neighbor rule, and in turn, refer to one as a training instance only when it is the subject of prediction.

3.2 Soft-Maximum Function

The soft maximum \cite{5} is an approximation of the function $\max(\cdot, \cdot)$ as $\log(\exp(\cdot) + \exp(\cdot))$. The approximation is convex and differentiable, which are desirable for numerical optimization.

A natural extension of the soft maximum for $m$ variables $x_1, \ldots, x_m$ is given by

$$M(x_1, \ldots, x_m) = \log \sum_{i=1}^m \exp(x_i)$$

(1)

The base of the exponential, $\beta$, is larger than 1 and modified to adjust for the scale of the input values.

3.3 Rank-Adjusted Nearest Neighbor Rule

This section introduces an extension of the nearest neighbor rule in order to parametrize the classification model and the prototype selection problem.

Let $x$ denote a target instance and $Z$ denote the set of prototypes. We denote by $R(x, x_j; Z)$ the rank of $d(x, x_j)$ in the set $\{d(x, x_j)\}_{x_j \in Z}$. The rank takes an integer value from $\{1, \ldots, \#Z\}$ and a smaller integer is a higher rank. The nearest neighbor rule for a target instance $x$ is as follows.

1. Compute distances $\{d(x, x_j)\}_{x_j \in Z}$
2. Assign rank $R(x, x_j; Z)$ to all $x_j \in Z$ based on their distances to $x$
3. Return the class value of the highest-ranked $x_j$
We now introduce a positive-value parameter $\alpha(x_j)$ for each prototype to define the adjusted rank $R'$.

$$R'(x, x_j; \mathcal{Z}) = R(x, x_j; \mathcal{Z}) + \alpha(x_j)$$ (2)

The nearest neighbor rule based on the adjusted rank $R'$ generalizes the original rule and produces identical predictions when $\alpha(x_1) = \ldots = \alpha(x_n)$.

Based on the adjusted ranks, the nearest prototype can be passed over for another prototype. We thus refer to $\alpha(x_j)$ as the degradation parameter of the prototype $x_j$. For brevity, we will use $\alpha_j$ as $\alpha(x_j)$ when it is clear from context. One may correct individual misclassifications by assigning larger degradations to the related prototypes.

That is, if $x_j$ and $x_k$ are respectively the highest and the second highest-ranked prototypes for a training case $(x, y)$ where $y_j \neq y$ and $y_k = y$, the misclassification of the basic rule may be corrected by assigning a relatively larger degradation to $\alpha_j$ than $\alpha_k$.

Our goal in this extension is to learn the degradation parameters that improve the average classification performance over all hold-out cases. Based on their values, we can identify which prototypes are more relevant or irrelevant in the prediction of the training set and reduce the prototype set accordingly. The motivation for focusing on ranks rather than distance values is to avoid the issue of scaling. The latter may vary substantially in scale depending on the function, and are more difficult to adjust.

3.4 Approximation of Nearest Neighbor Rule

Let $(x_i, y_i)$ denote a pair of a hold-out instance and its class label. We denote by $\mathcal{P}$ the subset of prototypes $\mathcal{Z}_i$ with the same label as $y_i$ and $\mathcal{Q}$ that of labels other than $y_i$. The condition that the nearest neighbor rule correctly predicts $y_i$ is that its nearest neighbor is an element of $\mathcal{P}$. That is,

$$\max \left\{ -d(x_i, p) \right\}_{p \in \mathcal{P}} > \max \left\{ -d(x_i, q) \right\}_{q \in \mathcal{Q}}$$ (3)

Note that the trivial prediction does not occur because $x_i$ is not an element of $\mathcal{P}$. In an ideal prototype set, the condition (3) is satisfied over all training cases for all $x_i \in \mathcal{X}$. In general, however, there may not exist such an ideal set. A practical principle for selecting the prototypes is, therefore, to reduce the cases where the conditions are violated as much as possible.

The formulation of such a principal is non-trivial due to the $\max$ function in (3). Alternatively, we rewrite (3) substituting the soft maximum $\mathcal{M}$ and the ranks of $d(x_i, q)$ and $d(x_i, p)$ for $\max$ and the distance values, respectively.

$$\mathcal{M} \left( \left\{ -R(x_i, p; \mathcal{Z}_i) \right\}_{p \in \mathcal{P}} \right) \geq \mathcal{M} \left( \left\{ -R(x_i, q; \mathcal{Z}_i) \right\}_{q \in \mathcal{Q}} \right) + 1, \ \forall x_i \in \mathcal{X}$$ (4)

In essence, (4) describes the same condition as (3) with regards to the ranks of the prototypes of the same and different classes as the hold-out case. The constant 1 is added to the RHS given that $R$ is the rank and takes an integer value.

Substituting (1) to (4), we have

$$\log \left( \sum_{p \in \mathcal{P}} \exp \left( -R(x_i, p; \mathcal{Z}_i) \right) \right) \geq \log \left( \sum_{q \in \mathcal{Q}} \exp \left( -R(x_i, q; \mathcal{Z}_i) \right) \right) + 1$$

Taking the exponential on both sides, we obtain

$$\sum_{p \in \mathcal{P}} \exp \left( -R(x_i, p; \mathcal{Z}_i) \right) - \sum_{q \in \mathcal{Q}} \exp \left( -R(x_i, q; \mathcal{Z}_i) \right) \geq \rho(x_i)$$ (5)
where $\rho(x_i)$ is

$$\rho(x_i) = (\beta - 1) \sum_{q \in Q} \exp(-R(x_i, q; Z_i))$$  \hspace{1cm} (6)

Since $\mathcal{P} \cup \mathcal{Q} = Z_i$, the LHS of (5) is rewritten as

$$\sum_{x_j \in Z_i} \delta(y_i, y_j) \exp(-R(x_i, x_j; Z_i)) \geq \rho(x_i)$$  \hspace{1cm} (7)

where

$$\delta(y_i, y_j) = \begin{cases} 1 & \text{if } y_i = y_j \\ -1 & \text{otherwise} \end{cases}$$

Considering (7) as a constraint, we introduce a slack variable $\xi_i$ for each $x_i$, to represent the violation of the nearest neighbor rule.

$$\sum_{x_j \in Z_i} \delta(y_i, y_j) \exp(-R(x_i, x_j; Z_i)) \geq \rho(x_i) - \xi_i$$  \hspace{1cm} (8)

Each $\xi_i$ takes a non-negative value, and the violation of the nearest neighbor rule over the training set is given as $\sum_{x_i \in \mathcal{X}} \xi_i$.

### 3.5 Problem Formulation

Combining the rank degradation parameter and the soft-max approximation, we formulate a constrained optimization problem for the parametrized nearest neighbor classification model.

First, we rewrite (8) using the adjusted rank.

$$\sum_{x_j \in Z_i} \delta(y_i, y_j) \exp(-R(x_i, x_j; Z_i) - \alpha(x_j)) \geq \rho'(x_i) - \xi_i$$  \hspace{1cm} (9)

where $\rho'(x_i)$ is the residual term that corresponds to $\rho(x_i)$ in (3), i.e.,

$$\rho'(x_i) = (\beta - 1) \sum_{q \in Q} \exp(-R'(x_i, q; Z_i))$$

$$= (\beta - 1) \sum_{q \in Q} \exp(-R(x_i, q; Z_i) - \alpha(q))$$  \hspace{1cm} (10)

(9) is not a desirable form for a constrained optimization problem because the parameter being trained appears on both sides. That is, $\alpha(x_j)$ on the LHS and $\alpha(q)$ in $\rho'(x_i)$. Alternatively, we replace $\rho'(x_i)$ with $\rho(x_i)$, which is constant for each $x_i$, and the solution that satisfies (8) will also satisfy (9) since $\rho'(x_i) \leq \rho(x_i)$. For brevity, we denote $\rho(x_i)$ as $\rho_i$ when it is clear in the context.

Secondly, we consider the regularization for the degradation parameters. The nearest neighbor rule with adjusted ranks will show the same behavior when all parameters are increased or decreased by the same amount, which is problematic for convergence. The issue can be addressed by the regularization on $\{\alpha(x_j)\}$, to promote less adjustments of ranks given the same amount of violations.

Let $\lambda$ denote the regularization function on $\{\alpha(x_j)\}$ and $C$ the trade-off coefficient. The constrained optimization problem is formalized as

$$\arg\min_{\alpha_j \geq 0, \xi_i \geq 0} \lambda(\alpha_1, \ldots, \alpha_n) + C \sum_{i=1}^{n} \xi_i$$  \hspace{1cm} (11)
subject to
\[ \sum_{x_j \in Z_i} \delta(y_i, y_j) \exp \left( -R \left( x_i, x_j; Z_i \right) - \alpha_j \right) \geq \rho_i - \xi_i, \quad \forall x_i \in \mathcal{X} \]

To gain further insight on the above problem, we rewrite the LHS of the constraints as a linear combination of \( w = (w_1, \ldots, w_n) \) and \( r_i = (r_{i1}, \ldots, r_{in}) \), where \( w_j = \exp(\alpha_j) \) and \( r_{ij} = \delta(y_i, y_j) \exp(-R(x_i, x_j; Z_i)) \). Using the \( \ell^2 \)-norm of \( w \), whose elements monotonically increase with \( \{\alpha_j\} \), as the regularizer, we obtain
\[
\begin{align*}
\arg\min_{w, \xi_i \geq 0, \rho_i \geq 0} & \quad \|w\|^2 + C \sum_i \xi_i \\
\text{subject to} & \quad w^\top r_i \geq \rho_i - \xi_i, \quad \forall x_i \in \mathcal{X}
\end{align*}
\]  
(12)

In (12), the problem has reduced to a quadratic programming for learning the vector \( w \) of a linear classifier by max-margin principle. In turn, \( r_i \) interprets as a mapping of the hold-out case to a feature vector space. Each feature in \( r_i \) corresponds to a prototype, and characterizes the training instance \( x_i \). Between the exponential definition of \( r_i \) and the regularization on \( w \), the problem induces a sparse representation which highlights the relevant prototypes.

Given the form of (13), the problem is solved by a constrained gradient descent \cite{20}. From the learned degradation values, each prototype is scored by its relevance, i.e., the highest adjusted rank, in the prediction of the hold-out cases,
\[
s(x_j) = \alpha_j + \min \left\{ \{R(x_i, x_j; Z_i)\}_{x_i \in Z_j} \right\}
\]  
(14)

The prototypes with low scores are removed from the prototype set as they have less effect on the predictions. If the size of the prototype set is fixed, the prototypes are selected by their scores from the highest to the lowest. When the size of the prototypes is not given, one can choose the size of set by cross validation. In the rest of the paper, we refer to the proposed method as Prototype Selection based on Rank-Exponential (REPS) and \( r_i \) as the Exponential Rank representation of \( x_i \).

4 Algorithm

Algorithm 1 illustrates the pseudocode of the selection procedure. With regards to the computational requirements, the space complexity is dominated by that of storing the mutual distance matrix among labeled instances, which is \( O(n^2) \). The time complexity is dominated by the computation of ranks, which is \( O(n \log n) \) for each instance.

**Algorithm 1 Pseudocode of REPS algorithm**

**INPUT:** Training set \( \mathcal{X} = \{x_i\}_{i=1}^n \), regularization parameter \( C \), prototype selection size \( k \)

**OUTPUT:** Prototype set \( Z \subset \mathcal{X} \)

**METHOD:**

Compute the distance matrix \( [d(x_i, x_j)]_{n \times n} \) and the ranks \( \{R(x_i, x; Z_i)\}_{x \in Z} \)

Compute \( r_{ij} = \delta(y_i, y_j) \exp(-R(x_i, x_j; Z_i)) \) and \( \rho(x_i) \) \cite{6}

Solve (12) for \( w \)

for \( j = 1 \) to \( n \) do

Compute \( \alpha_j = \log w_j \) and \( s(x_j) \) from (14)

end for

return \( \arg\max_{Z \subset \mathcal{X}, \# Z = k} \min \{s(x_j)\}_{x_j \in Z} \)
5 Empirical Results

We evaluate the effectiveness of REPS by empirical comparison using public benchmark datasets.

5.1 Datasets

For the first part of the experiment, we employed a collection of 41 dataset with vector features from the UCI Machine Learning Repository [3] and the KEEL datasets [1]. The summary of the datasets is shown in Table 3 in Appendix A. Although some of the benchmarks are not large datasets, a substantial reduction of the prototype set, even when the training set is small, is not irrelevant in practical aspect, as the execution time for the nearest neighbor prediction is a considerable obstacle in its applications.

In the second part of the experiment, we use 27 datasets from the UCR Time Series Dataset Repository [8]. The input data for time series classification is given as a dissimilarity matrix of Dynamic Time Warping (DTW) [4], which is known to be highly effective in time series classification [10]. We tested window sizes between five and fifty for DTW, and the results were generally consistent among them. The window size is five for all results shown here.

5.2 Baseline methods

We selected four recent prototype selection methods as baselines for comparative analysis. The prototype selection based on clustering (PSC) exploit cluster analysis for assessing class distribution [25]. Support Vector based prototype selection (SVBPS) uses support vectors of the SVM as prototypes [22]. Class conditional instance selection (CCIS) uses pair-wise relations among prototypes to define the selection criteria [24]. Fast condensed nearest neighbor (FCNN) exploits the condensation operation for prototype selection [2]. Template reduction for kNN (TRkNN) is based on the class boundary analysis [11]. These algorithms were executed in the KEEL software [1]. We also report the performance of the nearest neighbor classifier using the full training set as prototypes (NoPS).

The summary of parameter specifications for the baseline methods are included in the supplementary material (Table 6). Each method is evaluated using values in the second column and the best result is reported. With regards to the parameters of REPS, the regularization coefficient \( C \) was set to 0.001. Its results were robust with regards to the values of \( C \).

5.3 Evaluation

The proposed method and the baselines were evaluated in two aspects: the accuracy of the nearest neighbor classifier and the compactness of the prototype set, measured by the error rate (ERR) and the selection rate (SLR), respectively. ERR is the ratio of misclassifications with respect to the total number of predictions. SLR is defined as the ratio of the prototype set size with respect to the training set.

Since there is a natural trade-off between these two measure, it is necessary to compare them simultaneously. However, since multiobjective comparison can be unintuitive, we also use single-valued measures. The harmonic mean of the error and the selection rate is defined as

\[
\mu_{HM} = \frac{2 \times ERR \times SLR}{ERR + SLR}
\]

The empirical results are provided at [http://www.rs.tus.ac.jp/ando/DAMI16EXP.zip](http://www.rs.tus.ac.jp/ando/DAMI16EXP.zip)
When the error and the selection rates are close to 0 or 1, the differences in harmonic mean values are very small. In such cases, we use the log odds ratio (LOR) of the two rates for visualization. LOR against NoPS is defined as

\[
\text{LOR} = \log \frac{O(\text{SLR})O(\text{ERR})}{O(\text{ERR-NoPS})}
\]

\(O(p) = \frac{p}{1-p}\) is the odds given the probability \(p\).

The fixed selection rate (FSR) errors are evaluated for the proposed method with its selection rate fixed to the same value as each baseline. For all measures, a smaller value indicates a better performance. The performances are averaged over 5-fold cross validation for the vector datasets. The default training/test split is used for the time series datasets.

5.4 Results

5.4.1 Vector Data

Fig. 1 summarizes the comparison of the error and the selection rates. The \(x\)- and \(y\)-axes indicate the values of respective rates. Each point represent the performances of the methods for one dataset, and different methods are distinguished by the color and the shape of the points. The shift toward the \(x\)-axis indicate the advantage in terms of compactness and the shift towards the \(y\)-axis indicate the advantage in accuracy. Table A in Appendix A presents the comparisons in more detail. Each row shows ERR, SLR, and \(\mu_{HM}\) of each method for one dataset. For NoPS, the selection rate is always 1. The best values among related columns are indicated in bold.
In Fig. 1, the connecting lines indicate the results from the same problem, respectively, which are noted by asterisks in Table A. The general trend from top-left to bottom right is due to the trade-off between the two measures. Note that there is no intrinsic trade-off between the error and selection rates among the results of the same method as they are of different problems. The balance of the two measures vary depending on the method. REPS, along with CCIS, shows the tendency toward higher selection rates.

We tested our observations by signed rank tests of alternative hypotheses that the means of REPS are smaller than those of each baseline method. The summary of the tests are shown in Table 1. The 2nd and the 4th rows show the means of ERR and SLR, respectively. The 3rd and the 5th rows show the p-values of the tests comparing REPS with each baseline. REPS has substantial advantages in either ERR or SLR against all baselines and the null hypotheses are rejected with high confidence for those measures. The 6th and the 7th rows show the mean and the p-values of the signed ranked tests regarding the harmonic mean values, respectively. The null hypotheses are rejected with high confidence levels.

Table 2 summarizes the FSR errors. The second row shows the error rates of the baselines and the third row shows the FSR errors of REPS. The error rates on each column are directly comparable as the selection rates are identical. We conducted the signed rank tests with alternative hypotheses that the means of the fixed selection rate error are smaller than the error rate of the corresponding baseline. The p-value of each test, shown on the bottom row, indicates that the null hypotheses are rejected with high confidence levels and the significant advantage of REPS against baselines with same selection rates.

5.4.2 Time Series Data

Fig. 2 illustrates the error and the selection rates for the time series datasets. We only show the comparisons with REPS and NoPS, as other baseline methods do not take input in the dissimilarity space. The x- and y-axes indicate the error and the selection rates, respectively, and the results from the same datasets are connected by a gray line. More details of the results are shown in the supplementary material (Table 4). Fig. 2 shows that the error rates without prototype selection are generally smaller in this collection, which makes the reduction of prototypes without compromise in the accuracy is much more difficult than in the previous experiment.
Table 1 Comparison of Means and Signed Rank Test Results

|       | REPS  | CCIS  | FCNN  | PSC    | SVBPS | TRKNN | NoPS |
|-------|-------|-------|-------|--------|-------|-------|------|
| µERR | 0.32  | 0.36  | 0.32  | 0.37   | 0.38  | 0.34  | 0.25 |
| p-value | -     | 0.0051 | 0.25  | 0.00081 | 0.00015 | 0.043 | -    |
| µSLR | 0.094 | 0.14  | 0.35  | 0.28   | 0.27  | 0.46  | 1.0  |
| p-value | -     | 0.11  | 3.1×10⁻⁸ | 4.9×10⁻⁷ | 2.5×10⁻⁶ | 1.7×10⁻⁹ | -    |
| µHM  | 0.10  | 0.13  | 0.32  | 0.30   | 0.29  | 0.36  | 0.44 |
| p-value | -     | 0.026 | 1.0×10⁻⁸ | 9.8×10⁻⁸ | 3.8×10⁻⁷ | 1.4×10⁻⁹ | -    |

Table 2 Comparison of FSR Error Rates

|       | CCIS  | FCNN  | PSC    | SVBPS | TRKNN |
|-------|-------|-------|--------|-------|-------|
| µERR  | 0.36  | 0.32  | 0.37   | 0.38  | 0.34  |
| µFSRERR | 0.31  | 0.26  | 0.26   | 0.26  | 0.25  |
| p-value | 5.6×10⁻⁴ | 1.9×10⁻⁴ | 5.4×10⁻⁵ | 3.8×10⁻⁵ | 1.4×10⁻⁴ |

Fig. 3 Sensitivity Analysis for β

Additionally, we conducted a sensitivity analysis of the logarithm base β of the soft maximum function. We compared the performances of REPS over the same collection of datasets while changing β between 1 and 4. The log odds ratio are used for visualization as the harmonic means were close to 0. Fig. 3 shows the result of the sensitivity analysis in a box-whisker chart. The y-axis indicates the value of the log odds ratio. It shows that the largest median is achieved with β = 2, and the performance of REPS is robust with regards to the value of β.

6 Conclusion

We presented an extension of the nearest neighbor rule based on the adjustment of ranks to parametrize the prototype selection problem and also to approximate the violation of the rule over the training set. As a result, the problem is defined as discriminative learning in a sparse feature space. It is different from a relaxed combinatorial optimization problem.
The key intuition for this approach is to avoid intermediate evaluation and to learn the priorities of the prototypes in a direct relation with the performance over the training set. Its intuitive advantage is keeping track of the dependence among the prototypes that are stronger and more complex in a reduced set, which is difficult using local criteria and instance-based analyses.

In effect, the proposed approach exploits the sparse relationships among the neighboring instances instead of dense representations such as feature vectors and the distance matrix. Discarding such a large amount of information can be rationalized from the empirical results of the k-nearest neighbor algorithm, which is successful in many applications using a small k. When the number of relevant neighbors is limited, a dense representation can be redundant and a sparse representation can induce efficient learning.

References

1. Alcalá-Fdez, J., Sánchez, L., García, S., del Jesus, M., Ventura, S., Garrell, J., Otero, J., Romero, C., Bacardit, J., Rivas, V., Fernández, J., Herrera, F.: Keel: a software tool to assess evolutionary algorithms for data mining problems. Soft Computing 13(3), 307–318 (2009).
2. Angiulli, F.: Fast nearest neighbor condensation for large data sets classification. IEEE Trans. on Knowl. and Data Eng. 19(11), 1450–1464 (2007).
3. Bache, K., Lichman, M.: UCI Machine Learning Repository. http://archive.ics.uci.edu/ml (2013).
4. Berndt, D.J., Clifford, J.: Using Dynamic Time Warping to Find Patterns in Time Series. In: Proceedings of KDD-94: AAAI Workshop on Knowledge Discovery in Databases, pp. 359–370, Seattle, Washington (1994).
5. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York, NY, USA (2004).
6. Brighton, H., Mellish, C.: Advances in instance selection for instance-based learning algorithms. Data Min. Knowl. Discov. 6(2), 153–172 (2002).
7. Cano, J.R., Herrera, F., Lozano, M.: Using evolutionary algorithms as instance selection for data reduction in kdd: An experimental study. Trans. Evol. Comp 7(6), 561–575 (2003).
8. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The UCR time series classification archive (2015).
9. Devroye, L., Györfi, L., Lugosi, G.: Condensed and edited nearest neighbor rules. In: A Probabilistic Theory of Pattern Recognition, Stochastic Modelling and Applied Probability, vol. 31, pp. 303–313. Springer New York (1996).
10. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: Experimental comparison of representations and distance measures. Proc. VLDB Endow. 1(2), 1542–1552 (2008).
11. Fayyad, H.A., Atiya, A.F.: A novel template reduction approach for the k-nearest neighbor method. Trans. Neur. Netw. 20(5), 890–896 (2009).
12. Ferrandiz, S., Boullé, M.: Bayesian instance selection for the nearest neighbor rule. Mach. Learn. 81(3), 229–256 (2010).
13. Garain, U.: Prototype reduction using an artificial immune model. Pattern Anal. Appl. 11(3-4), 353–363 (2008).
14. García, S., Cano, J.R., Herrera, F.: A memetic algorithm for evolutionary prototype selection: A scaling up approach. Pattern Recogn. 41(8), 2693–2709 (2008).
15. García, S., Derrac, J., Cano, J., Herrera, F.: Prototype selection for nearest neighbor classification: Taxonomy and empirical study. IEEE Trans. Pattern Anal. Mach. Intell. 34(3), 417–435 (2012).
16. García-Pedrajas, N.: Constructing ensembles of classifiers by means of weighted instance selection. Trans. Neur. Netw. 20(2), 258–277 (2009).
17. García-Pedrajas, N., De Haro-García, A.: Boosting instance selection algorithms. Know.-Based Syst. 67, 342–360 (2014).
18. Kadous, M.W., Sammut, C.: Classification of Multivariate Time Series and Structured Data Using Constructive Induction. Mach. Learn. 58(2-3), 179–216 (2005).
19. Kelley, C.: Iterative Methods for Optimization. Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics (1999).
20. Kontorovich, A., Weiss, R.: Fast minimization of structural risk by nearest neighbor rule. Trans. Neur. Netw. 14(1), 127–137 (2003).
21. Kontorovich, A., Weiss, R.: Maximum margin multiclass nearest neighbors. In: Proceedings of The 31st International Conference on Machine Learning, pp.892–900 (2014).
22. Li, Y., Hu, Z., Cai, Y., Zhang, W.: Support vector based prototype selection method for nearest neighbor rules. In: Proceedings of the First International Conference on Advances in Natural Computation - Volume Part I, ICNC’05, pp. 528–535. Springer-Verlag, Berlin, Heidelberg (2005).
23. Lumini, A., Nanni, L.: A clustering method for automatic biometric template selection. Pattern Recognition 39(3), 495 – 497 (2006).
24. Marchiori, E.: Class conditional nearest neighbor for large margin instance selection. IEEE Trans. Pattern Anal. Mach. Intell. 32(2), 364–370 (2010).
25. Olvera-López, J.A., Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F.: A new fast prototype selection method based on clustering. Pattern Anal. Appl. 13(2), 131–141 (2010).
26. Olvera-López, J.A., Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F., Kittler, J.: A review of instance selection methods. Artif. Intell. Rev. 34(2), 133–143 (2010).
27. Pkalska, E., Duin, R.P.W., Paclík, P.: Prototype selection for dissimilarity-based classifiers. Pattern Recogn. 39(2), 189–208 (2006).
28. Riquelme, J.C., Aguilar-Ruiz, J.S., Toro, M.: Finding representative patterns with ordered projections. Pattern Recognition 36(4), 1009 – 1018 (2003).
29. Srisawat, A., Phienthrakul, T., Kijsirikul, B.: Svknn: An algorithm for improving the efficiency of k-nearest neighbor. In: Q. Yang, G. Webb (eds.) PRICAI 2006: Trends in Artificial Intelligence, Lecture Notes in Computer Science, vol. 4099, pp. 975–979. Springer Berlin Heidelberg (2006).
30. Triguero, I., Derrac, J., Garcia, S., Herrera, F.: A taxonomy and experimental study on prototype generation for nearest neighbor classification. Trans. Sys. Man Cyber Part C 42(1), 86–100 (2012).
31. Weinberger, K. Q., Saul, L. K.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res., 10, 207–244 (2009).
32. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Mach. Learn. 38(3), 257–286 (2000).
33. Wu, X., Kumar, V.: The Top Ten Algorithms in Data Mining. 1st edn. Chapman & Hall/CRC (2009).
34. Xi, X., Keogh, E., Shelton, C., Wei, L., Ratanamahatana, C.A.: Fast Time Series Classification using Numerosity Reduction. In: ICML '06: Proceedings of the 23rd International Conference on Machine Learning, pp. 1033–1040. ACM, New York, NY, USA (2006).
35. Zhang, H., Sun, G.: Optimal reference subset selection for nearest neighbor classification by tabu search. Pattern Recognition 35(7), 1481 – 1490 (2002).
Appendix

A Tables

Table 3 Dataset Properties (Vector)

| Dataset         | #Ex. | #Atts. | #CL |
|------------------|------|--------|-----|
| appendicitis     | 106  | 7      | 2   |
| australian       | 690  | 14     | 2   |
| automobile       | 205  | 25     | 6   |
| balance          | 625  | 4      | 3   |
| banana           | 5300 | 2      | 2   |
| bands            | 539  | 19     | 2   |
| breast           | 286  | 9      | 2   |
| bupa             | 345  | 6      | 2   |
| chess            | 3196 | 36     | 2   |
| cleveland        | 297  | 13     | 5   |
| contraceptive   | 1473 | 9      | 3   |
| crx              | 690  | 15     | 2   |
| dermatology      | 366  | 33     | 6   |
| ecoli            | 336  | 7      | 8   |
| flare            | 1066 | 11     | 2   |
| german           | 1000 | 20     | 2   |
| glass            | 214  | 9      | 7   |
| haberman         | 306  | 3      | 2   |
| hayes-roth       | 160  | 4      | 3   |
| heart            | 270  | 13     | 2   |
| hepatitis        | 155  | 19     | 2   |
| housevotes       | 435  | 16     | 2   |
| ionosphere       | 351  | 33     | 2   |
| iris             | 150  | 4      | 3   |
| led7digit        | 500  | 7      | 10  |
| lymphography     | 148  | 18     | 4   |
| mammographic     | 961  | 5      | 2   |
| monk-2           | 432  | 6      | 2   |
| movement-libras  | 360  | 90     | 15  |
| newthyroid       | 215  | 5      | 3   |
| phoneme          | 5404 | 5      | 2   |
| pima             | 768  | 8      | 2   |
| ring             | 7400 | 20     | 2   |
| saheart          | 462  | 9      | 2   |
| sonar            | 208  | 60     | 2   |
| spambase         | 4597 | 57     | 2   |
| spectfheart      | 267  | 44     | 2   |
| tae              | 151  | 5      | 3   |
| tic-tac-toe      | 958  | 9      | 2   |
| titanic          | 2201 | 3      | 2   |
| twonorm          | 7400 | 20     | 2   |
| vehicle          | 846  | 18     | 4   |
| vowel            | 990  | 13     | 11  |
| wdbc             | 569  | 30     | 2   |
| wine             | 178  | 13     | 3   |
| wisconsin        | 699  | 9      | 2   |
| yeast            | 1484 | 8      | 10  |
| zoo              | 101  | 16     | 7   |

Table 4 Error and Selection Rates (Time Series)

| Dataset            | NoPS | ERR  | SLR  |
|--------------------|------|------|------|
| 50words            | 0.068| 0.068| 0.96 |
| Adiac              | 0.42 | 0.43 | 0.94 |
| Beef               | 0.37 | 0.37 | 0.97 |
| CBF                | 0.013| 0.012| 0.97 |
| Chlorine           | 0.37 | 0.38 | 0.99 |
| Coffee             | 0.036| 0.036| 0.96 |
| Diatom             | 0.37 | 0.38 | 0.99 |
| ECG200             | 0.1  | 0.13 | 0.88 |
| ECG-CinC           | 0.37 | 0.38 | 0.99 |
| ECG-5Days          | 0.19 | 0.30 | 0.52 |
| FaceAll            | 0.068| 0.068| 0.96 |
| FaceFour           | 0.076| 0.076| 0.99 |
| FacesUCR           | 0.027| 0.040| 0.88 |
| Fish               | 0.18 | 0.17 | 0.90 |
| GunPoint           | 0.060| 0.060| 0.94 |
| Lighting2          | 0.23 | 0.23 | 0.99 |
| Lighting7          | 0.24 | 0.25 | 0.90 |
| MedicalImages      | 0.13 | 0.16 | 0.90 |
| MoteStrain         | 0.16 | 0.17 | 0.93 |
| OliveOil           | 0.13 | 0.13 | 0.97 |
| Plane              | 0.0095| 0.019| 0.70 |
| PowerDemand        | 0.18 | 0.34 | 0.72 |
| RobotSurface       | 0.089| 0.098| 0.52 |
| RobotSurface2      | 0.29 | 0.28 | 0.95 |
| Synthetic          | 0.080| 0.080| 0.92 |
| Trace              | 0.11 | 0.20 | 0.52 |
| TwoPatterns        | 0.32 | 0.32 | 0.97 |
### Table 5 Dataset Properties (Time Series)

| Data          | # Cl. | # Train | # Test | TS Length |
|---------------|-------|---------|--------|-----------|
| 50Words       | 50    | 450     | 455    | 270       |
| Adiac         | 37    | 390     | 391    | 176       |
| Beef          | 5     | 30      | 30     | 470       |
| CBF           | 3     | 30      | 900    | 128       |
| Chlorine      | 3     | 467     | 3840   | 166       |
| Coffee        | 2     | 28      | 28     | 286       |
| Diatom        | 4     | 16      | 306    | 345       |
| ECG200        | 2     | 100     | 100    | 96        |
| ECG-5Days     | 2     | 23      | 861    | 136       |
| ECG-CinC      | 4     | 40      | 1380   | 1639      |
| FaceAll       | 14    | 560     | 1690   | 131       |
| FaceFour      | 4     | 24      | 88     | 350       |
| FacesUCR      | 14    | 200     | 2050   | 131       |
| Fish          | 7     | 175     | 175    | 463       |
| GunPoint      | 2     | 50      | 150    | 150       |
| Lightning-2   | 2     | 60      | 61     | 637       |
| Lightning-7   | 7     | 70      | 73     | 319       |
| MedicalImages | 10    | 381     | 760    | 99        |
| MoteStrain    | 2     | 20      | 1252   | 84        |
| OliveOil      | 4     | 30      | 30     | 570       |
| Plane         | 7     | 105     | 105    | 144       |
| PowerDemand   | 2     | 67      | 1029   | 24        |
| RobotSurface  | 2     | 20      | 601    | 70        |
| RobotSurface2 | 2     | 27      | 953    | 65        |
| Synthetic     | 6     | 300     | 300    | 60        |
| Trace         | 4     | 100     | 100    | 275       |
| Two Patterns  | 4     | 1000    | 4000   | 128       |

### Table 6 Parameter Specifications for Baseline Methods

| Baseline | Parameters | Distance/Kernel |
|----------|------------|-----------------|
| FCNN     | neighbor size $\{1,2,3\}$ | Euclidean       |
| PSC      | # clusters $\{8,12,16\}$    |                 |
| SVBPS    | $C \in \{0.1,1,10,100,1000\}$ | Euclidean/RBF   |
| TRKNN    | $\alpha \in \{1.01,21.5\}$ | Euclidean       |
| Summary of Error and Selection Rates (Vector Data) |
|-----------------------------------------------|
| NoPS | REPS | CCIS | FCNN | PSC | SVBPS | TRKNN |
| 0.026 | 0.046 | 0.12 | 0.47 | 0.38 | 0.0036 | 0.0071 |
| 0.68 | 0.18 | 0.28 | 0.69 | 0.49 | 0.57 | 0.69 |
| 0.90 | 0.88 | 0.96 | 0.90 | 0.88 | 0.96 | 0.90 |
| 0.0088 | 0.017 | 0.035 | 0.055 | 0.031 | 0.015 | 0.024 |
| 0.08 | 0.015 | 0.03 | 0.045 | 0.028 | 0.012 | 0.018 |
| 0.56 | 0.22 | 0.32 | 0.57 | 0.43 | 0.35 | 0.42 |
| 0.34 | 0.21 | 0.35 | 0.38 | 0.36 | 0.35 | 0.36 |
| 0.79 | 0.47 | 0.64 | 0.85 | 0.76 | 0.67 | 0.74 |

| Summary of Error and Selection Rates (Vector Data) |
|-----------------------------------------------|
| NoPS | REPS | CCIS | FCNN | PSC | SVBPS | TRKNN |
| 0.026 | 0.046 | 0.12 | 0.47 | 0.38 | 0.0036 | 0.0071 |
| 0.68 | 0.18 | 0.28 | 0.69 | 0.49 | 0.57 | 0.69 |
| 0.90 | 0.88 | 0.96 | 0.90 | 0.88 | 0.96 | 0.90 |
| 0.0088 | 0.017 | 0.035 | 0.055 | 0.031 | 0.015 | 0.024 |
| 0.08 | 0.015 | 0.03 | 0.045 | 0.028 | 0.012 | 0.018 |
| 0.56 | 0.22 | 0.32 | 0.57 | 0.43 | 0.35 | 0.42 |
| 0.34 | 0.21 | 0.35 | 0.38 | 0.36 | 0.35 | 0.36 |
| 0.79 | 0.47 | 0.64 | 0.85 | 0.76 | 0.67 | 0.74 |