Graph-based Discrete Differential Geometry for Critical Instance Filtering

Elena Marchiori

Department of Computer Science, Radboud University, Nijmegen, The Netherlands

Abstract. Graph theory has been shown to provide a powerful tool for representing and tackling machine learning problems, such as clustering, semi-supervised learning, and feature ranking. This paper proposes a graph-based discrete differential operator for detecting and eliminating competence-critical instances and class label noise from a training set in order to improve classification performance. Results of extensive experiments on artificial and real-life classification problems substantiate the effectiveness of the proposed approach.

1 Introduction

In graph-based data analysis, a dataset is represented as a graph, where the vertices are the instances of the dataset and the edges encode a pairwise relationship between instances. For instance, the nearest neighbor relation between points of a finite set in the Euclidean space can be described by the popular nearest neighbor (proximity) graph [3, 27]. Concepts and methods from graph theory are then used for extracting knowledge from such a representation. In particular, the graph Laplacian provides a natural interpretation to the geometric structure of datasets. It has been used in machine learning for tackling diverse tasks such as dimensionality reduction and clustering, e.g., [4, 29], feature selection, e.g., [19, 34], and semi-supervised learning, e.g., [35, 36].

This paper shows how the graph Laplacian operator can be directly used for filtering competence-critical instances and class label noise from a training set, in order to improve test accuracy.

Research on instance selection focusses mainly on three types of filtering techniques [8]: competence preservation, competence enhancement, and hybrid approaches. *Competence preservation* algorithms, e.g., [1, 14], remove irrelevant points, that is, that do not affect the classification accuracy of the training set. *Competence enhancement* methods, e.g., [23, 26, 28, 31], remove noisy points, such as those with a wrong class label, as well as points close to the decision boundary, yielding to smoother decision boundaries, in order to increase classifier accuracy. *Hybrid* methods, e.g., [8, 20, 24, 25, 32], aim at finding a subset of the training set that is both noise free and does not contain irrelevant points. Alternative methods use prototypes instead of instances of the training set, see for instance [21].

The algorithm proposed here belongs to the so-called *competence enhancement* methods. It differs from previous methods for this task in the way it extracts information from the neighborhood of an instance in order to measure its relevance. Indeed, while previous methods are based on 'static' measures, such as being correctly classified, the proposed method uses a 'differential' measure, defined by means of a graph Laplacian operator. Specifically, we consider the graph-based representation of two class dependent K nearest neighbor (KNN) relations defined over pairs of instances: the within- and between- class KNN. The between-class KNN graph is used to define a graph Laplacian operator. The within-class KNN graph is used to define the within-class degree function, mapping vertices to their degree.

The application of the Laplacian operator to such function, called *Laplace scoring*, provides such differential measure of instance relevance. It measures the flow of the within-class degree function at each vertex of the between-class KNN graph. Vertices with negative Laplace score are either close to the KNN decision boundary or are outliers. This motivates the introduction of a simple Laplace-based instance filtering algorithm, which removes instances having negative Laplace score.

To the best of our knowledge, this work presents the first attempt to perform class noise instance filtering using a graph-based differential approach.

In order to test comparatively the effectiveness of this approach, extensive experiments on artificial and real-life data sets are conducted. We consider three classifiers: the KNN classifier without instance filtering, with the popular Wilson's editing [31], and with Laplace filtering. Results of the experiments indicate best test accuracy performance of Laplace filtering over the other methods, as well as superior robustness with respect to the presence of class noise in the training set.

Furthermore, comparison of Laplacian filtering with state-of-the-art editing algorithms indicate similar or improved generalization performance of the `1-NN`.

Finally, we investigate the use of Laplacian filtering for improving the performance of classifiers other than `1-NN`. We consider SVMs with RBF kernels. These are related to NN methods, because each RBF measures the distance of a test instance to one of the training instances. SVM training keeps certain training instances as support vectors, and discards others. In this way, SVM/RBF may also be viewed as a competence-enhancement filtering method. We investigate the effect of Laplacian filtering as pre-processing step by performing experiments on datasets with different levels of added class noise. Results of these experiments show that at all considered levels of class noise, Laplacian filtering has no significant positive effect on the generalization performance of SVMs with RBF kernel.

In general, the results substantiate the effectiveness of graph Laplacian operators for tackling the class noise filtering problem. Therefore this contribution adds yet another successful application of such powerful graph-based framework in machine learning.

We begin by setting the stage with the notation and main concepts used in this study.

2 Background

In this paper we use X to denote a dataset of n instances $X = \{x_1, \ldots, x_n\}$, where x_i is a real-valued vector of dimension m. Let C denote the set of class labels of X, and let $l : X \to C$ the function mapping each instance x_i to its class label $l(x_i)$.

A graph $G = (V, E)$ consists of a finite set V and a subset $E \subset V \times V$. The elements of V are the *vertices* of the graph and those of E are the edges of the graph. In this work we consider undirected graphs, that is, such that for each edge $(u, v) \in E$ we have also $(v, u) \in E$. We say that u and v are *adjacent* vertices, denoted by $u \sim v$, if $(u, v) \in E$. The degree function d of a graph is defined by $d(u) = |\{v \mid u \sim v\}|$, where $|S|$ denotes the cardinality of a set S.

The graph normalized Laplacian can be defined as follows. Suppose $|V| = n$. Consider the $n \times n$ matrix L, defined as

$$L(u,v) = \begin{cases} d(u) & \text{if } u = v, \\ -1 & \text{if } (u,v) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

The (normalized) Laplacian of G is the $n \times n$ matrix

$$\mathcal{L}(u,v) = \begin{cases} 1 & \text{if } u = v \text{ and } d(u) \neq 0, \\ \frac{-1}{\sqrt{d(u)d(v)}} & \text{if } (u,v) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

The graph Laplacian operator $\mathcal{L}$ maps real-valued functions on vertices to real-valued functions on vertices, defined by

$$\mathcal{L}(f)(u) = \frac{1}{\sqrt{d(u)}} \sum_{u \sim v} \left(\frac{f(u)}{\sqrt{d(u)}} - \frac{f(v)}{\sqrt{d(v)}} \right).$$

3 Laplacian Instance Filtering

Denote by l a generic element of C. Let X_l be the subset of X consisting of those instances having class label equal to l. Define $KNN(x, l)$ to be the set of K nearest neighbors of x computed among those instance in X_l, excluding x.

Define the following two graphs. The *within-class KNN graph*, denoted by $G_{wc} = (V, E_{wc})$, such that $V = X$, and

$$E_{wc} = \{(x_i, x_j) \mid x_j \in \texttt{KNN}(x_i, l(x_i)) \text{ or } x_i \in \texttt{KNN}(x_j, l(x_j))\}.$$

G_{wc} represents the (symmetric) nearest neighbor relation between points of the same class in the training set.

Analogously, define the *between-class KNN graph*, denoted by $G_{bc} = (V, E_{bc})$, such that $V = X$,

$$E_{bc} = \{(x_i, x_j) \mid (x_j \in \texttt{KNN}(x_i, l) \text{ and } l \neq l(x_i)) \text{ or } (x_i \in \texttt{KNN}(x_j, l) \text{ and } l \neq l(x_j))\}.$$

G_{bc} represents the (symmetric) KNN relation between points of each pair of different classes in the training set. Note that this relation differs from the nearest unlike neighbor relation (NUN) [15] because it considers all pairs of different classes, while NUN considers one class versus the union of all the other classes. Clearly, for binary classification problems the two relations coincide.

The within- and between-class graphs with $K = 1$ for a toy binary classification problem are shown in Figure 1.

Let $\mathcal{L}$ be the Laplacian operator of G_{bc}. Let g denote the within-class degree function, mapping vertex i to its degree in G_{wc}. $g(i)$ can be viewed as an estimate of the density of points of class label $l(x_i)$ around x_i, since the more instances with label $l(x_i)$ are close to x_i, the larger the $g(i)$ will be [34].

We define the *Laplace score*, denoted by $Score$, to be the real-valued function on vertices such that

$$Score(u) = \mathcal{L}(g)(u).$$

This function assigns a small score to an instance whose neighbors from different classes (that is, its adjacent vertices in G_{bc}) are in a region containing many points of their own class, and few points of classes different from their one.

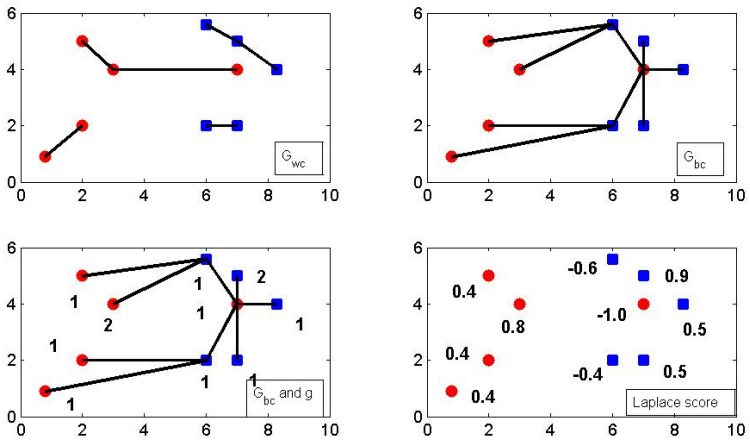

Fig. 1. Graphs and Laplace score with $K = 1$ of a training set for a toy binary classification problem in the real plane.

The within-class degree and Laplace score of instances for the considered toy classification example are shown in Figure 1. Observe that in this example points with negative Laplace score are close to the one nearest neighbor class decision boundary. This motivates the introduction of the simple Algorithm 1 for class noise filtering, which removes from the training set those instances with negative Laplace score.

Algorithm 1 Laplace instance filtering

Input: training data X of size n
number K of nearest neighbors
Output: subset S of X
G_{bc} = between-class KNN graph of X
G_{wc} = within-class KNN graph of X
g = degree function of G_{wc}
for $i = 1$ **to** n **do**
  $Score(i) = \mathcal{L}(g)(i)$
end for
$S = \{x_i \in X \mid Score(i) \geqslant 0\}$

The time complexity of this algorithm is dominated by the cost of building the between- and within-class graphs, which is quadratic in the size of the training set. However, this bound can be reduced to $O(n \log(n))$ (for small input dimension) by using metric trees or other spatial data structures, as shown for example in [18], as well as structures optimized for a data and query distribution [5, 9].

Application of Laplace instance filtering to an instance of the XOR classification problem is illustrated in Figure 2. Points filtered out by the algorithm are highlighted with circles.

3.1 Justification

The score of an instance can be interpreted as a discrete divergence measure. This can be shown by using discrete analogous of differential operators. To this end, we use the results contained in [36].

Indeed, consider the graph gradient operator, mapping real-valued functions of the vertices into real-valued functions on edges.

$$(\nabla\phi)(u,v) = \frac{\phi(v)}{d(v)} - \frac{\phi(u)}{d(u)}.$$

Observe that in this definition, before computing the variation of ϕ between two adjacent vertices, the function value is split at each vertex along its adjacent edges.

The graph gradient can also be defined at vertex v, as

$$\nabla\phi(u) = \{(\nabla\phi)(u,v) \mid (u,v) \in E\}.$$

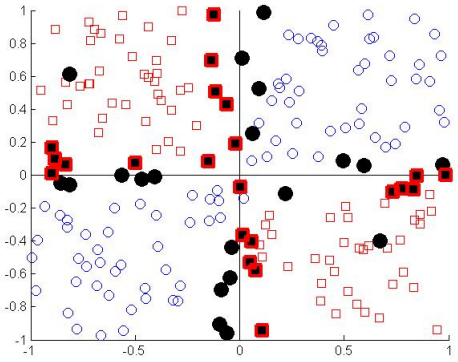

Fig. 2. Application of Laplace filtering with $K = 1$ to an instance, with class noise examples, of the XOR classification problem in the real plane. The points removed have filled markings.

The graph divergence maps real-valued functions of the edges into real-valued functions on vertices.

$$(\texttt{div}\ \ \psi)(u) = \sum_{u \sim v} \frac{1}{\sqrt{d(v)}} (\psi(u,v) - \psi(v,u)).$$

The divergence measures the net outflow of function ψ at each vertex.
The following equality relates the graph divergence and Laplacian operators:

$$\mathcal{L}(\phi) = -\frac{1}{2}\texttt{div}\ \ (\nabla\phi).$$

By instantiating the above formula with the graph Laplacian of G_{bc} and the within-class degree function g we obtain

$$Score = -\frac{1}{2}\texttt{div}\ \ (\nabla g).$$

Therefore, the Laplace instance score is a measure of negative divergence. An instance having high divergence value (hence small Score value) can be considered critical, since there is a high flow of within-class degree at that instance in a neighborhood characterized by means of the between-class graph.

4 Experiments

In order to assess comparatively the accuracy performance of the proposed filtering method, we conduct extensive experiments on 19 Machine Learning datasets, using the K-nearest neighbor classifier (KNN) with no training set pre-processing

Table 1. Datasets used in the experiments. CL = number of classes, TR = training set, TE = test set, VA = number of variables, Cl.Inst. = number of instances in each class.

Dataset	CL	VA	TR	Cl.Inst.	TE	Cl.Inst.
1 Banana	2	2	400	212-188	4900	2712-2188
2 B.Cancer	2	9	200	140-60	77	56-21
3 Diabetes	2	8	468	300-168	300	200-100
4 German	2	20	700	478-222	300	222-78
5 Heart	2	13	170	93-77	100	57-43
6 Image	2	18	1300	560-740	1010	430-580
7 Ringnorm	2	20	400	196-204	7000	3540-3460
8 F.Solar	2	9	666	293-373	400	184-216
9 Splice	2	60	1000	525-475	2175	1123-1052
10 Thyroid	2	5	140	97-43	75	53-22
11 Titanic	2	3	150	104-46	2051	1386-66
12 Twonorm	2	20	400	186-214	7000	3511-3489
13 Waveform	2	21	400	279-121	4600	3074-1526
14 Iris	3	4	120	40-40-40	30	10-10-10
15 Breast-W	2	9	546	353-193	137	91-46
16 Bupa	2	6	276	119-157	69	26-43
17 Pima	2	8	615	398-217	153	102-51
18 g50	2	50	550	252-248	50	23-27
19 g10n	2	10	550	245-255	50	29-21

[13], here called *No-filtering*, with *Laplace* instance filtering, and with the popular *Wilson*'s filtering algorithm. The latter one removes those instances that do not agree with the majority of its K nearest neighbors. We consider three instances of each algorithm obtained by setting the number K of neighbors to $1, 3, 5$, respectively, resulting in a total of nine classifiers.

Cross validation is applied to each dataset. Specifically, for each partition of the dataset, each filtering algorithm is applied to the training set X from which a subset S is returned. The KNN classifier that uses only points of S is applied to the test set.

4.1 Datasets

We consider 3 artificial datasets (Banana, g50c, g10n) and 16 real-life ones, with different characteristics as shown in Table 1. These datasets have been used in previous studies on model selection for (semi)supervised learning.

Specifically, Raetsch's binary classification benchmark datasets have been used in [22]: they consists of 1 artificial and 12 real-life datasets from the UCI, DELVE and STATLOG benchmark repositories. For each experiment, the 100 (20 for `Splice` and `Image`) partitions of each dataset into training and test set available in the repository are used here.

Two artificial binary classification problems from Chapelle's benchmark datasets [11], `g50c` and `g10n`, are generated from two standard normal multi-variate Gaussians. In `g50c`, the labels correspond to the Gaussians, and the means are located in a 50-dimensional space such that the Bayes' error is 5%. In contrast, `g10n` is a deterministic problem in 10 dimensions, where the decision function traverses the centers of the Gaussians, and depends on only two of the input dimensions. For each experiment, the 10 partitions of each dataset into training and test set available in the repository are used.

Finally, four standard benchmark datasets from the UCI Machine Learning repository are used: `Iris`, `Bupa`, `Pima`, and `Breast-W`. For each experiment, 100 partitions of each dataset into training and test set are used. Each partition randomly divides the dataset into training and test set, equal to 80% and 20% of the data, respectively.

4.2 Results

Results of the experiments are summarized in Table 2 (also plotted in the first row of Figure 3). The table contains average accuracy results of the algorithms on each classification task, their average and median, the outcome of a paired t-test on the results of each classification task, and the outcome of a paired Wilcoxon test on the (average) results of the entire set of classification tasks.

Results of a paired t-test at a 0.01 significance level shows improved accuracy performance of Laplace (see row 'S' in Table 2) on the majority of the datasets. Application of the non parametric Wilcoxon test for paired samples at a 0.01 significance level to the average results on the entire set of classification tasks, indicates that KNN with Laplace filtering outperforms the other algorithms.

In summary, the experimental analysis indicates effectiveness of Laplace-based instance filtering and robustness with respect to the presence of high number of variables, training examples, noise and irrelevant variables.

We turn now to the experimental analysis of classifier robustness with respect to the presence of class noise.

4.3 Robustness to Class Noise

In order to analyze experimentally the robustness of the methods with respect to the presence of class noise in the training set, all experiments are repeated with modified training sets. The new training sets are obtained by changing the class labels of a given percentage γ of randomly selected instances.

Figure 3 shows plots of the average accuracy of the nine KNN classifiers using the original datasets and those obtained by adding γ% class noise, with $\gamma = 10, 20, 40$. The Figure contains four rows, one for each value of γ (the original training set corresponds to setting $\gamma = 0$). Each row contains three plots, one for each value of K. Each plot shows average test accuracy of No-filtering, Laplace, and Wilson algorithms for the specific value of K and γ.

In all the considered cases, the average test accuracy curve of Laplace dominates those of the other two algorithms, with more improvement for higher

Table 2. A(M) = average (median) results over datasets. S = +/- number of times Laplace average accuracy is significantly better (+) or significantly worse (-) than the other algorithm, according to a paired t-test at 0.01 significance level. W = a '+' indicates Laplace significantly better than the other algorithm at a 0.01 significance level according to a Wilcoxon test for paired samples.

	Laplace			Wilson			No-Filtering		
	1	3	5	1	3	5	1	3	5
1 Banana	88.5	88.9	88.7	87.8	88.2	88.3	86.4	87.9	88.3
2 B.Cancer	70.6	73.0	74.2	69.4	73.4	73.6	67.3	68.5	71.2
3 Diabetes	73.9	74.0	73.8	72.7	73.2	73.6	69.9	72.4	72.6
4 German	74.0	74.0	73.1	73.0	73.9	73.9	70.5	73.1	74.1
5 Heart	81.6	82.6	82.9	80.6	82.0	82.7	76.8	80.5	81.8
6 Image	94.9	92.3	90.8	95.8	94.6	94.1	96.6	95.7	95.1
7 Ringnorm	67.3	63.2	61.0	54.8	51.2	50.6	64.9	59.5	56.7
8 F.Solar	64.0	64.6	64.7	61.4	62.8	62.7	60.7	62.3	62.2
9 Splice	73.3	76.4	77.1	68.4	68.2	66.7	71.1	72.6	73.3
10 Thyroid	94.3	91.9	89.5	94.0	91.9	89.5	95.6	93.8	92.6
11 Titanic	77.2	77.0	77.2	67.3	72.5	74.5	66.9	72.3	74.0
12 Twonorm	95.5	96.6	96.9	94.1	95.9	96.4	93.3	95.5	96.2
13 Waveform	86.2	87.4	87.3	85.4	86.9	87.5	84.1	86.3	87.3
14 Iris	95.2	95.1	94.8	96.1	95.5	95.8	95.6	95.1	95.8
15 Breast-W	97.1	97.3	97.1	96.9	97.2	96.9	96.2	97.1	97.4
16 Bupa	65.8	69.2	68.4	63.5	67.0	67.4	61.2	64.3	66.5
17 Pima	72.5	74.2	75.0	69.6	73.1	73.8	67.3	69.9	72.0
18 g50	85.6	89.8	91.2	82.2	87.2	92.4	79.6	88.4	92.0
19 g10	74.6	79.0	80.8	74.0	79.2	80.0	75.2	78.4	78.2
A	80.6	81.4	81.3	78.3	79.7	80.0	77.9	79.7	80.4
M	77.3	79.0	80.8	74.0	79.2	80.0	75.2	78.4	78.2
S	n/a	n/a	n/a	14/2	15/3	11/2	13/2	7/3	11/5
W	n/a	n/a	n/a	+	+	+	+	+	+

values of K. Indeed, in all these cases, KNN with Laplace filtering outperforms significantly the other classifiers.

These results substantiate robustness of the Laplace-based instance filtering approach for KNN with respect to the presence of class noise.

5 Comparison with Other Methods

5.1 Editing Algorithms

In order to compare the performance of the proposed method with that of state-of-the-art editing algorithms, we report in Figure 3 the test accuracy results of the `1-NN` classifier, achieved by the state-of-the-art instance editing algorithms recently investigated in [20]: Iterative Case Filtering (`ICF`) [7], Decremental Re-

duction Optimization (DROP3) [32, 33], and Hit Miss Network Editing (HMN-EI) [20].

ICF first applies E-NN noise reduction iteratively until it cannot remove any point, and next iteratively removes points. At each iteration all points for which the so-called *reachability* set is smaller than the *coverage* one are deleted. The reachability of a point x consists of the points inside the largest hyper-sphere containing only points of the same class as x. The *coverage* of x is defined as the set of points that contain x in their reachability set.

DROP3 first applies a pre-processing step which discards points of X misclassified by their K nearest neighbors, and then removes a point x from X if the accuracy of the KNN rule on the set of its associates does not decrease. Each point has a list of K nearest neighbors and a list of associates, which are updated each time a point is removed from X. A point y is an *associate* of x if x belongs to the set of K nearest neighbors of y. If x is removed then the list of K nearest neighbors of each of its associates y is updated by adding a new neighbor point z, and y is added to the list of associates of z. The removal rule is applied to the points sorted in decreasing order of distance from their nearest neighbor from the other classes (nearest enemy).

HMN-EI is an iterative heuristic algorithm based on a directed graph-based representation of the training set, called hit miss network. Topological properties of such network are used for designing an iterative algorithm that removes points considered irrelevant or harmful to the 1-NN generalization performance. The empirical error on the training set is used as criterion for terminating the iterative process.

Results in Figure 3 show that test accuracy of Laplace filtering is similar or better than the one of these state-of-the-art methods. However, observe that editing algorithms also reduce storage, while Laplace filtering is specifically designed for removing critical instances. In order to reduce also storage reduction, one could use Laplace instance filtering as pre-processing step, followed by the application of a competence preservation algorithm, such as [1].

5.2 SVM with RBF kernels and optimized parameters

A family of classifiers different from KNN, whose training process results in the selection of a subset of the training set, are the Support Vector Machines (SVMs). They map training points x into a higher (possibly infinite) dimensional feature space by the function θ. Then SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional feature space. Given a training set of real-valued instance-label pairs $(x_i, l(x_i)), i = 1, \ldots, n$ the support vector machines (SVM) [6, 12] require the solution of the following optimization problem:

$$\min_{w,b,\xi} w^T w + C \sum_{i=1}^{n} \xi_i$$

$$\text{such that } l(x_i)(w^T \xi(x_i) + b) \geqslant 1 - \xi_i \text{ and } \xi_i \geqslant 0.$$

Dataset	HMN-EI	ICF	DROP3	LAPLACE
BANANA	88.6	86.1	87.6	88.5
B.CANCER	69.2	67.0	69.7	70.6
DIABETES	73.5	69.8	72.3	73.9
GERMAN	72.9	68.6	72.0	74.0
HEART	81.6	76.7	80.2	81.6
IMAGE	92.7	93.8	95.1	94.9
RINGNORM	65.6	61.2	54.7	67.3
F.SOLAR	64.7	61.0	61.4	64.0
SPLICE	70.7	66.3	67.6	73.3
THYROID	93.2	91.9	92.7	94.3
TITANIC	76.0	67.5	67.7	77.2
TWONORM	95.9	89.2	94.3	95.5
WAVEFORM	85.4	82.1	84.9	86.2
IRIS	95.4	95.3	95.8	95.2
BREAST-W	96.9	95.4	96.8	97.1
BUPA	64.5	60.9	63.1	65.8
PIMA	71.7	67.9	69.4	72.5
G50	86.8	82.2	82.8	85.6
G10	79.2	73.0	75.0	74.6
Average	80.2	76.6	78.1	80.6
Median	79.2	73.0	75.0	77.3
S	10/2	18/0	13/0	N/A
W	$\sim$	+	+	N/A

Table 3. Results of experiments on ML benchmark datasets of HMN-EI, ICF, DROP3, and Laplace filtering.

$C > 0$ is the penalty parameter of the empirical error term. Furthermore, $K(x, y) = \xi(x)^T \xi(y)$ is called the kernel function. In particular, SVMs with Radial Basis Function (RBF) kernel use $K(x, y) = e^{-\sigma||x-y||^2}$. The set of points with $\xi_i > 0$ are called support vectors. They uniquely identify the separating hyperplane.

It is interesting to investigate whether the use of Laplacian filtering as preprocessing step improves the performance of SVMs with RBF kernel and optimized parameters.

To this aim, the experimental evaluation described in the previous section is used. Specifically, first cross-validation is applied to search for the given training set for the optimal values of the soft-margin C parameter and the RBF parameter σ[1]. Next, Laplacian filtering with $K = 1$ and Euclidean distance is applied for discarding critical instances from the training set. Finally, a SVM with RBF kernel is trained on the selected instances, using the given optimal values for σ and C.

[1] In the experiments we use the Matlab functions implemented by S. Hashemi of LIBSVM's library [10].

Results of experiments are reported in Table 4. The new training sets are obtained by changing the class labels of a given percentage γ of randomly selected instances. We consider $\gamma = 0, 20, 40$. Laplace filtering does not appear to affect significantly the test accuracy at the considered levels of class noise. This result is not very surprising, since SVMs with RBF kernel and 'optimized' parameters selection have in general good generalization accuracy in the presence of noise.

Table 4. Results of SVM/RBF with Laplace pre-processing (LAPLACE) and without (SVM-RBF) at different levels of class noise γ.

	$\gamma = 0$		$\gamma = 20$		$\gamma = 40$	
	SVM-RBF	LAPLACE	SVM-RBF	LAPLACE	SVM-RBF	LAPLACE
1	89.3	89.3	87.3	87.4	65.7	66.0
2	73.1	72.9	71.1	71.1	62.5	63.1
3	76.5	76.5	74.4	74.4	64.9	64.5
4	76.2	76.4	73.3	73.3	65.7	66.1
5	83.9	84.1	81.1	81.6	64.5	65.4
6	96.5	96.5	92.9	92.9	78.7	78.9
7	98.2	98.1	96.5	96.2	81.3	79.0
8	66.8	66.8	65.0	65.1	57.9	58.4
9	88.8	88.6	83.3	83.3	68.0	68.2
10	95.2	95.0	91.5	91.9	76.4	76.4
11	77.3	77.2	76.1	75.9	67.1	68.0
12	97.5	97.5	96.5	96.9	88.0	89.5
13	89.8	89.8	87.1	87.3	74.2	75.9
14	95.8	95.8	95.1	95.6	90.6	91.0
15	94.2	95.0	96.2	96.2	90.2	90.8
16	70.5	70.7	64.2	64.6	54.9	54.9
17	75.6	75.8	72.9	73.0	64.7	64.7
18	95.8	95.8	93.4	92.6	78.6	81.6
19	94.2	95.0	88.4	89.0	69.8	68.2
A	86.1	86.2	83.5	83.6	71.8	72.1
M	89.3	89.3	87.1	87.3	68.0	68.2
S	0/0	0/0	0/0	0/0	0/0	2/0
W	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$

6 Conclusions and Future Work

This paper introduced a graph differential operator for scoring instances of a training set. We showed how this scoring is related to the flow of the within-class density in the between-class KNN graph, and observed empirically that instances with negative score are close to the class decision boundary or are outliers. This observation motivated the design of a simple algorithm for class noise instance filtering which removes instances having negative score.

We performed extensive experiments on artificial and real-life datasets and analyzed the test accuracy of KNN classifier without filtering, with a traditional filtering algorithm, and with Laplace filtering. The results indicated superior performance of Laplace filtering over the other algorithms. Experiments with modified training sets obtained by permuting the class label of a percentage of their instances were conducted, to investigate robustness of the approach to the presence of class noise. Results of the experiments substantiated the robustness of Laplacian filtering, which achieved significantly better test accuracy performance than the other algorithms, at each of the considered levels of class noise. Comparison of Laplacian filtering with state-of-the-art editing algorithms indicated similar or improved generalization performance of the `1-NN`.

Finally, we investigated whether the use of Laplacian filtering as pre-processing step improves the performance of classifiers other than KNN. We considered SVMs with RBF kernels. These are related to NN methods, because each RBF measures the distance of a test instance to one of the training instances. SVM training keeps certain training instances as support vectors, and discards others. In this way, SVM/RBF may be view as a competence-enhancement filtering method. Results of extensive experiments seemed to indicate no significant effect of Laplacian filtering on the generalization performance of SVM with RBF kernel. The benefits of noise reduction are much more apparent for kNN because it does not really have an induction step and uses examples directly for classification. SVMs with RBF kernel and equipped with cross validation for selecting optimal values of their parameters, provide a rather powerful tool for selecting the centers and parameters of the RBF's, which is robust to the presence of noise.

In summary, these results show that Laplacian instance filtering provides a simple yet effective tool for improving accuracy performance of nearest neighbor classifiers.

We conclude with a discussion of issues and future research directions.

The Laplacian filtering algorithm does not takes into account the effect of removing one instances on the remaining ones. An adaptive approach, consisting in removing the instance having the largest negative score, and then updating the score of the remaining instances, and so on, could possibly improve the effectiveness of the algorithm. However, such an approach would increase the algorithmic complexity of the algorithm.

In the present algorithm, instances with negative Laplacian score are considered critical. Replacing such "rule of the thumb" with an incremental procedure for selecting a cutoff value will possibly have a beneficial effect. Such a procedure could be based on the leave-one-out error of the original training set, using the KNN classifier with actual set of instances incrementally constructed starting from a core subset consisting of instances with high score.

In those cases where the underlying metric is corrupted (e.g., due to irrelevant features), instance selection methods that directly depend on the underlying similarity measure, such as Laplacian filtering, may possibly fail to improve the classification performance of the KNN classifier. In such cases hybridization with

metric learning techniques (cf. e.g., [17, 16]), could help to overcome this drawback. In the metric learning approach the goal is typically to change the metric in order to repair the KNN classifier. We are investigating an hybridization of Laplacian filtering with Weinberger's et al. method [30], for effective repairing of the metric and removal of critical instances.

Another important issue in instance filtering is scalability. Recently, an instance selection method based on distributed computing has been proposed for speeding up execution of the algorithm without affecting training set accuracy [2]. It is interesting to investigate whether this approach can be used also to speed up execution of Laplace filtering, in order to allow its applicability to very large datasets.

References

1. F. Angiulli. Fast condensed nearest neighbor rule. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 25–32. ACM, 2005.
2. F. Angiulli and G. Folino. Distributed nearest neighbor-based condensation of very large data sets. *IEEE Trans. on Knowl. and Data Eng.*, 19(12):1593–1606, 2007.
3. V. Barnett. The ordering of multivariate data. *J. Roy. Statist. Soc.*, Ser. A 139(3):318–355, 1976.
4. M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*, pages 585–591. MIT Press, 2002.
5. A. Beygelzimer, S. Kakade, and J.Langford. Cover trees for nearest neighbor. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 97–104, New York, NY, USA, 2006. ACM.
6. B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, New York, NY, USA, 1992. ACM.
7. H. Brighton and C. Mellish. On the consistency of information filters for lazy learning algorithms. In *PKDD '99: Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, pages 283–288. Springer-Verlag, 1999.
8. H. Brighton and C. Mellish. Advances in instance selection for instance-based learning algorithms. *Data Mining and Knowledge Discovery*, (6):153–172, 2002.
9. L. Cayton and S. Dasgupta. A learning framework for nearest neighbor search. *NIPS*, 20, 2007.
10. C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.
11. O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 57–64, 2005.
12. C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, 1995.
13. T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.
14. B. V. Dasarathy. Minimal consistent set (mcs) identification for optimal nearest neighbor decision systems design. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(3):511–517, 1994.

15. B.V. Dasarathy. Nearest unlike neighbor (nun): An aid to decision confidence estimation. *Opt. Eng.*, 34(9):2785–2792, 1995.
16. C. Domeniconi, D. Gunopulos, , and J. Peng. Large margin nearest neighbor classifiers. *IEEE IEEE Transactions on Neural Networks*, 16(4):899909, 2005.
17. C. Domeniconi, J. Peng, and D. Gunopulos. Locally adaptive metric nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1281–1285, 2002.
18. P.J. Grother, G.T. Candela, and J.L. Blue. Fast implementation of nearest neighbor classifiers. *Pattern Recognition*, 30:459–465, 1997.
19. X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems 18*, 2005.
20. E. Marchiori. Hit miss networks with applications to instance selection. *Journal of Machine Learning Research*, 9:997–1017, 2008.
21. E. Pekalska, R.P. W. Duin, and P. Paclík. Prototype selection for dissimilarity-based classifiers. *Pattern Recognition*, 39(2):189–208, 2006.
22. G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, 2001.
23. J.S. Sánchez, F. Pla, and F.J. Ferri. Prototype selection for the nearest neighbour rule through proximity graphs. *Pattern Recognition Letters*, 18:507–513, 1997.
24. M. Sebban, R. Nock, and S. Lallich. Boosting neighborhood-based classifiers. In *ICML*, pages 505–512, 2001.
25. M. Sebban, R. Nock, and S. Lallich. Stopping criterion for boosting-based data reduction techniques: from binary to multiclass problem. *Journal of Machine Learning Research*, 3:863–885, 2002.
26. H. Shin and S. Cho. Neighborhood property based pattern selection for support vector machines. *Neural Computation*, (19):816–855, 2007.
27. G.T. Toussaint. Proximity graphs for nearest neighbor decision rules: recent progress. In *Interface-2002, 34th Symposium on Computing and Statistics*, pages 83–106, 2002.
28. A. Vezhnevets and O. Barinova. Avoiding boosting overfitting by removing "confusing" samples. In *Proceedings of the 18th European Conference on Machine Learning (ECML)*, volume 4701, pages 430–441. LNCS, 2007.
29. U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.
30. K.Q. Weinberger and L.K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.
31. D. L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man and Cybernetics*, (2):408–420, 1972.
32. D. Randall Wilson and Tony R. Martinez. Instance pruning techniques. In *Proc. 14th International Conference on Machine Learning*, pages 403–411. Morgan Kaufmann, 1997.
33. D.R. Wilson and T.R. Martinez. Reduction techniques for instance-based learning algorithms. *Machine Learning*, 38(3):257–286, 2000.
34. Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 1151–1157. ACM Press, 2007.
35. D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 1036–1043. ACM Press, 2005.
36. D. Zhou and B. Schölkopf. Regularization on discrete spaces. In *the 27th DAGM Symposium*, pages 361–368, Berlin, Germany, 08 2005. Springer.

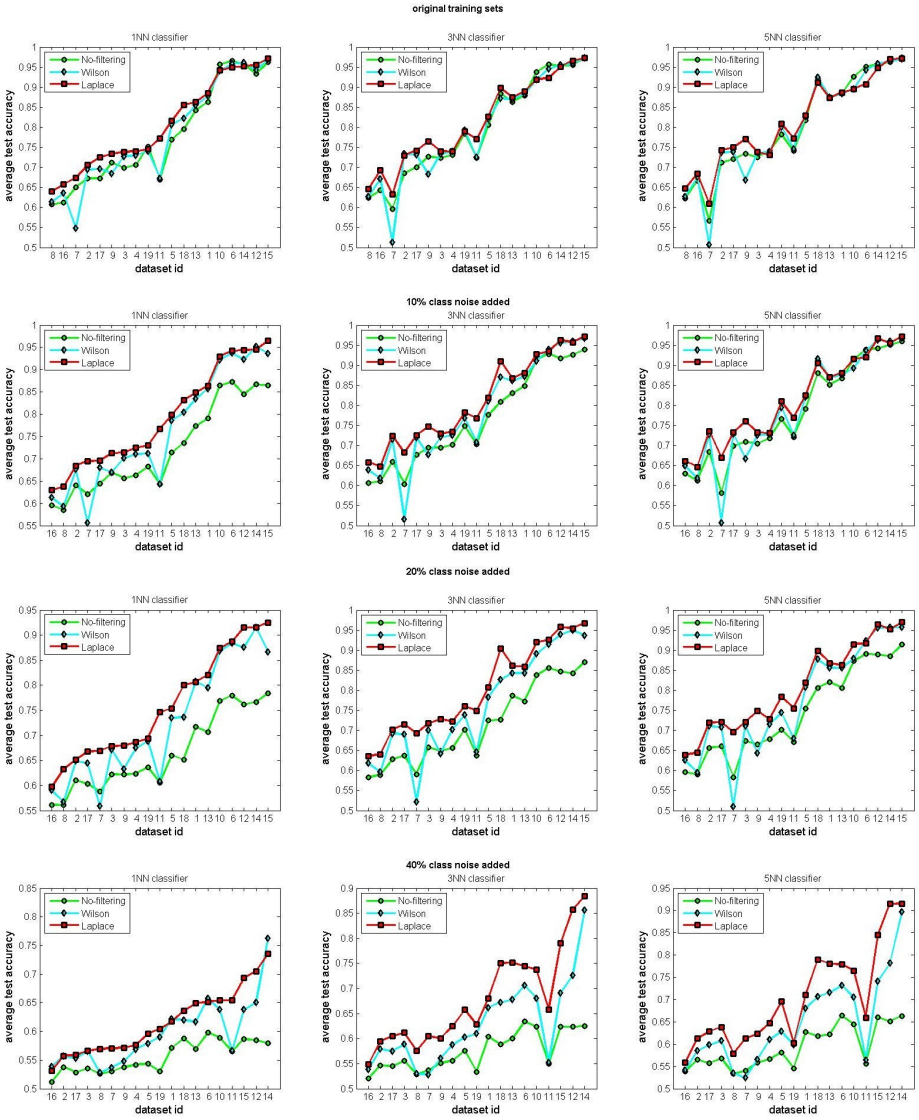

Fig. 3. Average test accuracy performance of the methods.