Effect of label noise in the complexity of classification problems

http://www.producao.usp.br/handle/BDPI/50947

Downloaded from: Biblioteca Digital da Produção Intelectual - BDPI, Universidade de São Paulo
Effect of label noise in the complexity of classification problems

Luís P.F. Garcia a,*, André C.P.L.F. de Carvalho a, Ana C. Lorena b

a Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, Trabalhador São-carlense Av. 400, São Carlos, São Paulo 13560-970, Brazil
b Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo, Tolim St. 330, São José dos Campos, São Paulo 12231-280, Brazil

A R T I C L E   I N F O

Article history:
Received 11 March 2014
Received in revised form 8 October 2014
Accepted 11 October 2014
Available online 11 February 2015

Keywords:
Classification
Label noise
Complexity measures
Noise Filter

A B S T R A C T

Noisy data are common in real-world problems and may have several causes, like inaccuracies, distortions or contamination during data collection, storage and/or transmission. The presence of noise in data can affect the complexity of classification problems, making the discrimination of objects from different classes more difficult, and requiring more complex decision boundaries for data separation. In this paper, we investigate how noise affects the complexity of classification problems, by monitoring the sensitivity of several indices of data complexity in the presence of different label noise levels. To characterize the complexity of a classification dataset, we use geometric, statistical and structural measures extracted from data. The experimental results show that some measures are more sensitive than others to the addition of noise in a dataset. These measures can be used in the development of new preprocessing techniques for noise identification and novel label noise tolerant algorithms. We thereby show preliminary results on a new filter for noise identification, which is based on two of the complexity measures which were more sensitive to the presence of label noise.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Noisy data can be regarded as objects that present inconsistencies in their predictive and/or target attribute values [1]. These inconsistencies can be either errors, absent information or unknown values [2]. Whereas noise needs to be identified and treated, secure data in a dataset must be preserved [3]. The term secure data usually refers to instances that are core of the knowledge necessary to build accurate learning models.

There are various strategies and techniques in the literature to handle with noisy data [4-8,9]. Generally, the identified noisy data are filtered and removed from the datasets. Nonetheless, it is usually hard to determine if a given instance is indeed noisy or not. Some recent proposals include designing classification techniques more tolerant and robust to noise, as surveyed in [10]. Despite the strategy employed to deal with noisy data, either by data cleansing or by the design of noise-tolerant learning algorithms, it is important to understand the effect of noise in the complexity of classification problems. This analysis can support the development of new techniques for dealing with noisy data more effectively.

This paper experimentally investigates the effect of distinct levels of label noise in the complexity of classification problems. For such, we employ a series of statistic and geometric descriptors described in [11]. These indices account for the difficulty of a classification problem by analyzing some characteristics of the dataset and the predictive performance of some simple classification models induced using the dataset. The descriptors are grouped into three main categories: measures of overlapping between feature values, measures of class separability and measures of data geometry, topology and density. We also employ some structural indices, captured by representing the dataset through a graph structure [12]. These measures extract topological and structural properties from the graphs built [13,14].

Throughout the experiments, we identify those indices that are more sensitive to the presence of label noise and can thereby be used to support noise identification. Afterwards, we preliminarily propose a simple preprocessing technique for label noise identification, which uses two of the measures that were more sensitive to the presence of label noise. Nonetheless, we emphasize that the studies carried out can support the development of other data cleaning techniques and of novel noise-tolerant learning algorithms too.

The main contributions from this study can be summarized as:

- Show that the presence of label noise at different levels influences the complexity of a classification problem. This is performed by monitoring a group of measures able to characterize the complexity of a classification problem from different perspectives;
- Analyze a new set of features which characterizes the complexity of the problem by modeling a classification dataset through...
a graph structure. These measures consider distinct topological properties of the graph built from the underlying classification dataset;

- Highlight the measures that are most sensitive to label noise imputation and use some of them to propose a new preprocessing technique able to identify label noise in a dataset.

This paper is structured as follows. Section 2 defines label noise. Section 3 describes the measures employed in this paper to characterize the complexity of the noisy classification problems, while Section 4 presents the experimental methodology followed to evaluate the sensitivity of the complexity measures to label noise imputation. Section 5 presents and discusses the results achieved in this analysis. Section 6 describes a new filter for noise identification, based on a subset of measures previously investigated and presents experimental results related to the technique performance. Finally, Section 7 concludes this paper.

2. Label noise

In supervised learning, a learning algorithm is applied to a dataset containing \( n \) pairs \( (x_i, y_i) \), where each \( x_i \) is a data point described by \( m \) predictive features and \( y_i \) corresponds to the expected label of \( x_i \). In classification problems, this label corresponds to a class or a category. The learning algorithm then induces a classification model, which should be able to predict the label of new data points. The presence of noise in a dataset affects its quality and may impair the predictive performance of the classifiers induced.

When dealing with supervised classification problems, noise can be found in [15]: (a) predictive features and (b) target attribute. Noise in predictive features is introduced in one or more predictive attributes as a consequence of incorrect, absent or unknown values. On the other hand, noise in target attributes occurs in the class labels. They can be caused by errors or subjectivity in data labeling, as well as by the use of inadequate information in the labeling process. Lately, noise in predictive features can lead to a wrong labeling of the data points, since they can be moved to the wrong side of the decision frontier.

The majority of the existent Machine Learning (ML) algorithms for solving classification problems try to minimize a cost function based on misclassification, assuming the target label values in the dataset as correct. Due to the importance of the class information in supervised learning, in this paper, we deal with noise in the target attribute only.

Learning in noisy environments has been largely investigated in recent decades [10]. Some authors prefer to deal with the problem while the classification model is induced, providing more robustness to the induced models. The pruning process in Decision Tree induction algorithms (DT) is an early initiative to increase their robustness to noisy data [16]. Nonetheless, if the noise level is high, the definition of the pruning degree can be challenging and can ultimately remove branches that are based on safe information too. Another example is the use of slack variables in Support Vector Machines training [17], which allow some examples to be misclassified or to lie within the margins of separation between the classes. This introduces an additional parameter to be tuned during the SVM training: the regularization constant. This constant accounts for the amount of training examples that can be misclassified or be placed near the decision boundary induced.

Recent work addresses noise-tolerant classifiers, where a label noise model is learnt jointly to the classification model itself [18]. For such, typically, some information must be available about the label noise or its effects [10]. The learning algorithm can also be modified to embed data cleansing [19]. Other authors prefer to treat noise previously, in a preprocessing step. Filters are developed for such, which scan the dataset for unreliable data [7,8,3]. The preprocessed dataset can then be used as input to any classification algorithm.

One of the contributions of this paper is experimentally showing how label noise affects the complexity of various classification datasets. Based on these results, we also present a new noise filtering technique. This evidences the importance of understanding the influence of label noise in classification problems. Furthermore, these results may contribute to the proposal of other filters and of novel noise tolerant algorithms.

3. Complexity indices for describing data

Each noise-tolerant technique and cleansing filter has a distinct bias when dealing with noise. To better understand their particularities, it is important to know how noisy data affects a classification problem. According to [20], noisy data tends to increase the complexity of the classification problem. Therefore, the identification and removal of noise can simplify the geometry of the separation border between the problem classes [21].

Singh [22] recommends a technique that estimates the complexity of the classification problem using neighborhood information for the identification of outliers. Sáez et al. [23] use measures able to characterize the complexity of the classification problem to predict when a noise filter can be effectively applied to a dataset. Smith et al. [9] propose a measure to capture instance hardness, considering an instance as hard if it is misclassified by a diverse set of classification algorithms. The instance hardness measure proposed is afterwards included into the learning process in two ways. They first propose a modification of the error function minimized during neural networks training, so that hard instances have a lower weight on the error function update. The second proposal is a filter that removes hard instances, which correspond to potential noisy data. In [24], we used data complexity measures as input to classifiers which were able to successfully predict whether a given dataset contains noise or not. All previous work confirm the effect of noise in the complexity of the classification problem.

In this paper we evaluate deeply the effects of different noise levels in the complexity of the classification problems, by extracting different measures from the datasets and monitoring their sensitivity to noise imputation. According to Ho and Basu [11], the difficulty of a classification problem can be attributed to three main aspects: the ambiguity among the classes, the complexity of the separation between the classes, and the data sparsity and dimensionality. Usually, there is a combination of these aspects. They propose a set of geometrical and statistical descriptors able to characterize the complexity of the classification problem associated with a dataset. Originally proposed for binary classification problems [11], some of these measures were later extended to multiclass classification in [25,26]. For measures only suitable for binary classification problems, we first transform the multiclass problem into a set of binary classification subproblems by using the one-vs-all approach. The mean of the complexity values obtained in such subproblems is then used as an overall measure for the multiclass dataset.

The descriptors of Ho and Basu [11] can be divided into three categories:

**Measures of overlapping in the feature values**: Assess the separability of the classes in a dataset. The discriminant power of each feature reflects its ambiguity level compared to the other features.

**Measures of class separability**: Quantify the complexity of the decision boundaries separating the classes. They are
usually based on linearity and on the distance between examples.

**Measures of geometry and topology**: They extract features from the local (geometry) and global (topology) structure of the data to describe the separation between classes and data distribution.

Additionally, we characterize the classification dataset as a graph and extract some structural measures from it. Modeling a classification dataset through a graph allows capturing additional topological and structural information from a dataset. In fact, graphs are powerful tools for representing the information of relations between data [27]. Therefore, we include an additional class of complexity measures in the experiments:

**Measures of structural representation**: They are measures extracted from a structural representation of the dataset using graphs, which are built taking into account the relationship among the examples.

The recent work of Smith et al. [9] also proposes a new set of measures, which are intended to understand why some instances are hard to classify. Since this type of analysis is not within the scope of this paper, these measures were not included in our experiments.

### 3.1. Measures of overlapping in feature values

**Fisher’s discriminant ratio** \((F1)\): Selects the feature that best discriminates the classes. It can be calculated by Eq. (1), for binary classification problems, and by Eq. (2) for problems with more than two classes \((C\) classes). In these equations, \(m\) is the number of input features and \(f_i\) is the \(i\)-th predictive feature:

\[
F1 = \max_{i \in 1} \frac{(\mu_{ci}^i - \mu_{cj}^i)^2}{(\sigma_{ci}^i)^2 + (\sigma_{cj}^i)^2}
\]

\[
F1 = \max_{i \in 1} \frac{\sum_{c \in C} \sum_{x \in y} p_{ci}(\mu_{ci}^i - \mu_{cj}^i)^2}{\sum_{c \in C} \sum_{x \in y} p_{cj}^2}
\]

For continuous features, \(\mu_{ci}^i\) and \((\sigma_{ci}^i)^2\) are, respectively, the average and standard deviation of the feature \(f_i\) within the class \(c_i\). Nominal features are first mapped into numerical values and \(\mu_{ci}^i\) is their median value, while \((\sigma_{ci}^i)^2\) is the variance of a binomial distribution, as presented in Eq. (3), where \(p_{ci}^i\) is the median frequency and \(n_{ci}\) is the number of examples in the class \(c_i\).

\[
\sigma_{ci}^i = \sqrt{p_{ci}^i (1 - p_{ci}^i) n_{ci}}
\]

High values of \(F1\) indicate that at least one of the features in the dataset is able to linearly separate data from different classes. Low values, on the other hand, do not indicate that the problem is non-linear, but that there is not an hyperplane orthogonal to one of the data axis that separates the classes.

**Overlapping of the per-class bounding boxes** \((F2)\): This measure calculates the volume of the overlapping region on the feature values for a pair of classes. This overlapping considers the minimum and maximum values of each feature per class in the dataset. A product of the calculated values for each feature is generated. Eq. (4) illustrates \(F2\), where \(f_i\) is the feature \(i\) and \(c_1\) and \(c_2\) are two classes.

\[
F2 = \prod_{i=1}^{n} \frac{\min(\max(f_i, c_1), \max(f_i, c_2)) - \max(\min(f_i, c_1), \min(f_i, c_2))}{\max(\min(f_i, c_1), \min(f_i, c_2)) - \min(\max(f_i, c_1), \min(f_i, c_2))}
\]

In multiclass problems, the final result is the sum of the values calculated for the underlying binary subproblems. A low \(F2\) value indicates that the features can discriminate the examples of distinct classes and have low overlapping.

**Maximum individual feature efficiency** \((F3)\): Evaluates the individual efficacy of each feature by considering how much each feature contributes to the classes separation. This measure uses examples that are not in overlapping ranges and outputs an efficiency ratio of linear separability. Eq. (5) shows how \(F3\) is calculated, where \(n\) is the number of examples in the training set and overlap is a function that returns the number of overlapping examples between two classes. High values of \(F3\) indicate the presence of features whose values do not overlap between classes.

\[
F3 = \max_{i=1}^{n} \frac{n - \text{overlap}(x_{f_i}^c_1, x_{f_i}^c_2)}{n}
\]

### 3.2. Measures of class separability

**Distance of erroneous instances to a linear classifier** \((L1)\): This measure quantifies the linearity of data, since the classification of linear separable data is considered a simpler classification task. \(L1\) computes the sum of the distances of erroneous data to a hyperplane separating two classes. Support Vector Machines (SVM) with a linear kernel [17] are used to induce the hyperplane. This measure is used only for binary classification problems. Values equal to 0 indicate a linearly separable problem.

**Training error of a linear classifier** \((L2)\): Measures the predictive performance of a linear classifier for the training data. It also uses a SVM with linear kernel. Lower training error indicate the linearity of the problem.

**Fraction of points lying on the class boundary** \((N1)\): Estimates the complexity of the correct hypothesis underlying the data. Initially, a Minimum Spanning Tree (MST) is generated from the data, connecting the data points by their distance. The fraction of points from different classes that are connected in the MST is returned. High values of \(N1\) indicate the need for more complex boundaries for separating the data.

**Average intra/inter class nearest neighbor distances** \((N2)\): The mean intra-class and inter-class distances use the \(k\)-nearest neighbor \((k\)-NN\) algorithm to analyse the spread of the examples from distinct classes. The intra-class distance considers the distance from each example to its nearest example in the same class, while the inter-class distance computes the distance of this example to its nearest example in other class. Eq. (6) illustrates \(N2\).

\[
N2 = \frac{\sum_{i=1}^{n} \text{intra}(x_i^c)}{\sum_{i=1}^{n} \text{inter}(x_i^c)}
\]

Low \(N2\) values indicate that examples of the same class are next to each other, while far from the examples of the other classes.

**Leave-one-out error rate of the 1-nearest neighbor algorithm** \((N3)\): Evaluates how distinct the examples are to different classes by considering the error rate of the 1-nearest
neighbor (1-NN) classifier, with the leave-one-out strategy. Low values indicate a high separation of the classes.

3.3. Measures of geometry and topology

**Nonlinearity of a linear classifier (L3):** Creates a new dataset by the interpolation of training data. New examples are created by linear interpolation with random coefficients of points chosen from a same class. Next, a SVM classifier with linear kernel is induced and its error rate for the original data is recorded. It is sensitive to the spread and overlapping of the data points and used for binary classification problems only. Low values indicate a high linearity.

**Nonlinearity of the one-nearest neighbor classifier (N4):** Has the same reasoning of L3, but using the 1-NN classifier instead of SVM.

**Fraction of maximum covering spheres on data (T1):** Builds hyperspheres centered on the data points. The radius of these hyperspheres is increased until touching any example of different classes. Smaller hyperspheres inside larger ones are eliminated. It outputs the ratio of the number of hyperspheres formed to the total number of data points. Low values indicate a low number of hyperspheres due to a low complexity of the data representation.

There are other measures presented in [11,26] that were not employed in this work because, by definition, they do not vary when the label noise level is increased. One of them is the dimensionality of the dataset and another is the ratio of the number of features to the number of data points (data sparsity). Similarly, measures like the directional-vector Fisher’s discriminant ratio (F1v) and collective feature efficiency (F4) from [26] were disregarded, since they have a concept similar to other measures already employed.

3.4. Measures of structural representation

Before using these measures, it is necessary to transform the classification dataset into a graph. This graph must preserve the similarities and distances between examples, so that the data relationships are captured. Each data point will correspond to a node or vertex of the graph. Edges are added connecting all pairs of nodes or some of the pairs.

Several techniques can be used to build a graph for a dataset. The most common are the $k$-nearest neighbor ($k$-NN) and the e-NN [28]. While $k$-NN connects a pair of vertices $i$ and $j$ whenever $i$ is one of the $k$-nearest neighbors of $j$, e-NN connects a pair of nodes $i$ and $j$ only if $d(i,j) < \epsilon$, where $d$ is a distance function. We employed the e-NN variant, since many edge and degree-based measures would be fixed for $k$-NN, despite the level of noise inserted in a dataset. Afterwards, all edges between examples from different classes are pruned from the graph [28]. This is a postprocessing step that can be employed for labeled datasets, which takes into account the class information.

There are various measures able to characterize the topological and structural properties of a graph. Some of them come from the statistical characterization of complex networks [14]. We used some of these graph-based measures in this paper, which are referred by their original nomenclature, as follows:

**Number of edges (Edges):** Total number of edges contained in the graph. High values for edge-related measures indicate that many of the vertices are connected and, therefore, that there are many regions of high densities from a same class. This is true because of the postprocessing of edges connecting examples from different classes applied in this work. Thus, the dataset is regarded as having low complexity if it shows a high number of edges.

**Average degree of the network (Degree):** The degree of a vertex $i$ is the number of edges connected to $i$. The average degree of a network is the average degree of all vertices in the graph. For undirected networks, it can be computed by Eq. (7), where $v_i$ is equal to 1 if $i$ and $j$ are connected, and 0 otherwise:

$$\text{degree} = \frac{1}{|v_i|} \sum_{j} v_{ij}$$

(7)

The same reasoning of edge-related measures applies to degree based measures, since the degree of a vertex corresponds to the number of edges incident to it. Therefore, high values for the degree indicates the presence of many regions of high densities from a same class, and the dataset can be regarded as having low complexity.

**Average density of network (Density):** The density of a graph is the fraction of the number of edges it contains by the number of possible edges that could be formed. The average density also allows capturing whether there are dense regions from the same class in the dataset. High values indicate the presence of such regions and a simpler dataset.

**Maximum number of components (MaxComp):** Corresponds to the maximal number of connected components of a graph. In an undirected graph, a component is a subgraph with paths between all of its nodes. When a dataset shows a high overlapping between classes, the graph will probably present a large number of disconnected components, since connections between different classes are pruned from the graph. The minimal component will tend to be smaller in this case. Thus, we will assume that smaller values of the MaxComp measures represent more complex datasets.

**Closeness centrality (Closeness):** Average number of steps required to access every other vertex from a given vertex, which is the number of edges traversed in the shortest path between them. It can be computed by the inverse of the distance between the nodes, as shown in the following equation:

$$\text{closeness} = \frac{1}{\sum_{j \neq i} d(v_i)}$$

(8)

Since the closeness measure uses the inverse of the shortest distance between vertices, larger values are expected for simpler datasets that will show low distances between examples from the same class.

**Betweenness centrality (Betweenness):** The vertex and edge betweenness are defined by the average number of shortest paths that traverses them. We employed the vertex variant. Eq. (9) represents the betweenness value of a vertex $v_i$, where $d(v_i)$ is the total number of the shortest paths from node $i$ to node $l$ and $d_j(v_i)$ is the number of those paths that pass through $j$:

$$\text{betweenness}(v_i) = \sum_{i \neq j \neq l} \frac{d_j(v_i)}{d(v_i)}$$

(9)

The indice for Betweenness will be small for simpler datasets, since the distance between the shortest paths and the paths which pass through $j$ will be close.

**Clustering Coefficient (CluCoef):** Measures the probability that adjacent vertices of a graph are connected. The clustering coefficient of a vertex $v_i$ is given by the ratio of the
number of edges between its neighbors and the maximum number of edges that could possibly exist between these neighbors. Measure ClsCoef will be larger for simpler datasets, which will produce large connected components joining vertices from the same class.

Hub score (Hubs): Measures the score of each node by the number of connections it has to other nodes, weighted by the number of connections these neighbors have. That is, more connected vertices, which are also connected to highly connected vertices, have higher hub score. The hub score is expected to have a low mean for high complexity datasets, since strong vertices will become less connected to strong neighbors. For instance, hubs are expected at regions of high density from a given class. Therefore, simpler datasets with high density will show larger values for this measure.

Average Path Length (AvgPath): Average size of all shortest paths in the graph. It measures the efficiency of information spread in the network. It is illustrated by Eq. (10), where \( n \) represents the number of vertices of the graph and \( d(v_{ij}) \) is the shortest distance between vertices \( i \) and \( j \).

\[
\text{AvgPath} = \frac{1}{n(n-1)} \sum_{i 
eq j} d(v_{ij});
\]

For the AvgPath measure, high values are expected for low density graphs, indicating an increase in complexity.

For those measures that are calculated for each vertex individually, we computed an average for all vertices in the graph. The graph measures used in this paper mainly evaluate the overlapping of the classes and their density.

A previous paper also investigated the use of complex-network measures to characterize supervised datasets [12]. It used part of the measures presented here to design meta-learning models able to predict the best performing model between a pair of classifiers for a given dataset. They also compared these measures to those used in [12], but in a distinct scenario from the one adopted here. It is not clear whether they employ a postprocessing of the graph for removing edges between nodes of different classes, as done in this paper. Also, some of the measures employed in that paper are not suitable for our scenario and are not used here. One example is the number of nodes of the graph, which will not vary for a given dataset despite of its noise level. The only measures in common to those used in [12] are the number of edges, the average clustering coefficient and the average degree. Morais and Prati [12] also do not analyse the expected values of the measures for simpler or more complex datasets. Besides introducing new measures, we also describe the behavior of all measures for simpler or complex problems. Moreover, we try to identify the best suited measures for detecting the presence of label noise in a dataset.

3.5. Summary of measures

Table 1 summarizes the measures employed to characterize the complexity of the datasets used in this study. For each measure, we present upper (Maximum value) and lower bounds (Minimum value) achievable and how they are associated with the increase or decrease of complexity of the classification problems (Complexity column). For a given measure, the value in column “Complexity” is “+” if higher values of the measure are observed for high complexity datasets, that is, when the measure value correlates positively to the complexity level. On the other hand, the “−” sign denotes the opposite, so that low values of the measure are obtained for high complexity datasets, denoting a negative correlation.

Most of the bounds were obtained considering the equations directly, while some of the graph-based bounds were experimentally defined. For instance, for the F1 measure, if the means of the feature values are always equal, meaning that the classes overlap for all features (an extreme case), the nominator of Eq. (2) will be 0. Similarly, a maximum value cannot be determined for F1, as it is dependent on the feature values of each dataset. We denote that by the “+” value in the table. In the case of graph-based measures, we generated graphs representing simple and complex relations between the same number of data points and observed the measure values achieved. A simple graph would correspond to a case where the classes are well separated and there is a high number of connections between examples from the same class, while a complex dataset would correspond to a graph where examples of different classes are always next to each other and ultimately the connections between them are pruned according to our graph construction method.

<table>
<thead>
<tr>
<th>Type of measure</th>
<th>Measure</th>
<th>Minimum value</th>
<th>Maximum value</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overlapping in feature values</td>
<td>F1</td>
<td>0</td>
<td>+∞</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>F2</td>
<td>0</td>
<td>+∞</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>F3</td>
<td>0</td>
<td>1</td>
<td>−</td>
</tr>
<tr>
<td>Class separability</td>
<td>L1</td>
<td>0</td>
<td>+∞</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>L2</td>
<td>0</td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>N1</td>
<td>0</td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>N2</td>
<td>0</td>
<td>+∞</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>N3</td>
<td>0</td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td>Geometry and topology</td>
<td>L3</td>
<td>0</td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>N4</td>
<td>0</td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>L1</td>
<td>0</td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td>Structural representation</td>
<td>Edges</td>
<td>0</td>
<td>n(n−1)/2</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Degree</td>
<td>0</td>
<td>n−1</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>MaxComp</td>
<td>1</td>
<td>n</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Closeness</td>
<td>0</td>
<td>1/(n−1)</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Betweenness</td>
<td>0</td>
<td>(n−1)(n−2)/2</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>Hubs</td>
<td>0</td>
<td>1</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Density</td>
<td>0</td>
<td>1</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ClsCoef</td>
<td>0</td>
<td>1</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AvgPath</td>
<td>1/n(n−1)</td>
<td>0.5</td>
<td>+</td>
</tr>
</tbody>
</table>
4. Experiments

This section presents the experiments performed to evaluate how the different data complexity measures from Section 3 behave in the presence of label noise for several benchmark public datasets. First, a set of classification benchmark datasets were chosen for the experiments. Different levels of label noise were afterwards added to each dataset. We then monitor how the complexity level of the datasets are affected by noise imputation. This is accomplished by:

1. Verifying the Spearman correlation between the measures values with the noise rates artificially imputed. This analysis allows identifying a set of measures more sensitive to the presence of noise in a dataset.
2. Evaluating the correlation between the measures values to identify those measures that (1) capture different concepts regarding noisy environments and (2) can be jointly used to support the development of new noise-handling techniques.

Next, we detail the experimental protocol previously outlined.

4.1. Datasets

For the experiments, we selected artificial and real datasets. The artificial datasets were introduced and generously provided by Amancio et al. [29]. These authors generated artificial classification datasets based on multivariate Gaussians, with different levels of overlapping between the classes. For this paper we selected 180 balanced datasets (with the same number of examples per class) with 2 classes, containing 2, 10 and 50 features and with different overlapping rates for each of the number of features. We chose such datasets based on the observations of a recent work [9], which points out that class overlap seems to be a principal contributor to instance hardness and that noisy data can ultimately be considered hard instances.

For the real datasets, we selected 53 benchmarks from UCI and Keel repositories [30,31] for our experiments. As most of them are real datasets, it is not possible to assert that they are noiseless, although some of them are artificial and show no label inconsistency. Nonetheless, a recent study showed that most of the datasets from UCI can be regarded as easy problems, once many classification techniques are able to attain high accuracies when applied to them [32]. Table 2 summarizes the main characteristics of these datasets: number of examples (Examples), number of features (Features), number of classes (Class) and percentage of the examples in the majority class (%MC).

4.2. Noise imputation

In order to corrupt the datasets, we used the most common type of artificial noise imputation method for classification problems: uniform random addition [15]. In this approach, each example has the same probability of having its label exchanged by another random label [33]. For each dataset, the noise was inserted at different levels, namely 5%, 10%, 20% and 40%. Thus, we were able to investigate the influence of increasing noise levels.

Once the selection of examples was random, we generated 10 different noisy versions of each dataset for each noise level. Since most of the original benchmark datasets may already contain some noise level, in this case we have as result a potential noise level of 5%, 10%, 20% and 40%.

4.3. Methodology

First, we created noisy versions of the original datasets from Section 4.1 by using the previously described systematic model of noise imputation. Despite all datasets being partitioned by 10-fold-cross-validation, noise was inserted in the training folds only. The complexity measures were extracted from the original training datasets and from their noisy versions.

To calculate the complexity measures described from Section 3.1 to Section 3.3, we used the Data Complexity Library (DCoL) [26]. All

Table 2
Summary of datasets characteristics: name, number of examples, number of features, number of classes and the percentage of the majority class of each dataset.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Examples</th>
<th>Features</th>
<th>Class</th>
<th>%MC</th>
<th>Dataset</th>
<th>Examples</th>
<th>Features</th>
<th>Class</th>
<th>%MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>4153</td>
<td>9</td>
<td>19</td>
<td>16</td>
<td>molecular-promoters</td>
<td>106</td>
<td>58</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>acute-inflammations</td>
<td>120</td>
<td>8</td>
<td>2</td>
<td>58</td>
<td>monk1</td>
<td>556</td>
<td>7</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>appendicitis</td>
<td>106</td>
<td>8</td>
<td>2</td>
<td>80</td>
<td>monk2</td>
<td>601</td>
<td>7</td>
<td>2</td>
<td>65</td>
</tr>
<tr>
<td>australian</td>
<td>690</td>
<td>15</td>
<td>2</td>
<td>55</td>
<td>movement-libras</td>
<td>360</td>
<td>91</td>
<td>15</td>
<td>06</td>
</tr>
<tr>
<td>backache</td>
<td>180</td>
<td>32</td>
<td>2</td>
<td>86</td>
<td>newthyroid</td>
<td>215</td>
<td>6</td>
<td>3</td>
<td>69</td>
</tr>
<tr>
<td>balance</td>
<td>625</td>
<td>5</td>
<td>3</td>
<td>46</td>
<td>page-blocks</td>
<td>5473</td>
<td>11</td>
<td>5</td>
<td>89</td>
</tr>
<tr>
<td>banana</td>
<td>5300</td>
<td>3</td>
<td>2</td>
<td>55</td>
<td>parkinsons</td>
<td>195</td>
<td>23</td>
<td>2</td>
<td>75</td>
</tr>
<tr>
<td>blood-transfusion-service</td>
<td>748</td>
<td>5</td>
<td>2</td>
<td>76</td>
<td>phoneme</td>
<td>5404</td>
<td>6</td>
<td>2</td>
<td>70</td>
</tr>
<tr>
<td>breast-tissue</td>
<td>106</td>
<td>10</td>
<td>6</td>
<td>20</td>
<td>pima</td>
<td>768</td>
<td>9</td>
<td>2</td>
<td>65</td>
</tr>
<tr>
<td>bupa</td>
<td>345</td>
<td>7</td>
<td>2</td>
<td>57</td>
<td>ringnorm</td>
<td>7400</td>
<td>21</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>car</td>
<td>1728</td>
<td>7</td>
<td>4</td>
<td>70</td>
<td>saheart</td>
<td>462</td>
<td>10</td>
<td>2</td>
<td>65</td>
</tr>
<tr>
<td>cardiootography</td>
<td>2126</td>
<td>21</td>
<td>10</td>
<td>27</td>
<td>segmentation</td>
<td>2310</td>
<td>19</td>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>cnc</td>
<td>1473</td>
<td>10</td>
<td>3</td>
<td>42</td>
<td>spectf</td>
<td>349</td>
<td>45</td>
<td>2</td>
<td>72</td>
</tr>
<tr>
<td>collins</td>
<td>485</td>
<td>22</td>
<td>13</td>
<td>16</td>
<td>statlog-german</td>
<td>1000</td>
<td>21</td>
<td>2</td>
<td>70</td>
</tr>
<tr>
<td>connectionist-mines-vs-rocks</td>
<td>208</td>
<td>61</td>
<td>2</td>
<td>53</td>
<td>statlog-heart</td>
<td>270</td>
<td>14</td>
<td>2</td>
<td>55</td>
</tr>
<tr>
<td>crabs</td>
<td>200</td>
<td>6</td>
<td>2</td>
<td>50</td>
<td>tae</td>
<td>151</td>
<td>6</td>
<td>3</td>
<td>34</td>
</tr>
<tr>
<td>expgen</td>
<td>207</td>
<td>80</td>
<td>5</td>
<td>57</td>
<td>tic-tac-toe</td>
<td>958</td>
<td>10</td>
<td>2</td>
<td>65</td>
</tr>
<tr>
<td>flags</td>
<td>178</td>
<td>29</td>
<td>5</td>
<td>33</td>
<td>titanic</td>
<td>2201</td>
<td>4</td>
<td>2</td>
<td>67</td>
</tr>
<tr>
<td>flare</td>
<td>1066</td>
<td>12</td>
<td>6</td>
<td>31</td>
<td>vehicle</td>
<td>846</td>
<td>19</td>
<td>4</td>
<td>25</td>
</tr>
<tr>
<td>glass</td>
<td>205</td>
<td>10</td>
<td>5</td>
<td>37</td>
<td>vowel</td>
<td>990</td>
<td>11</td>
<td>11</td>
<td>09</td>
</tr>
<tr>
<td>habermans-survival</td>
<td>306</td>
<td>4</td>
<td>2</td>
<td>73</td>
<td>waveform-5000</td>
<td>5000</td>
<td>41</td>
<td>3</td>
<td>33</td>
</tr>
<tr>
<td>hayes-roth</td>
<td>160</td>
<td>5</td>
<td>3</td>
<td>40</td>
<td>wdbc</td>
<td>569</td>
<td>31</td>
<td>2</td>
<td>62</td>
</tr>
<tr>
<td>ionosphere</td>
<td>351</td>
<td>34</td>
<td>2</td>
<td>64</td>
<td>wine</td>
<td>178</td>
<td>14</td>
<td>3</td>
<td>39</td>
</tr>
<tr>
<td>iris</td>
<td>150</td>
<td>5</td>
<td>3</td>
<td>33</td>
<td>wine-quality</td>
<td>6497</td>
<td>12</td>
<td>6</td>
<td>43</td>
</tr>
<tr>
<td>kr-vs-kp</td>
<td>3196</td>
<td>37</td>
<td>2</td>
<td>52</td>
<td>yeast</td>
<td>1484</td>
<td>9</td>
<td>9</td>
<td>31</td>
</tr>
<tr>
<td>led7digit</td>
<td>500</td>
<td>8</td>
<td>10</td>
<td>11</td>
<td>zoo</td>
<td>84</td>
<td>17</td>
<td>4</td>
<td>48</td>
</tr>
</tbody>
</table>
distance-based measures employed the normalized euclidean distance for continuous features and the overlap distance for nominal features (this distance is 0 for equal categorical values and 1 otherwise) [34]. To build the graph for the graph-based measures, we used the $\epsilon$-NN algorithm, with the $\epsilon$ threshold value equal to 15%, like in [12]. The measures described in Section 3.4 were calculated using the Igraph library [35].

As a result, we have one meta-level dataset that will be employed in the subsequent experiments. It contains 20 meta-features ( complexity and graph-based measures) describing the benchmark datasets and their noisy versions. This meta-level dataset has therefore 2173 examples: 53 ( original datasets) + 53 ( datasets) + 4 ( noise levels) * 10 ( random versions).

Two types of analysis were performed using the meta-level dataset:

1. Correlation between the measure values and the noise level of the datasets;
2. Correlation within the measure values.

The first analysis will consider all measures. Its results will then refine a subset of measures more sensitive to noise imputation, which will be further analysed in the second correlation study.

The first analysis verifies if there is a direct relation between the noise level of a dataset and the values of the measures extracted from it. This allows the identification of the measures that are more sensitive to the presence of noise. For such, the Spearman’s rank correlation between the measure values and the different noise levels was calculated for all datasets. Those measures that present a significant correlation according to the Spearman’s statistical test (at 95% of confidence value) were selected to be analyzed further.

The second analysis evaluates the Spearman correlation between the measures with the highest sensitivity to the presence of noise according to the previous experimental results. It looks for overlapping in the complexity concepts extracted by these measures. Similar analyses are carried out in [9] for accessing the relationship between some instance hardness measures proposed by them. While a high correlation could indicate that the measures are capturing the same complexity concepts, a low correlation indicates that the measures could complement each other, an issue that can be further explored.

5. Results of correlation analysis

This section presents the experimental results for the correlation analysis previously described. We also have evaluated the results for
some artificial datasets as described in Section 4.1. These results were quite similar to those observed for the benchmark datasets, with the difference that the absolute correlation values calculated were higher for the artificial datasets. The complete set of results, including those obtained for the artificial datasets and the datasets employed, can be consulted at http://lpfgarcia.github.io/corr/.

Fig. 1 presents histograms of the values of the complexity measures for all benchmark datasets when random noise is added. The bars are colored according to the amount of noise inserted, from 0% (original datasets) to 40%. The measure values were normalized considering all datasets to allow their direct comparison. It is possible to notice that some of the measures are more sensitive to noise imputation and present clear limits on their values for different noise levels. They are: N1, N3, Edges, Degree and Density. On the other hand, other measures like Betweenness do not present a clear contrast in their values for different noise levels.

Furthermore, it is also possible to notice from Fig. 1 that, as more noise is added to the datasets, the complexity of the classification problem tends to increase. This is reflected in the values of the majority of the complexity measures, that either increased or decreased when noise is added, in accordance to their positive or negative correlation to the complexity level, as shown in Table 1 (column “Complexity”). For instance, higher N1 values are expected for more complex datasets and the N1 values indeed increased for higher levels of noise. On the other hand, lower F1 values are expected for more complex datasets and we can observe that as more noise is added to the datasets, the F1 values tend to reduce.

5.1. Correlation of measures to the noise level

Fig. 2 shows the correlation between the values of the measures for the different noise levels in the datasets. Positive and negative values are plotted in order to show clearly which measures are directly or indirectly correlated to the noise levels. It is noticeable that, as the noise level increases, the values of the complexity measures either increase or reduce accordingly, indicating increases in the complexity level of the noisy datasets. The closer to 1 or −1, the higher is the relation between the measure and the noise level.

According to the statistical test employed, 18 measures presented significant correlation to the noise levels, at 95% of confidence. Among the measures with direct correlation to the noise level, nine are basic complexity measures from the literature (N3, N1, N2, N4, L2, L1, T1, F2, and L3). These measures mainly capture: classes separability (N3, N1, N2, L2 and L1), data topology according to a nearest neighbor classifier (N4, T1 and L3) and individual feature overlapping (F2). Regarding those measures indirectly related to the noise levels, two are basic complexity measures based on feature overlapping (F1 and F3), while six are based on structural representation (Density, Hub, Degree, ClsCoeff, Edges and MaxComp). Only the Closeness and Betweenness measures did not present significant correlation to the noise levels. As expected, the most prominent measures are the same that showed more distinct values for different noise levels in the histograms from Fig. 1.

Despite the statistical difference, it is possible to notice some low correlation values in Fig. 2. Only the measures N3, N1 and N2 presented correlation values higher than 0.5. These correlations were higher in the experiments with artificial datasets. This can be a result of the fact that, for real datasets, the amount of noise added is potential rather than actual.

5.2. Correlation between measures

In order to verify whether the measures capture similar or distinct information from data, we calculated pairwise correlations between their values. Only those measures considered more relevant in the previous analysis were considered. These measures were highlighted as more sensitive to noise imputation and can therefore be successfully employed for noise identification.

Fig. 3 shows a heatmap of the correlation between these pairs of measures. Each column and row corresponds to a measure. Each box is colored according to the correlation values calculated, from gray (highest correlation, despite positive or negative) to white (lowest correlation). The absolute values of all correlations are also shown inside the heatmap cells. We highlight in bold the correlation test (at 95% of confidence level). These pairs of measures correspond to those that can potentially complement each other.

According to the heatmap, various measures are weakly correlated to each other. Therefore, they capture distinct aspects from the data. As expected, the measures N1, N2, N3 and N4 from [11] are highly correlated. They are all based on nearest neighbor information. Despite the fact that all structural representation measures are extracted from a nearest neighbor graph, their correlation to N1, N2, N3 and N4 is low in several cases. Among the graph-based measures, high correlations are observed between Edges, Degree and MaxComp. Since the degree of a graph
is calculated considering the number of its edges and number of connected components, this correlation is expected by definition.

It is interesting to notice that many of the measures highlighted as distinguishing the noise levels have low correlation between them. This is particularly true for class separability measures (e.g., N3) when paired to the structural representation measures (e.g., Degree and Edges). Therefore, they could be combined to improve noise identification and handling. This issue is preliminarily investigated in the next section.

6. Proposed noise filtering technique

Based on the previous experimental results, this section proposes and investigates a new filter, named GraphNN, for label noise identification. It uses two of the measures more correlated to the increasing noise levels in the datasets: Leave-one-out error rate of the 1-nearest neighbor classifier (N3); and Average degree of the network (Degree).

The GraphNN filter identify noisy examples by first constructing a graph from the dataset, as described in Section 3.4. Afterwards, it uses the degree of each vertex for pointing an example as a potential noise. In fact, when an example is misclassified, it will be probably close to examples from another class. In this case, its edges to close examples will be pruned and the example will tend to have a low degree value. Safe examples, on the other hand, will be connected to a high number of examples from the same class and show a high degree value. Therefore, it is necessary to stipulate a threshold on the node degree so as its corresponding example can be considered as noisy or not.

When a dataset has a large amount of noise, a larger number of examples will have a low degree value and the threshold value can be higher. On the other hand, for datasets with a lower noise level, a lower threshold value can be required. Otherwise, many safe examples will be regarded as noisy. Due to the difficulty in selecting a threshold value, it would be appropriate for all of the datasets employed in our experiments to use the N3 measure value to estimate the percentage of noise in the dataset. This was the most correlated measure to the noise levels in our experiments and for which clearer limits on the values obtained for distinct noise levels can be observed. The $\epsilon$ value adopted to build the graph from data also influences on the average degree of the nodes and can influence the threshold choice, although this was not studied in our experiments.

![Fig. 3. Heatmap of correlation between measures.](image-url)
Therefore, in GraphNN we first order all examples according to their degree values. Afterwards, the N3 value delimits how much of the examples of lower degree can be regarded as noisy. Furthermore, among the examples with a degree lower than the threshold, only those that are misclassified by the NN classifier used in N3 are considered noisy. This polling allows more robustness for maintaining safe examples.

6.1. Experiments on noise filtering

We performed a set of experiments in order to analyse the performance of the GraphNN filter in identifying noisy examples. The same datasets used in our previous analysis were employed in these experiments.

Once noise was added in a controlled way, it is possible to assess which are the noisy examples and to record the performance of the filter in retrieving them. We thereby employ the F1-measure in this evaluation, as illustrated by Eq. (11) and proposed in [3]. This measure combines the precision and recall values of a filter in noise identification: precision is defined as the number of correctly identified noisy cases divided by the number of examples identified by the filter as noisy; recall is the number of correctly identified noisy cases divided by the total number of noisy examples introduced in the dataset.

\[ F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \]  

As baselines, we included some distance-based noise filtering techniques from literature [36,4]. All of them employ the k-nearest neighbor (k-NN) algorithm to determine whether an example is suspicious or not. Therefore, according to them, an example is consistent only if it is located close to examples from its class. This resembles our method, where the graph captures the similarity between examples from the same class. Otherwise, the example is either incorrectly labeled or in the decision border. In the later case, it is also considered unsafe, since small perturbations in a borderline example can move it to the wrong side of the decision border.

The Edited Nearest Neighbor (ENN) technique removes an example if the labels of its k nearest neighbors are different from its actual label [37]. Repeated ENN (RENN) is a variation of ENN, which applies ENN repeatedly until all objects have the majority of their neighbors from the same class. In All-kNN, instead of using a fixed value of k, increasing values of k are considered [4]. At the end of each iteration, examples that have the majority of their neighbors from other classes are marked as noisy. These signalized examples are then eliminated from the dataset.

6.2. Performance of the cleansing filter

The F1-measure values obtained by the filters in the retrieval of the artificially imputed noisy examples are shown in the heatmap of Fig. 4. In this figure, each column represents one filter, while each row corresponds to a dataset. The closer the color is to black, the higher is the F1-score computed. The gray color represents an intermediate F1-score and the white color represents low F1-score levels. According to the heatmap, the GraphNN filter had attained, in average, a good predictive performance for most of the datasets. For the banana, connectionist-mines-vs-rocks, crabs, kr-vs-kp, monk2, ringnorm, titanice and tic-tac-toe datasets, the performance decreased.
Fig. 5. Frequency of which each filter had best F1-score.

The number of times each filter performed better in noise identification for all noise rates and all datasets is presented at Fig. 5. The best performance was obtained by GraphNN, followed by RENN and AENN. The ENN filter had shown the worst performance.

7. Conclusion

This work investigated how label noise affects the complexity of classification tasks, by monitoring the values of simple measures extracted from datasets with increasing noise levels. Part of these measures were already used in the literature for understanding and analyzing the complexity of classification tasks. Some other measures that are based on modeling the datasets by graphs were introduced in this paper.

Experimentally, measures able to capture characteristics like separability of the classes, alterations in the class boundary and densities within the classes were the most affected ones by the introduction of label noise in the data. Therefore, they are good candidates for further exploitation and to support the design of new noise identification techniques and noise-tolerant classification algorithms. Moreover, experimental results showed a low correlation between the basic complexity measures and the graph-based measures, stressing the relevance of exploring different correlation between the basic complexity measures and the graph-based measures. Thus, the idea that data from the same class tend to be next from each other in the feature space, while far from examples from different classes, is reinforced.

Our experimental protocol and graph-based measures can also be used in other types of analysis, such as in verifying the effects of data unbalance, feature selection, feature discretization, among others. It is also possible to use other combinations of measures to devise new preprocessing filters. We also plan to employ feature selection strategies to evidence the best measures able to characterize noisy data. It would be also interesting to investigate how the graph-based measures are affected by the choice of the $c$ parameter used to build the graph. Finally, we plan to use some of the highlighted measures to develop new noise-tolerant algorithms and compare GraphNN with other up-to-date noise filters.

We created a webpage with additional information regarding the experimental results, the codes used for the experiments and the results for some artificial datasets. It can be found at http://lpfgarcia.github.io/rcorr/.

Acknowledgments

The authors would like to thank FAPESP, CNPq and CAPES for their financial support.

References


Luis P.F. Garcia received the B.Sc. degree in Computer Engineering in 2010, from University of São Paulo, Brazil. He is currently a Ph.D. candidate at the Institute of Computer Sciences and Mathematics, University of São Paulo, Brazil. His current research interests include Machine Learning, Data Mining, Meta-learning and Data Preprocessing.

André C.P.L.F. de Carvalho is a full-time Professor in the Department of Computer Science, University of São Paulo, Brazil, where he was head of Department from 2008 to 2010. He received his B.Sc. and M.Sc. degrees in Computer Science from the Universidade Federal de Pernambuco, Brazil. He received his Ph.D. degree in Electronic Engineering from the University of Kent, UK. He has published around 80 Journal and 200 Conference refereed papers. He has been involved in the organization of several conferences and journal special issues. His main interests are Machine Learning, Data Mining and Hybrid Intelligent Systems. He is in the editorial board of several journals and was a member of the Brazilian Computing Society, SBC, Council. He was the editor of the SBC/Elsevier textbook series until July 2011. Prof. de Carvalho is the Director of the Research Center for Machine Learning in Data Analysis of the Universidade de Sao Paulo, Brazil.

Ana C. Lorena has bachelor (2001) and Ph.D. (2006) degrees in Computer Science from the University of São Paulo – São Carlos. She is currently an Assistant Professor at the Federal University of São Paulo. She has experience in Computer Science, working mainly on the following topics: data mining, supervised machine learning, hybrid intelligent systems and support vector machines.