Classification in the presence of label noise: A survey
Frénay, Benoît; Verleysen, Michel

Published in:
IEEE Transactions on Neural Networks and Learning Systems

DOI:
10.1109/tnnls.2013.2292894

Publication date:
2014

Document Version
Publisher's PDF, also known as Version of record

Link to publication
Citation for published version (HARVARD):
https://doi.org/10.1109/tnnls.2013.2292894
Classification in the Presence of Label Noise: a Survey
Benoît Frénay and Michel Verleysen, Senior Member, IEEE

Abstract—Label noise is an important issue in classification, with many potential negative consequences. For example, the accuracy of predictions may decrease, whereas the complexity of inferred models and the number of necessary training samples may increase. Many works in the literature have been devoted to the study of label noise and the development of techniques to deal with label noise. However, the field lacks a comprehensive survey on the different types of label noise, their consequences and the algorithms that take label noise into account. This paper proposes to fill this gap. Firstly, the definitions and sources of label noise are considered and a taxonomy of the types of label noise is proposed. Secondly, the potential consequences of label noise are discussed. Thirdly, label noise-robust, label noise cleansing and label noise-tolerant algorithms are reviewed. For each category of approaches, a short discussion is proposed in order to help the practitioner to choose the most suitable technique in its own particular field of application. Eventually, the design of experiments is also discussed, what may interest the researchers who would like to test their own algorithms. In this survey, label noise consists of mislabelled instances: no additional information is assumed to be available, like e.g. confidence labels.

Index Terms—classification, label noise, class noise, mislabelling, robust methods, survey.

I. INTRODUCTION

Classification has been widely studied in machine learning. In that context, the standard approach consists in learning a classifier from a labelled dataset, in order to predict the class of new samples. However, real-world datasets may contain noise, which is defined in [1] as anything that obscures the relationship between the features of an instance and its class. In [2], noise is also described as consisting of non-systematic errors. Among other consequences, many works have shown that noise can adversely impact the classification performances of induced classifiers [3]. Hence, the ubiquity of noise seems to be an important issue for practical machine learning, e.g. in medical applications where most medical diagnosis tests are not 100 percent accurate and cannot be considered a gold standard [4]–[6]. Indeed, classes are not always as easy to distinguish as lived and died [4]. It is therefore necessary to implement techniques which eliminate noise or reduce its consequences. It is all the more necessary since reliably labelled data are often expensive and time consuming to obtain [4], what explains the commonness of noise [7].

In the literature, two types of noise are distinguished: feature (or attribute) noise and class noise [2], [3], [8]. On the one hand, feature noise affects the observed values of the features, e.g. by adding a small Gaussian noise to each feature during measurement. On the other hand, class noise alters the observed labels assigned to instances, e.g. by incorrectly setting a negative label on a positive instance in binary classification. In [3], [9], it is shown that class noise is potentially more harmful than feature noise, what highlights the importance of dealing with this type of noise. The prevalence of the impact of label noise is explained by the fact 1) that there are many features, whereas there is only one label and 2) that the importance of each feature for learning is different, whereas labels always have a large impact on learning. Similar results are obtained in [2]: feature noise appears to be less harmful than class noise for decision trees, except when a large number of features are polluted by feature noise.

Even if there exists a large literature about class noise, the field still lacks a comprehensive survey on the different types of label noise, their consequences and the algorithms that take label noise into account. This work proposes to cover the class noise literature. In particular, the different definitions and consequences of class noise are discussed, as well as the different families of algorithms which have been proposed to deal with class noise. As in outlier detection, many techniques rely on noise detection and removal algorithms, but it is shown that more complex methods have emerged. Existing datasets and data generation methods are also discussed, as well as experimental considerations.

In this work, class noise refers to observed labels which are incorrect. It is assumed that no other information is available, contrarily to other contexts where experts can e.g. provide a measure of confidence or uncertainty on their own labelling or answer with sets of labels. It is important to make clear that only the observed label of an instance is affected, not its true class. For this reason, class noise is called here label noise.

The survey is organised as follows. Section II discusses several definitions and sources of label noise, as well as a new taxonomy inspired by [10]. The potential consequences of label noise are depicted in Section III. Section IV distinguishes three types of approaches to deal with label noise: label noise-robust methods, label noise cleansing methods and label noise-tolerant methods. The three families of methods are discussed in Sections V, VI and VII, respectively. Section VIII discusses the design of experiments in the context of label noise and Section IX concludes the survey.
II. DEFINITION, SOURCES AND TAXONOMY OF LABEL NOISE

Label noise is a complex phenomenon, as shown in this section. First, Section II-A defines label noise and specifies the scope of the survey. Similarities and differences with outliers and anomalies are also highlighted, since outlier detection methods can be used to detect mislabelled instances. Next, Section II-B reviews various sources of label noise, including insufficient information, expert labelling errors, subjectivity of the classes and encoding and communication problems. Eventually, a taxonomy of the types of label noise is proposed in Section II-C in order to facilitate further discussions. The proposed taxonomy highlights the potentially complex relationships between the features of instances, their true class and their observed label. This complexity should be taken into account when designing algorithms to deal with label noise, as they should be adapted to the characteristics of label noise.

A. Definition of Label Noise and Scope of the Survey

Classification consists in predicting the class of new samples, using a model inferred from training data. In this survey, it is assumed that each training sample is associated with an observed label. This label often corresponds to the true class of the sample, but it may be subjected to a noise process before being presented to the learning algorithm [11]. It is therefore important to distinguish the true class of an instance from its observed label. The process which pollutes labels is called label noise and must be separated from feature (or attribute) noise [2], [3], [8] which affects the value of features. Some authors also consider outliers which are correctly labelled as label noise [12], what is not done here.

In this survey, label noise is considered to be a stochastic process, i.e. the case where the labelling errors may be intentionally (like e.g. in the food industry [13]–[16]) and maliciously induced by an adversary agent [17]–[26] is not considered. Moreover, labelling errors are assumed to be independent from each other [11]. Edmonds [27] shows that noise in general is a complex phenomenon. In some very specific contexts, stochastic label noise can be intentionally introduced e.g. to protect people privacy, in which case its characteristics are completely under control [28]. However, a fully specified model of label noise is usually not available, what explains the need for automated algorithms which are able to cope with label noise. Learning situations where label noise occurs can be called imperfectly supervised, i.e. pattern recognition applications where the assumption of label correctness does not hold for all the elements of the training sample [29]. Such situations are between supervised and unsupervised learning.

Dealing with label noise is closely related to outlier detection [30]–[33] and anomaly detection [34]–[38]. Indeed, mislabelled instances may be outliers, if their label has a low probability of occurrence in their vicinity. Similarly, such instances may also look anomalous, with respect to the class which corresponds to their incorrect label. Hence, it is natural that many techniques in the label noise literature are very close to outlier and anomaly detection techniques; this is detailed in Section VI. In fact, many of the methods which have been developed to deal with outliers and anomalies can also be used to deal with label noise (see e.g. [39], [40]). However, it must be highlighted that mislabelled instances are not necessarily outliers or anomalies, which are subjective concepts [41]. For example, if labelling errors occur in a boundary region where all classes are equiprobable, the mislabelled instances neither are rare events nor look anomalous. Similarly, an outlier is not necessarily a mislabelled sample [42], since it can be due to feature noise or simply be a low-probability event.

B. Sources of Label Noise

As outlined in [1], the identification of the source(s) of label noise is not necessarily important, when the focus of the analysis is on the consequences of label noise. However, when a label noise model has to be embedded directly into the learning algorithm, it may be important to choose a modelling which accurately explains the actual label noise.

Label noise naturally occurs when human experts are involved [43]. In that case, possible causes of label noise include imperfect evidence, patterns which may be confused with the patterns of interest, perceptual errors or even biological artefacts. See e.g. [44], [45] for a philosophical account on probability, imprecision and uncertainty. More generally, potential sources of label noise include four main classes.

Firstly, the information which is provided to the expert may be insufficient to perform reliable labelling [1], [46]. For example, the results of several tests may be unknown in medical applications [12]. Moreover, the description language may be too limited [47], what reduces the amount of available information. In some cases, the information is also of poor or variable quality. For example, the answers of a patient during anamnesis may be imprecise or incorrect or even may be different if the question is repeated [48].

Secondly, as mentioned above, errors can occur in the expert labelling itself [1]. Such classification errors are not always due to human experts, since automated classification devices are used nowadays in different applications [12]. Also, since collecting reliable labels is a time-consuming and costly task, there is an increasing interest in using cheap, easy-to-get labels from non-expert using frameworks like e.g. the Amazon Mechanical Turk\(^1\) [49]–[52]. Labels provided by non-expert are less reliable, but Snow et al. [49] show that the wealth of available labels may alleviate this problem.

Thirdly, when the labelling task is subjective, like e.g. in medical applications [53] or image data analysis [54], [55], there may exist an important variability in the labelling by several experts. For example, in electrocardiogram analysis, experts seldom agree on the exact boundaries of signal patterns [56]. The problem of inter-expert variability was also noticed during the labelling of the Penn Treebank, an annotated corpus of over 4.5 million words [57].

Eventually, label noise can also simply come from data encoding or communication problems [3], [11], [46]. For example, in spam filtering, sources of label noise include misunderstanding the feedback mechanisms and accidental click [58]. Real-world databases are estimated to contain around five

\(^1\)https://www.mturk.com
percents of encoding errors, all fields taken together, when no specific measures are taken [59]–[61].

C. Taxonomy of Label Noise

In the context of missing values, Schafer and Graham [10] discuss a taxonomy which is adapted below to provide a new taxonomy for label noise. Similarly, Nettleton et al. [62] characterise noise generation in terms of its distribution, the target of the noise (features, label, etc.) and whether its magnitude depends on the data value of each variable. Since it is natural to consider label noise from a statistical point of view, Fig. 1 shows three possible statistical models of label noise. In order to model the label noise process, four random variables are depicted: \( X \) is the vector of features, \( Y \) is the true class, \( \tilde{Y} \) is the observed label and \( E \) is a binary variable telling whether a labelling error occurred (\( Y \neq \tilde{Y} \)). The set of possible feature values is \( \mathcal{X} \), whereas the set of possible classes (and labels) is \( \mathcal{Y} \). Arrows report statistical dependencies: for example, \( \tilde{Y} \) is assumed to always depend on \( Y \) (otherwise, there is no sense in using the labels).

1) The Noisy Completely at Random Model: In Fig. 1(a), the relationship between \( Y \) and \( \tilde{Y} \) is called noisy completely at random (NCAR): the occurrence of an error \( E \) is independent of the other random variables, including the true class itself. In the NCAR case, the observed label is different from the true class with a probability \( p_e = P(E = 1) = P(Y \neq \tilde{Y}) \) [11], sometime called the error rate or the noise rate [63]. In other words, a biased coin is firstly flipped in order to decide whether the observed label is correct or not. If the label is wrong, a fair dice with \( n_y \) possible feature values is tossed to choose the observed, wrong label. This particularly simple model is called the uniform label noise.

2) The Noisy at Random Model: In Fig. 1(b), it is assumed that the probability of error depends on the true class \( Y \), what is called here noisy at random (NAR). \( E \) is still independent of \( X \), but this model allows modelling asymmetric label noise, i.e. when instances from certain classes are more prone to be mislabelled. For example, in medical case-control studies, control subjects may be more likely to be mislabelled. Indeed, the test which is used to label case subjects may be too invasive (e.g. a biopsy) or too expensive to be used on control subjects and is therefore replaced by a suboptimal diagnostic test for control subjects [66]. Since one can define the labelling probabilities

\[
P(\tilde{Y} = \hat{y}|Y = y) = \sum_{e \in \{0,1\}} P(\tilde{Y} = \hat{y}|E = e, Y = y)P(E = e|Y = y),
\]

the NAR label noise can equivalently be characterised in terms of the labelling (or transition) matrix [67], [68]

\[
\gamma = \begin{pmatrix}
\gamma_{11} & \cdots & \gamma_{1n_y} \\
\vdots & \ddots & \vdots \\
\gamma_{n_y1} & \cdots & \gamma_{n yn_y}
\end{pmatrix}
\begin{pmatrix}
P(\tilde{Y} = 1|Y = 1) & \cdots & P(\tilde{Y} = n_y|Y = 1) \\
\vdots & \ddots & \vdots \\
P(\tilde{Y} = 1|Y = n_y) & \cdots & P(\tilde{Y} = n_y|Y = n_y)
\end{pmatrix}
\]

(2)

where \( n_y = |\mathcal{Y}| \) is the number of classes. Each row of the labelling matrix must sum to 1, since \( \sum_{\hat{y} \in \mathcal{Y}} P(\tilde{Y} = \hat{y}|Y = y) = 1 \). For example, the uniform label noise corresponds to the labelling matrix

\[
\begin{pmatrix}
1 - p_e & \cdots & p_e \\
\vdots & \ddots & \vdots \\
p_e & \cdots & 1 - p_e
\end{pmatrix}
\]

(3)

Notice that NCAR label noise is a special case of NAR label noise. When true classes are known, the labelling probabilities can be directly estimated by the frequencies of mislabelling in data, but it is seldom the case [48]. Alternately, one can also use the incidence-of-error matrix [48]

\[
\begin{pmatrix}
\pi_{11} \gamma_{11} & \cdots & \pi_{11} \gamma_{1n_y} \\
\vdots & \ddots & \vdots \\
\pi_{n_y n_y} \gamma_{n_y n_y} & \cdots & \pi_{n_y n_y} \gamma_{n_y n_y}
\end{pmatrix}
\begin{pmatrix}
P(Y = 1, \tilde{Y} = 1) & \cdots & P(Y = 1, \tilde{Y} = n_y) \\
\vdots & \ddots & \vdots \\
P(Y = n_y, \tilde{Y} = 1) & \cdots & P(Y = n_y, \tilde{Y} = n_y)
\end{pmatrix}
\]

(4)

where \( \pi_y = P(Y = y) \) is the prior of class \( y \). The entries of the incidence-of-error matrix sum to one and may be of more practical interest.

With the exception of uniform label noise, NAR label noise is the most commonly studied case of label noise in the literature. For example, Lawrence and Schölkopf [67] consider arbitrary labelling matrices. In [3], [69], pairwise label noise is introduced: 1) two classes \( c_1 \) and \( c_2 \) are selected, then 2) each instance of class \( c_1 \) has a probability to be incorrectly labelled as \( c_2 \) and vice versa. For this label noise, only two non-diagonal entries of the labelling matrix are non-zero.

In the case of NAR label noise, it is no longer trivial to decide whether the labels are helpful or not. One solution is to compute the expected probability of error

\[
p_e = P(E = 1) = \sum_{y \in \mathcal{Y}} P(Y = y)P(E = 1|Y = y)
\]

(5)

and to require that \( p_e < \frac{1}{2} \), similarly to NCAR label noise. However, this condition does not prevent the occurrence of very small correct labelling probabilities \( P(\tilde{Y} = y|Y = y) \) for some class \( y \in \mathcal{Y} \), in particular if the prior probability \( P(y) \) of this class is small. Instead, conditional error probabilities \( p_e(y) = P(E = 1|Y = y) \) can also be used.
3) The Noisy not at Random Model: Most works on label noise consider that the label noise affects all instances with no distinction. However, it is not always realistic to assume the two above types of label noise [11], [70]. For example, samples may be more likely mislabelled when they are similar to instances of another class [70]–[76], as illustrated e.g. in [77] where empirical evidence is given that more difficult samples in a text entailment dataset are labelled randomly. It also seems natural to expect less reliable labels in regions of low density [78]–[80], where experts predictions may be actually based on a very small number of similar previously encountered cases.

Let us consider a more complex and realistic model of label noise. In Fig. 1(c), $E$ depends on both variables $X$ and $Y$, i.e. mislabelling is more probable for certain classes and in certain regions of the $X$ space. This noisy not at random (NNAR) model is the most general case of label noise [81], [82]. For example, mislabelling near the classification boundary or in low density regions can only be modelled in terms of NNAR label noise. Such a situation occurs e.g. in speech recognition, where automatic speech recognition is more difficult in case of phonetic similarity between the correct word and the recognised word [83]. The context of each word can be considered in order to detect incorrect recognitions. Notice that the medical literature distinguishes differential (feature-dependent, i.e. NNAR) label noise and non-differential (feature-independent, i.e. NCAR or NAR) label noise [84].

The reliability of labels is even more complex to estimate that for NCAR or NAR label noise. Indeed, the probability of error also depends in that case on the value of $X$. As before, one can define an expected probability of error which becomes

$$p_e = P(E = 1) = \sum_{y \in Y} P(Y = y) \times$$

$$\int_{x \in X} P(X = x | Y = y) P(E = 1 | X = x, Y = y) dx \quad (6)$$

if $X$ is continuous. However, this quantity does not reflect the local nature of label noise: in some cases, $p_e$ can be almost zero although the density of labelling errors shows important peaks in certain regions. The quantity $p_e(x, y) = P(E = 1 | X = x, Y = y)$ may therefore be more appropriate to characterise the reliability of labels.

III. CONSEQUENCES OF LABEL NOISE ON LEARNING

In this section, the potential consequences of label noise are described to show the necessity to take label noise into account in learning problems. Section III-A reviews theoretical and empirical evidences of the impact of label noise on classification performances, which is the most frequently reported issue. Section III-B shows that the presence of label noise also increases the necessary number of samples for learning, as well as the complexity of models. Label noise may also pose a threat for related tasks, like e.g. class frequencies estimation and feature selection, which are discussed in Section III-C and Section III-D, respectively.

This section presents the negative consequences of label noise, but artificial label noise also has potential advantages. For example, label noise can be added in statistical studies to protect people privacy: it is e.g. used in [28] to obtain statistics for questionnaires, while making impossible to recover individual answers. In [85]–[89], label noise is added to improve classifier results. Whereas bagging produces different training sets by resampling, these works copy the original training set and switch labels in new training sets to increase the variability in data.

A. Deterioration of Classification Performances

The more frequently reported consequence of label noise is a decrease in classification performances, as shown in the theoretical and experimental works described below.

1) Theoretical Studies of Simple Classifiers: There exist several theoretical studies of the consequences of label noise on prediction performances. For simple problems and symmetric label noise, the accuracy of classifiers may remain unaffected. Lachenbruch [71] consider e.g. the case of binary classification when both classes have Gaussian distribution with identical covariance matrix. In such a case, a linear discriminant function can be used. For a large number of samples, the consequence of uniform noise is noticeable only if the error rates $\alpha_1$ and $\alpha_2$ in each class are different. In fact, the change in decision boundary is completely described in terms of the difference $\alpha_1 - \alpha_2$. These results are also discussed asymptotically in [90].

The results of Lachenbruch [71] are extended in [91] for quadratic discriminant functions, i.e. Gaussian conditional distributions with unequal covariance matrices. In that case,
prediction is affected even when label noise is symmetric among classes \((\alpha_1 = \alpha_2)\). Consequences worsen when differences in covariance matrices or misclassification rates increase. Michalek and Tripathi [92] and Bi and Jeske [93] show that label noise affects normal discriminant and logistic regression: their error rates are increased and their parameters are biased. Logistic regression seems to be less affected.

In [64], the single-unit perceptron is studied in the presence of label noise. If the teacher providing learning samples is absent-minded, i.e. labels are flipped with a given probability (uniform noise), the performances of a learner who takes the labels for granted are damaged and even get worse than the performances of the teacher.

Classification performances of the \(k\) nearest neighbours (\(k\)NN) classifier are also affected by label noise [94], [95], in particular when \(k = 1\) [96]. Okamoto and Nobuhiro [96] present an average-case analysis of the \(k\)NN classifier. When \(k\) is optimised, the consequences of label noise are reduced and remain small unless a large amount of label noise is added. The optimal value of \(k\) depends on both the number of training instances and the presence of label noise. For small noise-free training sets, \(1\)NN classifiers are often optimal. But as soon as label noise is added, the optimal number of neighbours \(k\) is shown to monotonically increase with the number of instances even for small training sets, what seems natural since \(1\)NN classifiers are particularly affected by label noise.

2) Experimental Assessment of Specific Models: Apart from theoretical studies, many works show experimentally that label noise may be harmful. First of all, the impact of label noise is not identical for all types of classifiers. As detailed in Section V, this fact can be used to cope (at least partially) with label noise. For example, Nettleton et al. [62] compare the impact of label noise on four different supervised learners: naive Bayes, decision trees induced by C4.5, \(k\)NNs and support vector machines (SVMs). In particular, naive Bayes achieves the best results, what is attributed to the conditional independence assumption and the use of conditional probabilities. This should be contrasted with the results in [12], where naive Bayes is sometime dominated by C4.5 and \(k\)NNs. The poor results of SVMs are attributed to its reliance on support vectors and the feature interdependence assumption.

In text categorization, Zhang and Yang [97] consider the robustness of regularized linear classification methods. Three linear methods are tested by randomly picking and flipping labels: linear SVMs, Ridge regression and logistic regression. The experiments show that the results are dramatically affected by label noise for all three methods, which obtain almost identical performances. Only 5% of flipped labels already leads to a dramatic decrease of performances, what is explained by the presence of relatively very small classes with only a few samples in their experiments.

Several studies have shown that boosting [98] is affected by label noise [99]–[102]. In particular, the adaptive boosting algorithm AdaBoost tends to spend too much efforts on learning mislabelled instances [100]. During learning, successive weak learners are trained and the weights of instances which are misclassified at one step are increased at the next step. Hence, in the late stages of learning, AdaBoost tends to increase the weights of mislabelled instances and starts overfitting [103], [104]. Dietterich [100] clearly shows that the mean weight per training sample becomes larger for mislabelled samples than for correctly labelled samples as learning goes on. Interestingly, it has been shown in [105]–[108] that AdaBoost tends to increase the margins of the training examples [109] and achieves asymptotically a decision with hard margin, very similar to the one of SVMs for the separable case [108]. This may not be a good idea in the presence label noise and may explain why AdaBoost overfits noisy training instances. In [110], it is also shown that ensemble methods can fail simply because the presence of label noise affects the ensembled models. Indeed, learning through multiple models becomes harder for large levels of label noise, where some samples become more difficult for all models and are therefore seldom correctly classified by an individual model.

In systems which learn Boolean concepts with disjuncts, Weiss [111] explains that small disjuncts (which individually cover only a few examples) are more likely to be affected by label noise than large disjuncts covering more instances. However, only large levels of label noise may actually be a problem. For decision trees, it appears in [2] that destroying class information produces a linear increase in error. Taking logic to extremes, when all class information is noise, the resulting decision tree classifies objects entirely randomly.

Another example studied in [58] is spam filtering where performances are decreased by label noise. Spam filters tend to overfit label noise, due to aggressive online update rules which are designed to quickly adapt to new spam.

3) Additional Results for More Complex Types of Label Noise: The above works deal with NAR label noise, but more complex types of label noise have been studied in the literature. For example, in the case of linear discriminant analysis (LDA), i.e. binary classification with normal class distributions, Lachenbruch [70] considers that mislabelling systematically occurs when samples are too far from the mean of their true class. In that NAR label noise model, the true probabilities of misclassification are only slightly affected, whereas the populations are better separated. This is attributed to the reduction of the effects of outliers. However, the apparent error rate [112] of LDA is highly influenced, what may cause the classifier to overestimate its own efficiency.

LDA is also studied in the presence of label noise by [72], which generalises the results of [70], [71], [90]–[92]. Let us define 1) the misallocation rate \(\alpha_y\) for class \(y\), i.e. the number of samples with label \(y\) which belong to the other class and 2) a \(z\)-axis which passes through the center of both classes and is oriented towards the positive class, such that each center is located at \(z = \pm \frac{\sqrt{2}}{2}\). In [72], three label noise models are defined and characterised in terms of the probability of misallocation \(g_y(z)\), which is a monotone decreasing (increasing) function of \(z\) for positive (negative) samples. In random misallocation, \(g_y(z) = \alpha_y\) is constant for each class, what is equivalent to the NAR label noise. In truncated label noise, \(g(z) = 0\) as long as the instance is close enough to the mean of its class. Afterwards, the mislabelling probability is equal to a small constant. This type of NNAR label noise is equivalent to the model of
when the constant is equal to one. Eventually, in the exponential model, the probability of misallocation becomes for the negative class

\[ g_y(z) = \begin{cases} 0 & \text{if } z \leq -\frac{\Delta}{2} \\ 1 - \exp \left( -\frac{1}{2} k_y (z + \frac{\Delta}{2})^2 \right) & \text{if } z > -\frac{\Delta}{2} \end{cases} \]  

(7)

where \( \Delta \) is the distance between the centres of both classes and \( k_y = (1 - 2\alpha_y)^{-2} \). A similar definition is given for the positive class. For equivalent misallocation rates \( \alpha_y \), random misallocation has more consequences than truncated label noise, in terms of influence on the position and variability of the discriminant boundary. In turn, truncated label noise itself has more consequences than exponential label noise. The same ordering appears when comparing misclassification rates.

B. Consequences on Learning Requirements and Model Complexity

Label noise can affect learning requirements (e.g., number of necessary instances) or the complexity of learned models. For example, Quinlan [2] warns that the size of decision trees may increase in case of label noise, making them overly complicated, what is confirmed experimentally in [46]. Similarly, Abellán and Masegosa [104] show that the number of nodes of decision trees induced by C4.5 for bagging is increased, while the resulting accuracy is reduced. Reciprocally, Brodley and Friedl [46] and Libralon et al. [113] show that removing mislabelled samples reduces the complexity of SVMs (number of support vectors), decision trees induced by C4.5 (size of trees) and rule-based classifiers induced by RIPPER (number of rules). Post-pruning also seems to reduce the consequences of label noise [104]. Noise reduction can therefore produce models which are easier to understand, what is desirable in many circumstances [114–116].

In [11], it is shown that the presence of uniform label noise in the probably approximately correct (PAC) framework [117] increases the number of necessary samples for PAC identification. An upper bound for the number of necessary samples is given, which is strengthened in [118]. Similar bounds are also discussed in [65], [119]. Also, Angluin and Laird [11] discuss the feasibility of PAC learning in the presence of label noise for propositional formulas in conjunctive normal form (CNF), what is extended in [120] for Boolean functions represented by decision trees and in [73], [121] for linear perceptrons.

C. Distortion of Observed Frequencies

In medical applications, it is often necessary to perform medical tests for disease diagnosis, to estimate the prevalence of a disease in a population or to compare (estimated) prevalence in different populations. However, label noise can affect the observed frequencies of medical test results, what may lead to incorrect conclusions. For binary tests, Bross [4] shows that mislabelling may pose a serious threat: the observed mean and variance of the test answer is strongly affected by label noise. Let us consider a simple example taken from [4]: if the minority class represents 10% of the dataset and 5% of the test answers are incorrect (i.e. patients are mislabelled), the observed proportion of minority cases is 0.95 × 10% + 0.05 × 90% = 14% and is therefore overestimated by 40%. Significance tests which assess the difference between the proportions of both classes in two populations are still valid in case of mislabelling, but their power may be strongly reduced. Similar problems occur e.g. in consumer survey analysis [122].

Frequency estimates are also affected by label noise in multiclass problems. Hout and Heijden [28] discuss the case of artificial label noise, which can be intentionally introduced after data collection in order to preserve privacy. Since the label noise is fully specified in this case, it is possible to adjust the observed frequencies. When a model of the label noise is not available, Tenenbein [123] proposes to solve the problem pointed by [4] using double sampling, which uses two labelers: an expensive, reliable labeller and a cheap, unreliable labeler. The model of mislabelling can thereafter be learned from both sets of labels [124], [125]. In [48], the case of multiple experts is discussed in the context of medical anamnesis; an algorithm is proposed to estimate the error rates of the experts.

Evaluating the error rate of classifiers is also important for both model selection and model assessment. In that context, Lam and Stork [126] show that label noise can have an important impact on the estimated error rate, when test samples are also polluted. Hence, mislabelling can also bias model comparison. As an example, a spam filter with a true error rate of 0.5%, for example, might be estimated to have an error rate between 5.5% and 6.5% when evaluated using labels with an error rate of 6.0%, depending on the correlation between filter and label errors [127].

D. Consequences for Related Tasks

The aforementioned consequences are not the only possible consequences of label noise. For example, Zhang et al. [128] show that the consequences of label noise are important in feature selection for microarray data. In an experiment, only one mislabelled sample already leads to about 20% of not identified discriminative genes. Notice that in microarray data, only a few data are available. Similarly, Shanab et al. [129] show that label noise decreases the stability of feature rankings. The sensitivity of feature selection to label noise is also illustrated for logistic regression in [130]. A methodology to achieve feature selection for classification problems polluted by label noise is proposed in [131], based on a probabilistic label noise model combined with a nearest neighbours-based estimator of the mutual information.

E. Conclusion

This section shows that the consequences of label noise are important and diverse: decrease in classification performances, changes in learning requirements, increase in the complexity of learned models, distortion of observed frequencies, difficulties to identify relevant features, etc. The nature and the importance of the consequences depend, among others, on the type and the level of label noise, the learning algorithm and the characteristics of the training set. Hence, it seems important
for the machine learning practitioner to deal with label noise and to consider these factors, prior to the analysis of polluted data.

IV. METHODS TO DEAL WITH LABEL NOISE

In light of the various consequences detailed in Section III, it seems important to deal with label noise. In the literature, there exist three main approaches to take care of label noise [12], [82], [132]–[137]; these approaches are described below. Manual review of training samples is not considered in this survey, because it is usually prohibitively costly and time consuming, if not impossible in the case of large datasets.

A first approach relies on algorithms which are naturally robust to label noise. In other words, the learning of the classifier is assumed to be not too sensitive to the presence of label noise. Indeed, several studies have shown that some algorithms are less influenced than others by label noise, what advocates for this approach. However, label noise is not really taken into account in this type of approach. In fact, label noise handling is entrusted to overfitting avoidance [132]–[134].

Secondly, one can try to improve the quality of training data using filter approaches. In such a case, noisy labels are typically identified and being dealt with before training occurs. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

A first approach relies on algorithms which are naturally robust to label noise. In other words, the learning of the classifier is assumed to be not too sensitive to the presence of label noise. Indeed, several studies have shown that some algorithms are less influenced than others by label noise, what advocates for this approach. However, label noise is not really taken into account in this type of approach. In fact, label noise handling is entrusted to overfitting avoidance [132]–[134].

Secondly, one can try to improve the quality of training data using filter approaches. In such a case, noisy labels are typically identified and being dealt with before training occurs. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

A first approach relies on algorithms which are naturally robust to label noise. In other words, the learning of the classifier is assumed to be not too sensitive to the presence of label noise. Indeed, several studies have shown that some algorithms are less influenced than others by label noise, what advocates for this approach. However, label noise is not really taken into account in this type of approach. In fact, label noise handling is entrusted to overfitting avoidance [132]–[134].

Secondly, one can try to improve the quality of training data using filter approaches. In such a case, noisy labels are typically identified and being dealt with before training occurs. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

A first approach relies on algorithms which are naturally robust to label noise. In other words, the learning of the classifier is assumed to be not too sensitive to the presence of label noise. Indeed, several studies have shown that some algorithms are less influenced than others by label noise, what advocates for this approach. However, label noise is not really taken into account in this type of approach. In fact, label noise handling is entrusted to overfitting avoidance [132]–[134].

Secondly, one can try to improve the quality of training data using filter approaches. In such a case, noisy labels are typically identified and being dealt with before training occurs. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

A first approach relies on algorithms which are naturally robust to label noise. In other words, the learning of the classifier is assumed to be not too sensitive to the presence of label noise. Indeed, several studies have shown that some algorithms are less influenced than others by label noise, what advocates for this approach. However, label noise is not really taken into account in this type of approach. In fact, label noise handling is entrusted to overfitting avoidance [132]–[134].

Secondly, one can try to improve the quality of training data using filter approaches. In such a case, noisy labels are typically identified and being dealt with before training occurs. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

A first approach relies on algorithms which are naturally robust to label noise. In other words, the learning of the classifier is assumed to be not too sensitive to the presence of label noise. Indeed, several studies have shown that some algorithms are less influenced than others by label noise, what advocates for this approach. However, label noise is not really taken into account in this type of approach. In fact, label noise handling is entrusted to overfitting avoidance [132]–[134].

Secondly, one can try to improve the quality of training data using filter approaches. In such a case, noisy labels are typically identified and being dealt with before training occurs. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

A first approach relies on algorithms which are naturally robust to label noise. In other words, the learning of the classifier is assumed to be not too sensitive to the presence of label noise. Indeed, several studies have shown that some algorithms are less influenced than others by label noise, what advocates for this approach. However, label noise is not really taken into account in this type of approach. In fact, label noise handling is entrusted to overfitting avoidance [132]–[134].

Secondly, one can try to improve the quality of training data using filter approaches. In such a case, noisy labels are typically identified and being dealt with before training occurs. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.

Eventually, there exist algorithms which directly model label noise during learning or which have been modified to take label noise into account in an embedded fashion. The advantage of this approach is to separate the classification model and the label noise model, what allows using information about the nature of label noise.

The literature for the three above trends of approaches is reviewed in the three next sections. In some cases, it is not always clear whether an approach belongs to one category or the other. For example, some of the label noise-tolerant variants of SVMs could also be seen as filtering. Mislabelled instances can either be relabelled or simply removed [138]. Filter approaches are cheap and easy to implement, but some of them are likely to remove a substantial amount of data.
Section VII.A: probabilistic label noise-tolerant methods

A. Bayesian approaches [68] including e.g. 1) priors on the mislabelling probabilities [5], [122], [227], [228] like Beta priors [5], [128], [229], [230], [232]–[236] and Dirichlet priors [237], [238], 2) Bayesian methods for logistic regression [130], [236], [239]–[241], hidden Markov models [84] and graphical models [242] and 3) procedures based on mislabelling indicator variables [128], [235], [245], [246]

Section VII.B model-based label noise-tolerant methods

B.1 embedded data cleansing for SVMs [273]–[277] and robust losses [280], [285] to produce label noise-tolerant SVMs without filtering

B.2 label noise-tolerant variants of the perceptron algorithm [286] like the λ-trick [287], [288], the α-bound [289] and PAM [286], [290]

B.3 decision trees with a good trade-off between accuracy and simplicity obtained using e.g. the CN2 algorithm [291]

B.4 boosting methods that 1) carefully update weights like MadaBoost [292], AveBoost [293] and AveBoost2 [294], 2) combine bagging and boosting like BB [297] and MB [298] and 3) distinguish safe, noisy and borderline patterns like reverse boosting [299]

B.5 semi-supervised methods that 1) prevent mislabelled instances to influence the label of unlabelled instances [7], 2) detect mislabelled instances using unlabelled instances [300]–[302] and 3) deal with mistakes done when labelling unlabelled samples like in [304]–[306] or in the case of co-training [307]–[309]

Section V: label noise-robust methods

A robust losses for classification [74], [81], [82], [140]

B ensemble methods like LogitBoost [141], BrownBoost [142] and boosting with margin maximisation [108], [143]–[145]

C split criteria for trees like the imprecise info-gain [104], [148]–[150]

Section VI: data cleansing methods

A detection of mislabelled instances with measures like the classification confidence [157] and the model complexity [158]–[162]

B model predictions-based filtering, i.e. 1) classification filtering that remove misclassified training instances [165]–[167] with e.g. local models [115], [116], [174], [178], 2) voting filtering [46], [138], [161], [173], [180], [182]–[184] and 3) partition filtering [69], [185]

C model influence [53], [187], [188] and introspection [64]

D k nearest-neighbours-based methods [95], [193], including e.g. CNN [195], RNN [196], BBNR [197], DROPl-6 [95], [193], GE [29], [200], IB3 [204], [205], Tomek links [206], [207] and PRISM [208]

E neighbourhood graph-based methods [94], [209], [212]–[214]

F ensemble-based methods with removal of e.g. instances with highest weights [184], [215] and often misclassified instances [217]

Section VII: label noise-robust methods

A.2 frequentist approaches including e.g. mixture models [249], [250]

A.3 clustering methods assigning clusters to classes [136], [262], [263]

A.4 belief function-based methods that directly infer belief masses from data [78], [80], [271] to account for the uncertainty on labels

D. Other Methods

Most of the studies on label noise robustness have been presented in Section III. They show that complete label noise robustness is seldom achieved, as discussed in Section V-A. An exception is [81], where the 0-1 loss is directly optimised using a team of continuous-action learning automata: 1) a probability distribution is defined on the weights of a linear classifier, then 2) weights are repetitively drawn from the distribution to classify training samples and 3) the 0-1 losses for the training samples are used at each iteration as a reinforcement to progressively tighten the distribution around the optimal weights.

In the case of separable classes, the approach converges to the true optimal separating hyperplane, even in the case of NNAR label noise. In [151], eleven classifiers are compared on imbalanced datasets with asymmetric label noise. In all cases, the performances of the models are affected by label noise. Random forests [147] are shown to be the most robust among the eleven methods, what is also the case in another study by the same authors [152]. C4.5, radial basis function (RBF) networks and rule-based classifiers obtain the worst results. The sensitivity of C4.5 to label noise is confirmed in [153], where multilayer perceptrons are shown to be less affected. In [135], a new artificial immune recognition system (AIRS) is proposed, called RWTSAIRS, which is shown to be less sensitive to label noise. In [154], two procedures based on argumentation theory are also shown to be robust to label noise. In [12], it is shown that feature extraction can help

TABLE I

<table>
<thead>
<tr>
<th>Section V: label noise-robust methods</th>
<th>Section VII.A: probabilistic label noise-tolerant methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. robust losses for classification [74], [81], [82], [140]</td>
<td>A. Bayesian approaches [68] including e.g 1) priors on the mislabelling probabilities [5], [122], [227], [228] like Beta priors [5], [128], [229], [230], [232]–[236] and Dirichlet priors [237], [238], 2) Bayesian methods for logistic regression [130], [236], [239]–[241], hidden Markov models [84] and graphical models [242] and 3) procedures based on mislabelling indicator variables [128], [235], [245], [246]</td>
</tr>
<tr>
<td>B. ensemble methods like LogitBoost [141], BrownBoost [142] and boosting with margin maximisation [108], [143]–[145]</td>
<td>B.1 embedded data cleansing for SVMs [273]–[277] and robust losses [280], [285] to produce label noise-tolerant SVMs without filtering</td>
</tr>
<tr>
<td>C. split criteria for trees like the imprecise info-gain [104], [148]–[150]</td>
<td>B.2 label noise-tolerant variants of the perceptron algorithm [286] like the λ-trick [287], [288], the α-bound [289] and PAM [286], [290]</td>
</tr>
<tr>
<td></td>
<td>B.3 decision trees with a good trade-off between accuracy and simplicity obtained using e.g. the CN2 algorithm [291]</td>
</tr>
<tr>
<td></td>
<td>B.4 boosting methods that 1) carefully update weights like MadaBoost [292], AveBoost [293] and AveBoost2 [294], 2) combine bagging and boosting like BB [297] and MB [298] and 3) distinguish safe, noisy and borderline patterns like reverse boosting [299]</td>
</tr>
<tr>
<td></td>
<td>B.5 semi-supervised methods that 1) prevent mislabelled instances to influence the label of unlabelled instances [7], 2) detect mislabelled instances using unlabelled instances [300]–[302] and 3) deal with mistakes done when labelling unlabelled samples like in [304]–[306] or in the case of co-training [307]–[309]</td>
</tr>
</tbody>
</table>

C. Decision trees

It is well-known that decision trees are greatly impacted by label noise [2], [104]. In fact, their instability makes them well suited for ensemble methods [146]–[148]. In [148], different node split criteria are compared for ensembles of decision trees in the presence of label noise. The imprecise info-gain [149] is shown to improve accuracy, with respect to the information gain, the information gain ratio and the Gini index. Compared to ensembles of decision trees inferred by C4.5, Abellán and Masegosa [104] also show that the imprecise info-gain allows reducing the size of the decision trees. Eventually, they observe that post-pruning of decision trees can reduce the impact of label noise. The approach is extended for continuous features and missing data in [150].

D. Other Methods

Most of the studies on label noise robustness have been presented in Section III. They show that complete label noise robustness is seldom achieved, as discussed in Section V-A. An exception is [81], where the 0-1 loss is directly optimised using a team of continuous-action learning automata: 1) a probability distribution is defined on the weights of a linear classifier, then 2) weights are repetitively drawn from the distribution to classify training samples and 3) the 0-1 losses for the training samples are used at each iteration as a reinforcement to progressively tighten the distribution around the optimal weights.

In the case of separable classes, the approach converges to the true optimal separating hyperplane, even in the case of NNAR label noise. In [151], eleven classifiers are compared on imbalanced datasets with asymmetric label noise. In all cases, the performances of the models are affected by label noise. Random forests [147] are shown to be the most robust among the eleven methods, what is also the case in another study by the same authors [152]. C4.5, radial basis function (RBF) networks and rule-based classifiers obtain the worst results. The sensitivity of C4.5 to label noise is confirmed in [153], where multilayer perceptrons are shown to be less affected. In [135], a new artificial immune recognition system (AIRS) is proposed, called RWTSAIRS, which is shown to be less sensitive to label noise. In [154], two procedures based on argumentation theory are also shown to be robust to label noise. In [12], it is shown that feature extraction can help

The table highlights the structure of each section, summarises their respective content and points to specific references.
to reduce the impact of label noise. Also, Sánchez et al. [9], [155] shows that using one-vs-one decomposition in multiclass problems can improve the robustness, which could be due to the distribution of the noisy examples in the subproblems, the increase of the separability of the classes and collecting information from different classifiers.

E. Discussion

Theoretically, common losses in machine learning are not completely robust to label noise [139]. However, overfitting avoidance techniques like e.g. regularisation can be used to partially handle label noise [132]–[134], even if label noise may interfere with the quality of the classifier, whose accuracy might suffer and the representation might be less compact [132]. Experiments in the literature show that the performances of classifiers inferred by label noise-robust algorithms are still affected by label noise. Label noise-robust methods seem to be adequate only for simple cases of label noise, which can be safely managed by overfitting avoidance.

VI. DATA CLEANSING METHODS FOR LABEL NOISE-POLLUTED DATASETS

When training data is polluted by label noise, an obvious and tempting solution consists in cleansing the training data themselves, what is similar to outlier or anomaly detection. However, detecting mislabelled instances is seldom trivial: Weiss and Irsh [156] show e.g. in the context of learning with disjuncts that true exceptions may be hard to distinguish from mislabelled instances. Hence, many methods have been proposed to cleanse training sets, with different degrees of success. The whole procedure is illustrated by Fig. 2, which is inspired by [46]. This section describes several methods which detect, remove or relabel mislabelled instances. First, simple methods based on thresholds are presented in Section VI-A. Model prediction-based filtering methods are discussed in Section VI-B, which includes classification filtering, voting filtering and partition filtering. Methods based on measures of the impact of label noise and introspection are considered in Section VI-C. Sections VI-D, VI-E and VI-F address methods based on nearest neighbours, graphs and ensembles. Eventually, several other methods are discussed in Section VI-G and a general discussion about data cleansing methods is proposed in Section VI-H.

A. Measures and Thresholds

Similarly to outlier detection [30]–[33] and anomaly detection [34]–[38], several methods in label noise cleansing are based on ad hoc measures. Instances can e.g. be removed when the anomaly measure exceeds a predefined threshold. For example, in [157], the entropy of the conditional distribution \( P(Y|X) \) is estimated using a probabilistic classifier. Instances with a low entropy correspond to confident classifications. Hence, such instances for which the classifier disagrees with the observed label are relabelled using the predicted label.

As discussed in Section III, label noise may increase the complexity of inferred models. Therefore, complexity measures can be used to detect mislabelled instances, which disproportionately increase model complexity when added to the training set. In [158], the complexity measure for inductive concept learning is the number of literals in the hypothesis. A cleansing algorithm is proposed, which 1) finds for each literal the minimal set of training samples whose removal would allow going without the literal and 2) awards one point to each sample in the minimal set. Once all literals have been reviewed, the sample with the higher score is removed, if the score is high enough. This heuristic produces less complex models. Similarly, Gamberger and Lavrač [159] measure the complexity of the least complex correct hypothesis (LCCH) for a given training set. Each training set is characterised by a LCCH value and is saturated if its LCCH value is equal to the complexity of the target hypothesis. Mislabelled samples are removed to obtain a saturated training set. Gamberger et al. [160]–[162] elaborate on the above notions of complexity and saturation, which result in the so-called saturation filter.

B. Model Predictions-Based Filtering

Several data cleansing algorithms rely on the predictions of classifiers: classification filtering, voting filtering and partition filtering. In [163], such methods are extended in the context of cost-sensitive learning, whereas Khoshgoftaar and Rebours [164] propose a generic algorithm which can be specialised to classification filtering, voting filtering or partition filtering by a proper choice of parameters.

1) Classification Filtering: The predictions of classifiers can be used to detect mislabelled instances, what is called classification filtering [161], [164]. For example, Thongkam et al. [165] learn a SVM using the training data and removes all instances which are misclassified by the SVM. A similar method is proposed in [166] for neural networks. Miranda et al. [167] extend the approach of [165]: four classifiers are induced by different machine learning techniques and are combined by voting to detect mislabelled instances. The above methods can be applied to any classifier, but it eliminates all instances which stand on the wrong side of the classification boundary, what be can dangerous [168], [169]. In fact, as discussed in [170], classification filtering (and data cleansing in general) suffers from a chicken-and-egg dilemma, since 1) good classifiers are necessary for classification filtering and 2) learning in the presence of label noise may precisely produce poor classifiers. An alternative is proposed in [169], which 1) defines a pattern as informative if it is difficult to predict by a model trained on previously seen data and 2) sent a pattern to the human operator for checking if its informativeness is above a threshold found by cross-validation. Indeed, such patterns can either be atypical patterns that are actually informative or garbage patterns. The level of surprise is considered to be a good indication of how informative a pattern is, what is quantified by the information gain \( -\log P(Y = y|X = x) \).

In [171], an iterative procedure called robust-C4.5 is introduced. At each iteration, 1) a decision tree is inferred and pruned by C4.5 and 2) training samples which are misclassified by the pruned decision tree are removed. The procedure is akin to regularisation, in that the model is repeatedly made simpler. Indeed, each iteration removes training samples, what
in turn allows C4.5 to produce smaller decision trees. Accuracy is slightly improved, whereas the mean and variance of the tree size are decreased. Hence, smaller and more stable decision trees are obtained, which also perform better. Notice that caution is advised when comparing sizes of decision trees in data cleaning [172], [173]. Indeed, Oates and Jensen [172] show that the size of decision trees naturally tends to increase linearly with the number of instances. It means that the removal of randomly selected training samples already leads to a decrease in tree sizes. Therefore, Oates and Jensen [172] propose the measure

$$100 \times \left(\frac{\text{initial tree size} - \text{tree size with random filtering}}{\text{initial tree size} - \text{tree size with studied filtering}}\right)$$

(8)

to estimate the percentage of decrease in tree size which is simply due to a reduction in the number of samples. For example, Oates and Jensen [172] show experimentally for robust-C4.5 that 42% of the decrease in tree size can be imputed to the sole reduction in training set size, whereas the remaining 58% are due to an appropriate choice of the instances to be removed. A similar analysis could be done for other methods in this section.

Local models [174] can also be used to filter mislabelled training samples. Such models are obtained by training a standard model like e.g. LDA [175] or a SVM [176], [177] on a training set consisting of the $k$ nearest neighbours of the sample to be classified. Many local models have to be learnt, but the respective local training sets are very small. In [116], local SVMs are used to reject samples for which the prediction is not confident enough. In [115], the local SVM noise reduction method is extended for large datasets, by reducing the number of SVMs to be trained. In [178], a sample is removed if it is misclassified by a $k$ nearest centroid neighbours classifier [179] trained when the sample itself is removed from the training set.

2) Voting Filtering: Classification filtering faces the risk to remove too many instances. In order to solve this problem, ensembles of classifiers are used in [46], [138], [180] to identify mislabelled instances, what is inspired by outlier removal in regression [181]. The first step consists in using a $K$-fold cross-validation scheme, which creates $K$ pairs of distinct training and validation datasets. For each pair of sets, $m$ learning algorithms are used to learn $m$ classifiers using the training set and to classify the samples in the validation set. Therefore, $m$ classifications are obtained for each sample, since each instance belongs to exactly one validation set. The second step consists in inferring from the $m$ predictions whether a sample is mislabelled or not, what is called voting filtering in [173] or ensemble filtering in [164]. Two possibilities are studied in [46], [138], [180]: a majority vote and a consensus vote. Whereas majority vote classifies a sample as mislabelled if a majority of the $m$ classifiers misclassified it, the consensus vote requires that all classifiers have misclassified the sample. One can also require high agreement of classifiers, i.e. misclassification by more than a given percentage of the classifiers [182]. The consensus vote is more conservative than the majority vote and results in fewer removed samples. The majority vote tends to throw out too many instances [183], but performs better than consensus vote, because keeping mislabelled instances seems to harm more than removing too many correctly labelled samples.

The $K$-fold cross-validation is also used in [161]. For each training set, a classifier is learnt and directly filters its corresponding validation set. The approach is intermediate between [165] and [46], [138], [180] and has been shown to be non-selective, i.e. too many samples are detected as being potentially noisy [161]. Eventually, Verbaeten [173] performs an experimental comparison of some of the above methods and proposes several variants. In particular, $m$ classifiers from the same type are learnt using all combinations of the $K-1$ parts in the training set. Voting filters are also iterated until no more samples are removed. In [184], voting filters are obtained by generating the $m$ classifiers using bagging: $m$ training sets are generated by resampling and the inferred classifiers are used to classify all instances in the original training set.

3) Partition Filtering: Classification filtering is adapted for large and distributed datasets in [69], [185], which proposes a partition filter. In the first step, samples are partitioned and rules are inferred for each partition. A subset of good rules are chosen for each partition using two factors which measure the classification precision and coverage for the partition. In a second step, all samples are compared to the good rules of all partitions. If a sample is not covered by a set of rules, it is not classified, otherwise it is classified according to these rules. This mechanism allows distinguishing between exceptions (not covered by the rules) and mislabelled instances (covered by the rules, but misclassified). Majority or consensus vote is used to detect mislabelled instances. Privacy is preserved in distributed datasets, since each site (or partition) only shares its good rules. The approach is experimentally shown to be less aggressive than [161]. In [186], partitioning is repeated and several classifiers are learned for each partition. If all classifiers predict the same label which is different from the observed label, the instance is considered as potentially mislabelled. Votes are summed over all iterations and can be used to order the instances.

C. Model Influence and Introspection

Mislabelled instances can be detected by analysing their impact on learning. For example, Malossini et al. [53] define
the leave-one-out perturbed classification (LOOPC) matrix where the \((i,j)\) entry is the label predicted for the \(j\)th training sample if 1) the \(j\)th sample itself is removed from the training set and 2) the label of the \(i\)th sample is flipped. The LOOPC matrix is defined only for binary classification. Two algorithms are proposed to analyse the LOOPC matrix in search for wrong labels. The classification-stability algorithm (CL-stability) analyses each column to detect suspicious samples: good samples are expected to be consistently classified even in the case of small perturbation in training data. The leave-one-out-error-sensitivity (LOOE-sensitivity) algorithm detects samples whose label flip improves the overall results of the classifier. The computation of the LOOPC matrix is expensive, but it can be afforded for small datasets. Experiments show that CL-stability dominates LOOE-sensitivity. The approach is extended in [187], [188].

Based on introspection, Heskes [64] proposes an online learning algorithm for the single-unit perceptron, when labels coming from the teacher are polluted by uniform noise. The presented samples are accepted only when the confidence of the learner in the presented labelled sample is large enough. The propensity of the learner to reject suspicious labels is called the stubbornness: the learner only accepts to be taught when it does not contradict its own model too much. The stubbornness of the learner has to be tuned, since discarding too many samples may slow the learning process. An update rule is proposed for the student self-confidence: the stubbornness is increased by learner-teacher contradictions, whereas learner-teacher agreements decrease stubbornness. The update rule itself depends on the student carefullness, which reflects the confidence of the learner and can be chosen to outperform any absent-minded teacher.

D. \(k\) Nearest Neighbours-Based Methods

The \(k\) nearest neighbours (\(k\)NN) classifiers [189], [190] are sensitive to label noise [94], [95], in particular for small neighbourhood sizes [96]. Hence, it is natural that several methods have emerged in the \(k\)NN literature for cleansing training sets. Among these methods, many are presented as editing methods [191], what may be a bit misleading: most of these methods do not edit instances, but rather edit the training set itself by removing instances. Such approaches are also motivated by the particular computational and memory requirements of \(k\)NN methods for prediction, which linearly depend on the size of the training set. See e.g. [192] for a discussion on instance selection methods for case-based reasoning.

Wilson and Martinez [95], [193] provide a survey of \(k\)NN-based methods for data cleansing, propose several new methods and perform experimental comparisons. Wilson and Martinez [95] show that mislabelled training instances degrade the performances of both the \(k\)NN classifiers built on the full training set and the instance selection methods which are not designed to take care of label noise. This section presents solutions from the literature and is partially based on [95], [193]. See e.g. [194] for a comparison of several instance-based noise reduction methods.

\(k\)NN-based instance selection methods are mainly based on heuristics. For example, the condensed nearest neighbour (CNN) rule [195] builds a subset of training instances which allows classifying correctly all other training instances. However, such a heuristic systematically keeps mislabelled instances in the training set. There exist other heuristics which are more robust to label noise. For example, the reduced nearest neighbours (RNN) rule [196] successively removes instances whose removal do not cause other instances to be misclassified, i.e. it removes noisy and internal instances. The blame-based noise reduction (BBNR) algorithm [197] removes all instances which contribute to the misclassification of another instance and whose removal does not cause any instance to be misclassified. In [198], [199], instances are ranked based on a score rewarding the patterns that contribute to a correct classification and punishing those that provide a wrong one. An important danger of instance selection is to remove too many instances [200], if not all instances in some pathological cases [95].

More complex heuristics exist in the literature; see e.g. [113], [201] for an experimental comparison for gene expression data. For example, Wilson [202] removes instances whose label is different from the majority label in its \(k = 3\) nearest neighbours. This method is extended in [203] by the all-\(k\)NN method. In [95], [193], six heuristics are introduced and compared with other methods: DROP1-6. For example, DROP2 is designed to reduce label noise using the notion of instance associates, which have the instance itself in their \(k\) nearest neighbours. DROP2 removes an instance if its removal does not change the number of its associates which are incorrectly classified in the original training set. This algorithm tends to retain instances which are close to the classification boundary. In [200], generalised edition (GE) checks whether there are at least \(k'\) samples in the locally majority class among the \(k\) neighbours of an instance. In such a case, the instance is relabelled with the locally majority label, otherwise it is simply removed from the training set. This heuristic aims at keeping only instances with strong support for their label. Barandela and Gasca [29] show that a few repeated applications of the GE algorithm improves results in the presence of label noise.

Other instance selection methods designed to deal with label noise include e.g. IB3 which employs a significance test to determine which instances are good classifiers and which ones are believed to be noisy [204], [205]. Lorena et al. [206] propose to use Tomek links [207] to filter noisy instances for splice junction recognition. Different instance selection methods are compared in [114]. In [192], a set of instances are selected by using Fisher discriminant analysis, while maximising the diversity of the reduced training set. The approach is shown to be robust to label noise for a simple artificial example. In [208], different heuristics are used to distinguish three types of training instances: normal instances, border samples and instances which should be misclassified (ISM). ISM instances are such that, based on the information in the dataset, the label assigned by the learning algorithm is the most appropriate even though it is incorrect. For example, one of the heuristics uses a nearest neighbours approach to estimate the hardness of a training sample, i.e. how hard it is
to classify correctly. ISM instances are simply removed, what results in the so-called PRISM algorithm.

E. Graph-Based Methods

Several methods in the data cleansing literature are similar to $k$NN-based editing methods, except that they represent training sets by neighbourhood graphs [209], where the instances (or nodes) are linked to other close instances. The edge between two instances can be weighted depending on the distance between them. Such methods work directly on the graphs to detect noisy instances. For example, Sánchez et al. [94] propose variants of $k$NN-based algorithms which use Gabriel graphs and relative neighbourhood graphs [210], [211]. In [212], [213], mode filters, which preserve edges and remove impulsive noise in images, are extended to remove label noise in datasets represented by a graph. In [209], [214], the $i$th instance is characterised by its local cut edge weight statistic $J_i$, which is the sum of the weights of edges linking the instance to its neighbours with a different label. Three types of instances are distinguished: good samples with a small $J_i$, doubtful samples with an intermediate $J_i$, and bad samples with a large $J_i$. Two filtering policies are considered: 1) to relabel doubtful samples and to remove bad samples or 2) to relabel doubtful and bad samples using the majority class in good neighbours (if any) and to remove doubtful and bad samples which have no good neighbours.

F. Ensemble and Boosting-Based Methods

As discussed in Section III-A2, AdaBoost is well known to overfit noisy datasets. Indeed, the weights of mislabelled instances tend to become much larger than the weights of normal instances in the late iterations of AdaBoost. Several works presented below show that this propensity to overfitting can be exploited in order to remove label noise.

A simple data cleansing method is proposed in [184], which removes a given percentage of the samples with the highest weights after $m$ iterations of AdaBoost. Experiments show that the precision of this boosting-based algorithm is not very good, what is attributed to the dynamics of Adaboost. In the first iterations, mislabelled instances quickly obtain large weights and are correctly spotted as mislabelled. However, consequently, several correctly labelled instances then obtain large weights in late iterations, what explains that they are incorrectly removed from the training set by the boosting filter.

A similar approach is pursued in [215]. Outlier removal boosting (ORBoost) is identical to AdaBoost, except that instance weights which are above a certain threshold are set to zero during boosting. Hence, data cleansing is performed while learning and not after learning as in [184]. ORBoost is sensitive to the choice of the threshold, which is performed using validation. In [216], mislabelled instances are also removed during learning, if they are misclassified by the ensemble with high confidence.

In [217], edge analysis is used to detect mislabelled instances. The edge of an instance is defined as the sum of the weights of weak classifiers which misclassified the instance [218]. Hence, an instance with a large edge is often misclassified by the weak learners and is classified by the ensemble with a low confidence, what is the contrary of the margin defined in [106]. Wheway [217] observes a homogenisation of the edge as the number of weak classifiers increases: the mean of the edge stabilises and its variance goes to zero. It means that observations which were initially classified correctly are classified incorrectly in later rounds in order to classify harder observations correctly, what is consistent with results in [106], [218]. Mislabelled data have edge values which remain high due to persistent misclassification. It is therefore proposed to remove the instances corresponding e.g. to the 5% top edge values.

G. Others Methods

There exist other methods for data cleansing. For example, in ECG segmentation, Hughes et al. [56] delete the label of the instances (and not the instances themselves) which are close to classification boundaries, since experts are known to be less reliable in that region. Thereafter, semi-supervised learning is performed using both the labelled and the (newly) unlabelled instances. In [219], a genetic algorithm approach based on a class separability criterion is proposed. In [220], [221], the automatic data enhancement (ADE) method and the automatic noise reduction (ANR) method are proposed to relabel mislabelled instances with a neural network approach. A similar approach is proposed in [222] for decision trees.

H. Discussion

One of the advantages of label noise cleansing is that removed instances have absolutely no effects on the model inference step [158]. In several works, it has been observed that simply removing mislabelled instances is more efficient than relabelling them [167], [223]. However, instance selection methods may remove too many instances [132]–[134], [200], if not all instances in some pathological cases [95]. On the one hand, Matic et al. [168] show that overcleansing may reduce the performances of classifiers. On the other hand, it is suggested in [46] that keeping mislabelled instances may harm more than removing too many correctly labelled samples. Therefore, a compromise has to be found. The overcleansing problem is of particular importance for imbalanced datasets [224]. Indeed, minority instances may be more likely to be removed by e.g. classification filtering (because they are also more likely to be misclassified), what makes learning even more difficult. In [225], it is shown that dataset imbalance can affect the efficiency of data cleansing methods. Label noise cleansing can also reduce the complexity of inferred models, but it is not always trivial to know if this reduction is not simply due to the reduction of the training set size [172], [173].

Surprisingly, to the best of our knowledge, the method in [56] has not been generalised to other label noise cleansing methods, what would be easy to do. Indeed, instead of completely removing suspicious instances, one could only delete their labels and perform semi-supervised learning on the resulting training set. The approach in [56] has the advantage of keeping the distribution of the instances unaltered (what is
not the case for their conditional distributions, though), what is of particular interest for generative approaches. An interesting open research question is whether this method would improve the results with respect to the classical solution of simply removing suspicious instances. Another alternative would be to resubmit the suspicious samples to a human expert for removing suspicious instances. Another alternative would be the results with respect to the classical solution of simply

VII. LABEL NOISE-TOLERANT LEARNING ALGORITHMS

When some information is available about label noise or its consequences on learning, it becomes possible to design models which take label noise into account. Typically, one can learn a label noise model simultaneously with a classifier, what uncouples both components of the data generation process and improves the resulting classifier. In a nutshell, the resulting classifier learns to classify instances according to their true, unknown class. Other approaches consist in modifying the learning algorithm in order to reduce the influence of label noise. Data cleansing can also be embedded directly into the learning algorithm, like e.g. for SVMs. Such techniques are described in this section and are called label noise-tolerant, since they can tolerate label noise by modelling it. Section VII-A reviews probabilistic methods, whereas model-based methods are discussed in Section VII-B.

A. Probabilistic Methods

Many label noise-tolerant methods are probabilistic, in a broad sense. They include Bayesian and frequentist methods, as well as methods based on clustering or belief functions. An important issue which is highlighted by these methods is the identifiability of label noise. The four families of methods are discussed in the following four subsections.

1) Bayesian Approaches: Detecting mislabelled instances is a challenging problem. Indeed, there are identifiability issues [226]–[228], as illustrated in [122], where consumers answer a survey with some error probability. Under the assumption that it results in a Bernoulli process, it is possible to obtain an infinite number of maximum likelihood solutions for the true proportions of answers and the error probabilities. In other words, in this simple example, it is impossible to identify the correct model for observed data. Several works claim that prior information is strictly necessary to deal with label noise. In particular, [5], [122], [227], [228] propose to use Bayesian priors on the mislabelling probabilities to break ties. Label noise identifiability is also considered for inductive logic programming in [226], where a minimal description length principle prevents the model to overfit on label noise.

Several Bayesian methods to take care of label noise are reviewed in [68] and summarised here. In medical applications, it is often necessary to assess the quality of binary diagnosis tests with label noise. Three parameters must be estimated: the population prevalence (i.e. the true proportion of positive samples) and the sensitivity and specificity of the test itself [5]. Hence, the problem has one degree of freedom in excess, since only two data-driven constraints can be obtained (linked to the observed proportions of positive and negative samples). In [5], [229], [230], it is proposed to fix the degree of freedom using a Bayesian approach: setting a prior on the model parameters disambiguates maximum likelihood solutions. Indeed, whereas the frequentist approach considers that parameters have fixed values, the Bayesian approach considers that all unknown parameters have a probability distribution that reflects the uncertainty in their values and that prior knowledge about unknown parameters can be formally included [231]. Hence, the Bayesian approach can be seen as a generalisation of constraints on the parameters values, where the uncertainty on the parameters is taken into account through priors.

Popular choices for Bayesian priors for label noise are Beta priors [5], [128], [229], [230], [232]–[236] and Dirichlet priors [237], [238], which are the conjugate priors of binomial and multinomial distributions, respectively. Bayesian methods have also been designed for logistic regression [130], [236], [239]–[241], hidden Markov models [84] and graphical models for medical image segmentation [242]. In the Bayesian approaches, although the posterior distribution of parameters may be difficult (or impossible) to calculate directly, efficient implementations are possible using Markov chain Monte Carlo (MCMC) methods, which allow approximating the posterior of model parameters [68]. A major advantage of using priors is the ability to include any kind of prior information in the learning process [68]. However, the priors should be chosen carefully, for the results obtained depend on the quality of the prior distribution used [243], [244].

In the spirit of the above Bayesian approaches, an iterative procedure is proposed in [128] to correct labels. For each sample, Rekaya et al. [235] define an indicator variable $\alpha_i$ which is equal to 1 if the label of the $i$th instance was switched. Hence, each indicator follows a Bernoulli distribution parametrised by the mislabelling rate (which itself follows a Beta prior). In [128], the probability that $\alpha_i = 1$ is estimated for each sample and the sample with the higher mislabelling probability is relabelled. The procedure is repeated as long as the test is significant. Indicators are also used in [245] for Alzheimer disease prediction, where four out of sixteen patients are detected as potentially misdiagnosed. The correction of the supposedly incorrect labels leads to a significant increase in predictive ability. A similar approach is used in [246] to robustify multiclass Gaussian process classification. If the indicator for a given sample is zero, then the label of that sample is assumed to correspond to a latent function. Otherwise, the label is assumed to be randomly chosen. The same priors as in [235] are used and the approach is shown to yield better results than other methods which assume that the latent function is polluted by a random Gaussian noise [247] or which use Gaussian processes with heavier tails [248].

2) Frequentist Methods: Since label noise is an inherently stochastic process, several frequentist methods have emerged to deal with it. A simple solution consists in using mixture models, which are popular in outlier detection [32]. In [249], each sample is assumed to be generated either from a majority (or normal) distribution or an anomalous distribution, with respective priors $1 - \lambda$ and $\lambda$. The expert error probability $\lambda$ is assumed to be relatively small. Depending on prior knowledge,
any appropriate distribution can be used to model the majority and anomalous distributions, but the anomalous distribution may be simply chosen as uniform. The set of anomalous samples is initially empty, i.e., all samples initially belong to the majority set. Samples are successively tested and added to the anomalous set whenever the increase in log-likelihood due to this operation is higher than a predefined threshold. Mansour and Parnas [250] also consider the mixture model and propose an algorithm to learn conjunctions of literals.

Directly linked with the definition of NAR label noise in Section II-C, Lawrence and Schölkopf [67] propose another probabilistic approach to label noise. The label of an instance is assumed to correspond to two random variables (see Fig. 3, inspired by [67]): the true hidden label $Y$ and the observed label $\tilde{Y}$, which is possibly noisy. $\tilde{Y}$ is assumed to depend only on the true label $Y$, whose relationship is described by a labelling matrix (see Section II-C2). Using this simple model of label noise, a Fisher discriminant is learned using an EM approach. Eventually, the approach is kernelised and is shown to effectively deal with label noise. Interestingly, the probabilistic modelling also leads to an estimation of the noise level.

Later, Li et al. [251] extended this model by relaxing the Gaussian distribution assumption and carried out extensive experiments on more complex datasets, which convincingly demonstrated the value of explicit label noise modelling. More recently, the same model has been extended to multiclass datasets [252] and sequential data [253]. Asymmetric label noise is also considered in [66] for logistic regression. It is shown that conditional probabilities are altered by label noise and that this problem can be solved by taking a model of label noise into account. A similar approach was developed for neural networks in [254], [255] for uniform label noise. Repeatedly, a neural network is trained to predict the conditional probability of each class, what allows optimising the mislabelling probability before retraining the neural network. The mislabelling probability is optimised either using a validation set [254] or a Bayesian approach with a uniform prior [255]. In [256], Gaussian processes for classification are also adapted for label noise by assuming that each label is potentially affected by a uniform label noise. It is shown that label noise modelling increases the likelihood of observed labels when label noise is actually present.

Valizadegan and Tan [257] propose a method based on a weighted KNN. Given the probability $p_i$ that the $i$th training example is mislabelled, the binary label $y_i$ is replaced by its expected value $-p_i y_i + (1 - p_i) y_i = (1 - 2 p_i) y_i$. Then, the sum of the consistencies

$$\delta_i = (1 - 2 p_i) y_i \sum_{j \in N(x_i)} w_{ij} (1 - 2 p_j) y_j$$

(9)

between the expected value of $y_i$ and the expected value of the weighted KNN prediction is maximised, where $N(x_i)$ contains the neighbours of $x_i$ and $w_{ij}$ is the weight of the $j$th neighbour. To avoid declaring all the examples from one of the two classes as mislabelled, a $L1$ regularisation is enforced on the probabilities $p_i$.

Contrarily to the methods described in Section VII-A1, Bayesian priors are not used in the above frequentist methods.

We hypothesise that the identifiability problem discussed in Section VII-A1 is solved by using a generative approach and setting constraints on the conditional distribution of $X$. For example, in [67], Gaussian distributions are used, whereas Li et al. [251] consider mixtures of Gaussian distributions. The same remark applies to Section VII-A3.

3) Clustering-Based Methods: In the generative statistical models of Section VII-A2, it is assumed that the distribution of instances can help to solve classification problems. Classes are not arbitrary; they are linked to a latent structure in the distribution of $X$. In other words, clusters in instances can be used to build classifiers, what is done in [136]. Firstly, a clustering of the instances [258] is performed using an unsupervised algorithm. Labels are not used and the procedure results in a mixture of $K$ models $p_k(x)$ with priors $\pi_k$ for components $k = 1 \ldots K$. Secondly, instances are assumed to follow the density

$$p(x) = \sum_{y \in \mathcal{Y}} \sum_{k=1}^{K} r_{yk} \pi_k p_k(x)$$

(10)

where $r_{yk}$ can be interpreted as the probability that the $k$th cluster belongs to the $y$th class. The coefficients $r_{yk}$ are learned using a maximum likelihood approach. Eventually, classification is performed by computing the conditional probabilities $P(Y = y | X = x)$ using both the unsupervised (clusters) and supervised ($r_{yk}$ probabilities) parts of the model.

When a Gaussian mixture model is used to perform clustering, the mixture model can be interpreted as a generalisation of mixture discriminant analysis (MDA, see [259]). In this case, the model is called robust mixture discriminant analysis (RMDA) and is shown to improve classification results with respect to MDA [136], [260]. In [261], the method is adapted to discrete data for DNA barcoding and is called robust discrete discriminant analysis. In that case, data are modelled by a multivariate multinomial distribution. A clustering approach is also used in [262] to estimate a confidence on each label, where each instance inherits the distribution of classes within its assigned cluster. Confidences are averaged over several clusterings and a weighted training set is obtained.

In this spirit, El Gayar et al. [263] propose a method which is similar to [136]. Labels are converted into soft labels in order to reflect the uncertainty on labels. Firstly, a fuzzy clustering of the training instances is performed, which gives a set of cluster and the membership of each instance to each cluster. Then, the membership $L_{yk}$ of the $k$th cluster to the
yth class is estimated using the fuzzy memberships. Each instance with label $y$ increases the membership $L_{yk}$ by its own membership to cluster $k$. Eventually, the fuzzy label of each instance is computed using the class memberships of the clusters where the instance belongs. Experiments show improvements with respect to other label fuzzification methods like $k$NN soft labels and Keller soft labels [264].

4) Belief Functions: In the belief function theory, each possible subset of classes is characterised by a belief mass, which is the amount of evidence which supports the subset of classes [265]. For example, let us consider an expert who 1) thinks that a given case is positive, but 2) has a very low confidence in its own prediction. In the formalism of belief functions, one can translate the above judgement by a belief mass function (BMF, also called basic probability assignment) $m$ such that $m(\{−1, +1\}) = 0.8$, $m(\{-1\}) = 0$ and $m(\{+1\}) = 0.2$. Here, there is no objective uncertainty on the class itself, but rather a subjective uncertainty on the judgement itself. For example, if a coin is flipped, the BMF would simply be $m(\{\text{head}, \text{tail}\}) = 1$, $m(\{\text{head}\}) = 0$ and $m(\{\text{tail}\}) = 0$ when the bias of the coin is unknown. If the coin is known to be unbiased, the BMF becomes $m(\{\text{head}, \text{tail}\}) = 0$, $m(\{\text{head}\}) = \frac{1}{2}$ and $m(\{\text{tail}\}) = \frac{1}{2}$. Again, this simple example illustrates how the belief function theory allows distinguishing subjective uncertainty from objective uncertainty. Notice that Smets [266] argues that it is necessary to fall back to classical probabilities in order to make decisions. Different decision rules are analysed in [79]. Interestingly, the belief function formalism can be used to modify standard machine learning methods like e.g. $k$NN classifiers [78], neural networks [80], decision trees [267], mixture models [268], [269] or boosting [270].

In the context of this survey, belief functions cannot be used directly, since the belief masses are not available. Indeed, they are typically provided by the expert itself as an attempt to quantify its own (lack of) confidence, but we made the hypothesis in Section I that such information is not available. However, several works have proposed heuristics to infer belief masses directly from data [78], [80], [271].

In [78], a $k$NN approach based on Dempster-Shafer theory is proposed. If a new sample $x_s$ has to be classified, each training sample $(x_i, y_i)$ is considered as an evidence that the class of $x_s$ is $y_i$. The evidence is represented by a BMF $m_{s,i}$ such that $m_{s,i}(\{y_i\}) = \alpha$, $m_{s,i}(\forall) = 1 - \alpha$ and $m_{s,i}$ is zero for all other subsets of classes, where

$$\alpha = \alpha_0 \Phi(d_{s,i})$$ (11)

such that $0 < \alpha_0 < 1$ and $\Phi$ is a monotonically decreasing function of the distance $d_{s,i}$ between both instances. There are many possible choices for $\Phi$:

$$\Phi(d) = \exp(-\gamma d^\beta)$$ (12)

is chosen in [78], where $\gamma > 0$ and $\beta \in \{1, 2\}$. Heuristics are proposed to select proper values of $\alpha_0$ and $\gamma$. For the classification of the new sample $x_s$, each training sample provides an evidence. These evidences are combined using the Dempster rule and it becomes possible to take a decision (or to refuse to take a decision if the uncertainty is too high). The case of mislabelling is experimentally studied in [78], [272] and the approach is extended to neural networks in [80].

In [271], a $k$NN approach is also used to infer BMFs. For a given training sample, the frequency of each class in its $k$ nearest neighbours is computed. Then, the sample is assigned to a subset of classes containing 1) the class with the maximum frequency and 2) the classes whose frequency is not too different from the maximum frequency. A neural network is used to compute beliefs for test samples.

B. Model-Based Methods

Apart from probabilistic methods, specific strategies have been developed to obtain label noise-tolerant variants of popular learning algorithms, including e.g. support vector machines, neural networks and decision trees. Many publications also propose label noise-tolerant boosting algorithms, since boosting techniques like AdaBoost are well-known to be sensitive to label noise. Eventually, label noise is also tackled in semi-supervised learning. These five families of methods are discussed in the following five subsections.

1) Support Vector Machines and Robust Losses: SVMs are not robust to label noise [62], [82], even if instances are allowed to be misclassified during learning. Indeed, instances which are misclassified during learning are penalised in the objective using the hinge loss

$$[1 - y_i(x_i, w)]_+ (13)$$

where $[z]_+ = \max(0, z)$ and $w$ is the weight vector. The hinge loss increases linearly with the distance to the classification boundary and is therefore significantly affected by mislabelled instances which stand far from the boundary.

Data cleansing can be directly implemented into the learning algorithm of SVMs. For example, instances which correspond to very large dual weights can be identified as potentially mislabelled [273]. In [274], $k$ samples are allowed to be not taken into account in the objective function. For each sample, a binary variable (indicating whether or not to consider the sample) is added and the sum of the indicators is constrained to be equal to $k$. An opposite approach is proposed in [275] for aggregated training sets, which consists of several distinct training subsets labelled by different experts. The percentage of support vectors in training samples is constrained to be identical in each subset, in order to decrease the influence of low-quality teachers which tend to require more support vectors due to more frequent mislabelling. In [276], [277], SVMs are adapted by weighting the contribution of each training sample in the objective function. The weights (or fuzzy memberships) are computed using heuristics. Similar work is done in [278] for relevance vector machines (RVMs). Empathetic constraints SVMs [279] relax the constraints of suspicious samples in the SVM optimisation problem.

Xu et al. [280] propose a different approach, which consists in using the loss

$$\eta_i [1 - y_i(x_i, w)]_+ + (1 - \eta_i) (14)$$

where $0 \leq \eta_i \leq 1$ indicates whether the $i$th sample is an outlier. The $\eta_i$ variables must be optimised together with the
weights vector, what is shown to be equivalent to using the robust hinge loss
\[
\min(1, [1 - y_i \langle x_i, w \rangle]^+).
\] (15)

Notice that there exist other bounded, non-convex losses [281]–[284] which could be used similarly. A non-convex loss is also used in [285] to produce label noise-tolerant SVMs. A non-convex loss computed on the noise-free training set. In other words, it is possible to estimate the noise-free errors from the noisy data. Theoretical guarantees are given and the proposed approach is shown to outperform SVMs, but error probabilities must be known a priori.

2) Neural Networks: Different label noise-tolerant variants of the perceptron algorithm are reviewed and compared experimentally in [286]. In the standard version of this algorithm, samples are presented repeatedly (on-line) to the classifier. If a sample is misclassified, i.e.
\[
y_i [wx_i + b] < 0
\] (17)
where \( w \) is the weight vector and \( b \) is the bias, then the weight vector is adjusted towards this sample. Eventually, the perceptron algorithm converges to a solution.

Since the solution of the perceptron algorithm can be biased by mislabelled samples, different variants have been designed to reduce the impact of mislabelling. With the \( \lambda \)-trick [287], [288], if an instance has already been misclassified, the adaptation criterion becomes \( y_i [wx_i + b] + \lambda \|x_i\|^2 < 0 \). Large values of \( \lambda \) may prevent mislabelled instances to trigger updates. Another heuristic is the \( \alpha \)-bound [289], which does not update \( w \) for samples which have already been misclassified \( \alpha \) times. This simple solution limits the impact of mislabelled instances. Although not directly designed to deal with mislabelling, Khardon and Wachman [286] also describe the perceptron algorithm using margins (PAM, see [290]). PAM updates \( w \) for instances with \( y_i [wx_i + b] < \tau \), similarly to support vector classifiers and to the \( \lambda \)-trick.

3) Decision Trees: Decision trees can easily overfit data, if they are not pruned. In fact, learning decision trees involves a trade-off between accuracy and simplicity, which are two requirements for good decision trees in real-world situations [291]. It is particularly important to balance this trade-off in the presence of label noise, what makes the overfitting problem worse. For example, Clark and Niclett [291] propose the CN2 algorithm which learns a disjunction of logic rules while avoiding too complex ones.

4) Boosting Methods: In boosting, an ensemble of weak learners \( h_t \) with weights \( \alpha_t \) is formed iteratively using a weighted training set. At each step \( t \), the weights \( w_t^{(i)} \) of misclassified instances are increased (resp. decreased for correctly classified samples), what progressively reduces the ensemble training error because the next weak learners focus on the errors of the previous ones. As discussed in Section III, boosting methods tend to overfit label noise. In particular, AdaBoost obtains large weights for mislabelled instances in late stages of learning. Hence, several methods propose to update weights more carefully to reduce the sensitivity of boosting to label noise. In [292], MadaBoost imposes an upper bound for each instance weight, which is simply equal to the initial value of that weight. The AveBoost and AveBoost2 [293], [294] algorithms replace the weight \( w_t^{(i+1)} \) of the \( t \)th instance at step \( t + 1 \) by
\[
\frac{t w_t^{(i)} + w_t^{(i+1)}}{t+1}.
\] (18)

With respect to AdaBoost, AveBoost2 obtains larger training errors, but smaller generalisation errors. In other words, AveBoost2 is less prone to overfitting than AdaBoost, what improves results in the presence of label noise. Kim [295] proposes another ensemble method called Averaged Boosting (A-Boost), which 1) does not take instances weights into account to compute the weights of the successive weak classifiers and 2) performs similarly to bagging on noisy data. Other weighting procedures have been proposed in e.g. [296], but they were not assessed in the presence of label noise.

In [297], two approaches are proposed to reduce the consequences of label noise in boosting. Firstly, AdaBoost can be early-stopped: limiting the number of iterations prevents AdaBoost from overfitting. A second approach consists in smoothing the results of AdaBoost. The proposed BB algorithm combines bagging and boosting: 1) \( K \) training sets consisting of \( \rho \) percents of the training set (sub-sampled with replacement) are created, 2) \( K \) boosted classifiers are trained for \( M \) iterations and 3) the predictions are aggregated. In [297], it is advised to use \( K = 15 \), \( M = 15 \) and \( \rho = \frac{1}{2} \). The BB algorithm is shown to be less sensitive to label noise than AdaBoost. A similar approach is proposed in [298]: the multiple boosting (MB) algorithm.

A reverse boosting algorithm is proposed in [299]. In adaptive boosting, weak learners may have difficulties to obtain good separation frontiers because correctly classified samples get lower and lower weights as learning goes on. Hence, safe, noisy and borderline patterns are distinguished, whose weights are respectively increased, decreased and unaltered during boosting. Samples are classified into these three categories using parallel perceptrons, a specific type of committee machine whose margin allows to separate the input space into three regions: a safe region (beyond the margin), a noisy region (before the margin) and a borderline region (inside the margin). The approach improves the results of parallel perceptrons in the presence of label noise, but is most often dominated by classical perceptrons.

5) Semi-Supervised Learning: In [7], a particle competition-based algorithm is proposed to perform semi-supervised learning in the presence of label noise. Firstly, the dataset is converted into a graph, where instances are nodes with edges between similar instances. Each labelled node is associated with a labelled particle. Particles walk through the graph and cooperate with identically-labelled
particles to label unlabelled instances, while staying in the
neighbourhood of their home node. What interests us in [7]
is the behaviour of mislabelled particles: they are pushed
away by the particles of near instances with different labels,
what prevents a mislabelled instance to influence the label of
close unlabelled instances. In [300], unlabelled instances are
firstly labelled using a semi-supervised learning algorithm,
then the new labels are used to filter instances. Similarly,
context-sensitive semi-supervised support vector machines
[301], [302] first use labelled instances to label unlabelled
instances which are spatially close (e.g. in images) to them
and second these new semilabels are used to reduce the
effect of mislabelled training instances. Other works on
label noise for semi-supervised learning include e.g. [303]
or [304]–[306], which are particular because they model the
label noise induced by the labelling of unlabelled samples. A
similar problem occur in co-training [307]–[309] where two
different views are available for each instance, like e.g. the
text in a web page and the text attached to the hyperlinks
pointing to this page. In the seminal work of Blum and
Mitchell [307], co-training consists in 1) learning two distinct
weak predictors from labelled data with each of the two
views, 2) predicting labels with the weak predictors for a
random subset of the unlabelled data and 3) keeping the most
confident labels to enlarge the pool of labelled instances. See
e.g. [310]–[314] for examples of studies on the effectiveness
of co-training. Co-training allows each weak predictor to
provide labels to improve the other weak predictor, but
the problem is that each weak predictor is likely to make
prediction errors. Incorrect labels are a source of label noise
which has to be taken into account, like e.g. in [308], [309].

C. Discussion

The probabilistic methods to deal with label noise are
grounded in a more theoretical approach than robust or data
cleansing methods. Hence, probabilistic models of label noise
can be directly used and allow to take advantage of prior
knowledge. Moreover, the model-based label noise-tolerant
methods allow us to use the knowledge gained by the analysis
of the consequences of label noise. However, the main problem
of the approaches described in this section is that they increase
the complexity of learning algorithms and can lead to over-
fitting, because of the additional parameters of the training
data model. Moreover, the identifiability issue discussed in
Section VII-A1 must be addressed, what is done explicitly in
the Bayesian approach (using Bayesian priors) and implicitly
in the frequentist approach (using generative models).

As highlighted in [1], different models should be used for
training and testing in the presence of label noise. Indeed,
a complete model of the training data consists of a label
noise model and a classification model. Both parts are used
during training, but only the classification model is useful for
prediction: one has no interest in making noisy predictions.
Dropping the label noise model is only possible when label
noise is explicitly modelled, as in the probabilistic approaches
discussed in Section VII-A. For other approaches, the learning
process of the classification model is supposed to be robust or
tolerant to label noise and to produce a good classification
model.

VIII. EXPERIMENTS IN THE PRESENCE OF LABEL NOISE

This section discusses how experiments are performed in
the label noise literature. In particular, existing datasets, label
noise generation techniques and quality measures are high-
lighted.

A. Datasets with Identified Mislabelled Instances and Label
Noise Generation Techniques

There exist only a few datasets where incorrect labels have
been identified. Among them, Lewis et al. [315] provide a
version of the Reuters dataset with corrected labels and
Malossini et al. [53] propose a short analysis of the reliability
of instances for two microarray datasets. In spam filtering,
where the expert error rate is usually between 3% and 7%,
the TREC datasets have been carefully labelled by experts
adhering to the same definition of spam, with a resulting
expert error rate of about 0.5% [127]. Mislabelling is also
discussed for a medical image processing application in [316]
and Alzheimer disease prediction in [245]. However, artificial
label noise is more common in the literature. Most studies on
label noise use NCAR label noise, which is introduced in real
datasets by 1) randomly selecting instances and 2) changing
their label onto one of the other remaining labels [135]. In
this case, label noise is independent of $Y$. In [317], it is also
proposed to simulate label noise for artificial datasets by 1)
computing the membership probabilities $P(Y = y|X = x)$
for each training sample $x$, 2) adding a small uniform noise
to these values and 3) choosing the label corresponding to the
largest polluted membership probability.

Several methods have been proposed to introduce NAR
label noise. For example, in [62], label noise is artificially
introduced by changing the labels of some randomly chosen
instances from the majority class. In [3], [69], [301], label
noise is introduced using a pairwise scheme. Two classes $c_1$
and $c_2$ are selected, then each instance of class $c_1$ has a
probability $P_e$ to be incorrectly labelled as $c_2$ and vice versa.
In other words, this label noise models situations where only
certain types of classes are mislabelled. In [1], label noise
is introduced by increasing the entropy of the conditional
mass function $P(Y|X)$. The proposed procedure is called
majorisation: it leaves the probability of the majority class
unchanged, but the remaining probability is spread more
evenly on the other classes, with respect to the true conditional
mass function $P(Y|X)$. In [151], [153], the percentage of
mislabelled instances is firstly chosen, then the proportions
of mislabelled instances in each class are fixed.

NNAR label noise is considered in much less works than
NCAR and NAR label noise. For example, Chikaraka and
McKeon [72] introduce the truncated and the exponential label
noise models which are detailed in Section III-A3 and where
the probability of mislabelling depends on the distance to the
classification boundary. A special case of truncated label noise
is studied in [70]. In [81], two features are randomly picked
and the probability of mislabelling depends on which quadrant
(with respect to the two selected features) the sample belongs to.

In practice, it would be very interesting to obtain more real-world datasets where mislabelled instances are clearly identified. Also, an important open research problem is to find what the characteristics of real-world label noise are. Indeed, it is not yet clear in the literature if and when NCAR, NAR or NNAR label noise is the most realistic.

**B. Validation and Test of Algorithms in the Presence of Label Noise**

An important issue for methods which deal with label noise is to prove their efficiency. Depending on the consequence of label noise which is targeted, different criteria can be used. In general, a good method must either 1) maintain the value of the quality criterion when label noise is introduced or 2) improve the value of the criterion with respect to other methods in the presence of label noise. In the literature, most experiments assess the efficiency of methods to take care of label noise in terms of accuracy (see e.g. [46], [69], [138], [160], [161], [171], [180], [184]), since a decrease in accuracy is one of the main consequences of label noise, as discussed in Section III-A.

Another common criterion is the model complexity [46], [138], [184], e.g. the number of nodes for decision trees or the number of rules in inductive logic. Indeed, as discussed in Section III-B, some inference algorithms tend to overfit in the presence of label noise, what results in overly complex models. Less complex models are considered better, since they are less prone to overfitting.

In some contexts, the estimated parameters of the models themselves can also be important, as discussed in Section III-C. Several works focus on the estimation of true frequencies from observed frequencies [4], [122], [123], [126], what is important e.g. in disease prevalence estimation.

Eventually, in the case of data cleansing methods, one can also investigate the filter precision. In other words, do the removed instances actually correspond to mislabelled instances and conversely? Different measures are used in the literature, which can be explained using Fig. 4 inspired by [46], [138]. In [46], [69], [180], [184], [318], two types of errors are distinguished. Type 1 errors are correctly labelled instances which are erroneously removed. The corresponding measure is

$$\text{ER}_1 = \frac{\text{# of correctly labelled instances which are removed}}{\text{# of correctly labelled instances}}.$$  \hspace{1cm} (19)

Type 2 errors are mislabelled instances which are not removed. The corresponding measure is

$$\text{ER}_2 = \frac{\text{# of mislabelled instances which are not removed}}{\text{# of mislabelled instances}}.$$  \hspace{1cm} (20)

The percentage of removed samples which are actually mislabelled is also computed in [46], [69], [180], [183], [184], [318], what is given by the noise elimination precision

$$\text{NEP} = \frac{\text{# of mislabelled instances which are removed}}{\text{# of removed instances}}.$$  \hspace{1cm} (21)

A good data cleansing method must find a compromise between ER$_1$, ER$_2$ and NEP [46], [69], [180], [184]. On the one hand, conservative filters remove few instances and are therefore precise (ER$_1$ is small and NEP is large), but they tend to keep most mislabelled instances (ER$_2$ is large). Hence, classifiers learnt with data cleansed by such filters achieve low accuracies. On the other hand, aggressive filters remove more mislabelled instances (ER$_2$ is small) in order to increase the classification accuracy, but they also tend to remove too many instances (ER$_1$ is large and NEP is small). Notice that Verbaeten and Van Assche [184] also compute the percentage of mislabelled instances in the cleansed training set.

Notice that a problem which is seldom mentioned in the literature is that model validation can be difficult in the presence of label noise. Indeed, since validation data are also polluted by label noise, methods like e.g. cross-validation or bootstrap may poorly estimate generalisation errors and choose meta-parameters which are not optimal (with respect to clean data). For example, the choice of the regularisation constant in regularised logistic regression will probably be affected by the presence of mislabelled instances far from the classification boundary. We think that this is an important open research question.

**IX. Conclusion**

This survey shows that label noise is a complex phenomenon with many potential consequences. Moreover, there exist many different techniques to address label noise, which can be classified as label noise-robust methods, label noise cleansing methods or label noise-tolerant methods. As discussed in Section VII-A1, an identification problem occurs in practical inference: mislabelled instances are difficult to distinguish from correctly labelled instances. In fact, without additional information beyond the main data, it is not possible to take into account the effect of mislabelling [84]. A solution is to make assumptions which allow selecting a compromise between naively using instances as they are and seeing any instance as possibly mislabelled.

All methods described in this survey can be interpreted as making particular assumptions. Firstly, in label noise-
robust methods described in Section V, overfitting avoidance is assumed to be sufficient to deal with label noise. In other words, mislabelled instances are assumed to cause overfitting in the same way as any other instance would. Secondly, in data cleansing methods presented in Section VI, different heuristics are used to distinguish mislabelled instances from exceptions. Each heuristic is in fact a definition of what is label noise. Thirdly, label noise-tolerant methods described in Section VII impose different constraint using e.g. Bayesian priors or structural constraints (i.e. in generative methods) or attempt to make existing methods less sensitive to the consequences of label noise.

In conclusion, the machine learning practitioner has to choose the method whose definition of label noise seems more relevant in his particular field of application. For example, if experts can provide prior knowledge about the values of the parameters or the shape of the conditional distributions, probabilistic methods should be used. On the other hand, if label noise is only marginal, label noise-robust methods could be sufficient. Eventually, most data cleansing methods are easy to implement and have been shown to be efficient and to be good candidates in many situations. Moreover, underlying heuristics are usually intuitive and easy-to-interpret, even for the non-specialist who can look at removed instances.

There are many open research questions related to label noise and many avenues remain to be explored. For example, to the best of our knowledge, the method in [56] has not been generalised to other label noise cleansing methods. Hughes et al. delete the label of the instances (and not the instances themselves) whose labels are less reliable and perform semi-supervised learning using both the labelled and the (newly) unlabelled instances. This approach has the advantage of not altering the distribution of the instances and it could be interesting to investigate whether this improve the results with respect to simply removing suspicious instances. Also, it would be very interesting to obtain more real-world datasets where mislabelled instances are clearly identified, since there exist only a few such datasets [53], [127], [245], [315], [316].

It is also important to find what the characteristics of real-world label noise are, since it is not yet clear if and when NCAR, NAR or NNAR label noise is the most realistic. Answering this question could lead to more complex and realistic models of label noise in the line of e.g. [5], [56], [67], [70]–[72], [90], [91], [122], [227]–[230], [235], [251]. Label noise should be also be studied in more complex settings than standard classification, like e.g. image processing [242], [301], [302] and sequential data analysis [84], [253]. The problem of meta-parameter selection in the presence of label noise is also an important open research problem, since estimated error rates are also biased by label noise [112], [126], [127].

REFERENCES


W. An and M. Liang, “Fuzzy support vector machine based on within-class scatter for classification problems with outliers or noises,” Neurocomputing, in press.


Benoît Frénay received the Engineer’s degree from the Université catholique de Louvain (UCL), Belgium, in 2007. He is now Ph.D. student at the UCL Machine Learning Group. His main research interests in machine learning include support vector machines, extreme learning, graph-based models, classification, data clustering, probability density estimation, feature selection and label noise.


Michel Verleysen received the M.S. and Ph.D. degrees in electrical engineering from the Université catholique de Louvain (Belgium) in 1987 and 1992, respectively. He is Full Professor at the Université catholique de Louvain, and Honorary Research Director of the Belgian F.N.R.S. (National Fund for Scientific Research). He is editor-in-chief of the Neural Processing Letters journal (published by Springer), chairman of the annual ESANN conference (European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning), past associate editor of the IEEE Trans. on Neural Networks journal, and member of the editorial board and program committee of several journals and conferences on neural networks and learning. He is author or co-author of more than 250 scientific papers in international journals and books or communications to conferences with reviewing committee. He is the co-author of the scientific popularization book on artificial neural networks in the series "Que Sais-Je?", in French, and of the "Nonlinear Dimensionality Reduction" book published by Springer in 2007. His research interests include machine learning, artificial neural networks, self-organization, time-series forecasting, nonlinear statistics, adaptive signal processing, and high-dimensional data analysis.