A Machine Learning Technique Resilient to Biased Labels: The Network-Flow Based Confidence HNC

author names withheld

Editor: Under Review

Abstract

The quality and quantity of labeled data is a major challenge to machine learning methods. Obtaining training data is expensive and time consuming, and the use of historical labeled data introduces biases and noise into the process. HNC (Hochbaum's Normalized Cut) is a semi-supervised classification method that relies on pairwise similarities between samples and, in that sense, is less reliant on labels. As a machine learning technique, HNC combines the effect of pairwise similarities between all samples, including the labeled samples, with constraining the labeled data to belong to the labeled class. In addition, HNC is solved, or determine a classification, using a very efficient parametric minimum cut procedure. HNC was shown to be competitive with many well-known classification methods. We introduce here *Confidence HNC*, which is HNC where the labeled samples belong to their label class with limited confidence level. A labeled sample can then be classified differently from its label if the effect of similarity to other samples, whether labeled or unlabeled, overrides the penalty of such misclassification. We present an extensive experimental study where we compare Confidence HNC with leading algorithms for coping with noisy labels, for both real and synthetic data sets. We evaluate the classification accuracy of the method as well as its capability to detect noise in comparison to a variant of k-nearest neighbor classifier, a deep learning method Co-teaching+ and a recent noise filtering technique Confident learning, demonstrating the dominant performance of Confidence HNC.

Keywords: Classification, Semi-supervised Learning, Label Noise, Network Flow, Hochbaum's Normalized Cut

1. Introduction

In the past few decades, the availability of labeled datasets affected improvements in making better decisions in various subject fields. Many machine learning algorithms now exhibit better performance partly due to this data availability. However, the quality of the final result of the machine learning pipeline is highly dependent on the data quality and in particular on the reliability of the labels. Label noise or historical bias in the labels is one of the concerning issues that has a tremendous impact on the outcome of learning methods and receives attention from researchers in the community.

We present here a classification method that is particularly resilient to corrupt labels. The method, HNC (Hochbaum's Normalized Cut), is a binary classification technique that utilizes pairwise similarities between samples. In HNC, samples are partitioned into two sets in a way that maximizes pairwise similarities in the same group while minimizing similarities between groups. The method leverages the unlabeled data by utilizing the pairwise relationship between labeled and unlabeled samples as well as those among the unlabeled samples. Baumann et al. (2019) showed, via an extensive experimental study, that HNC is competitive, and often improves, with leading classifiers.

© .

We devise here a variant of HNC called *Confidence HNC*, CHNC, to handle corrupt or noisy labels. This is done by relaxing the requirement that labeled samples fall in their labeled class, and instead assigning *confidence weights* that reflect the confidence in the reliability of the labels and serve as a penalty for a labeled sample to be placed in a class that does not match its label, and hence mislabeled. This reduces the effect of noisy labels and provides a detection mechanism for those labels. That is, if a labeled sample is mislabeled in spite of the penalty, it is deemed noisy.

We compare CHNC with three leading classification methods designed to deal with noisy labels: the *Nearest Neighbor Editing Aided by Unlabeled Data* that, like HNC, also utilizes pairwise similarity; the *Confident Learning* filtering method that is used with any classifier; and a deep learning method called *Co-teaching+*.

2. Overview of Classification Methods that Handle Noisy Labels

Classification methods that handle noisy labels have been categorized broadly into three classes, (Teng, 2001; Frénay and Verleysen, 2013): 1) methods that devised to avoid overfitting and are thus inherently robust to noise, 2) methods that employ filters, editing or preprocessors to detect noise and possibly relabel noisy samples, and 3) methods that model or reduce the influence of noise .

Methods in the first class were not devised specifically to handle noisy label but mainly to avoid overfitting. Consequently, they are not overly tuned to the noise (Teng, 2001). Examples of methods in this group include the C4.5 decision tree (Quinlan, 1987) and the decision tree model that uses the imprecise information gain as a split criterion (Abellán and Masegosa, 2012).

Methods in the second class use preprocessors to detect samples with noisy labels prior to removing or relabeling them. These methods are usually referred to as filtering or editing methods. Editing methods such as "editing nearest neighbors", "repeated edited nearest neighbors" and "All-kNN" (Tomek, 1976) are among the prominent filtering methods used early on in the k-nearest neighbor classifier. These methods remove labeled instances that are deemed incorrectly classified by a model trained on the labeled data only. A more recent work is Nearest Neighbor Editing Aided by Unlabeled Data (NNEAU), (Guan et al., 2009), in which unsupervised information from unlabeled data such as distances between unlabeled and labeled samples is leveraged to enhance the quality of noise filtering.

Another recent noise filtering and editing method is the Confident Learning framework (Northcutt et al., 2021). In this work, the joint distribution between noisy labels and true labels is estimated. Probabilistic thresholds and ranking method are then used to filter data. This method can be coupled with any learning method.

Methods in the third group are those that control the influence of samples that are likely noisy. The concept of fuzzy membership has been introduced to the support vector machine, usually referred to as Fuzzy SVM (Lin et al., 2004). For this SVM method, the likelihood that each sample's label is correct, or its "fuzzy membership", is computed. This likelihood is used as a weight in a part of the objective function that penalizes samples that are on the wrong side of the hyperplane. The choice of fuzzy membership function includes the kernel target alignment and the fraction of agreeing k-nearest neighbors (Lin et al., 2004). Another extension on Fuzzy SVM is to take into account the variance of feature vectors of samples in each class (An and Liang, 2013).

In the past decade many deep learning methods have been improved to become more robust to label noise or even to model noise. For example, Sukhbaatar et al. (2014) added a linear layer that functions as a transition matrix before the output layer to model the transition probability between

noisy and true labels. Goldberger and Ben-Reuven (2016) considered the dependence between label noise and features of samples and modeled it in the transition layer . One of the leading deep learning methods that exhibit robustness against label noise is Co-teaching+, which is a development of Co-teaching (Han et al., 2018; Yu et al., 2019). Co-teaching leverages the concept of memorization effect, which states that the model learns mostly from clean labels in the early epochs, resulting in small losses of good samples and high losses of bad samples. Two networks are trained based on the small-loss samples of the other network. Co-teaching+ incorporates the disagreement between the two networks by training only on small-loss samples where the two networks disagree.

An alternative way to leverage the memorization effect is to consider the network parameters rather than the samples (Xia et al., 2020). Network parameters are divided into critical and non-critical ones on the hypothesis that parameters whose loss gradients are large in early epochs are considered relevant to clean labels. These two sets of parameters are updated using different rules.

Confidence HNC is related to methods in the third group in the way that labeled samples exert different influences on the classification of unlabeled samples depending on how likely each of them is noisy. The resemblance between Confidence HNC and editing methods in the second group is the way we evaluate the quality of labels by training the base model on some labeled samples to predict pseudo-labels for other labeled samples. In our experiments, we select three leading methods to compare with Confidence HNC. These methods are NNEAU, Co-teaching+ and Confident Learning.

3. Hochbaum's Normalized Cut (HNC)

Hochbaum's Normalized Cut or HNC is an optimization model that has been shown to be effective in classification and clustering. We show here how HNC works as a method for binary classification as well as how it is extended to Confidence HNC (CHNC).

Given a data that contains both labeled and unlabeled samples, we construct a graph representation, $G = (V, E)$, that consists of a set of vertices V and a set of edges E. Each vertex in V corresponds to each data sample. Any two nodes, or samples, are connected via an edge whose weight is the pairwise similarity between the two samples. The objective of the Hochbaum's Normalized Cut or HNC (Hochbaum, 2010, 2013) is to partition V into two disjoint sets S and $\bar{S}$ to minimize the function:

$$\frac{C(S, \bar{S})}{C(S, S)} \tag{1}$$

$C(S, S) = \sum_{[i,j]\in E, i\in S, j\in S} w_{ij}$ is the total similarity weight within S, reflecting the coherence of S whereas $C(S, \bar{S}) = \sum_{[i,j]\in E, i\in S, j\in \bar{S}} w_{ij}$ is the similarity weight between the two sets of the partition, indicating how distinct S and $\bar{S}$ are. Hochbaum (2010) proved that problem (1) is equivalent to the minimization of the following objective function:

$$\frac{C(S, \bar{S})}{d(S)} \tag{2}$$

where $d(S) = \sum_{i\in S} d_i$ where $d_i = \sum_{[i,j]\in E, j\in V} w_{ij}$ is the weighted degree of i. The optimization problem (2) is closely related to the Normalized Cut (NC) problem introduced by Shi and Malik (2000), which minimizes $\frac{C(S,\bar{S})}{d(S)} + \frac{C(S,\bar{S})}{d(\bar{S})}$. Shi and Malik (2000) showed that the NC problem is NP-hard. However, the HNC problem is solvable in polynomial time (Hochbaum, 2010, 2013).

Sharon et al. (2006) mistakenly assumed that NC and HNC are equivalent and, therefore, HNC is NP-hard. Indeed, the main distinction between NC and HNC is that NC is symmetric with respect to the set S and its complement $\bar{S}$ whereas HNC optimizes the cluster properties for the set S only.

It has been further shown in Hochbaum (2010) that the minimization problem (2) is equivalent to finding the smallest positive λ such that the optimal objective function of the following linearized problem is non-positive.

$$\underset{\varnothing \subset S \subset V}{\text{minimize}} \; C(S, \bar{S}) - \lambda \sum_{i \in S} d_i. \tag{3}$$

In the linearized version of the problem (3), λ is interpreted as a tradeoff between the intra-similarity within S and the inter-similarity between S and $\bar{S}$. In order to guarantee that the set S is nonempty and not equal to V, there are "seed" nodes, one in S and the second in $\bar{S}$. In the context of binary classification, we designate the set S to be the positive class and $\bar{S}$ to be the negative class. Unlabeled samples that belong to the optimal set S are then predicted as positive whereas other unlabeled samples are predicted negative. The labeled samples are set to be seed nodes for the positive and negative classes that "supervise" the partition by forcing them to belong to S and $\bar{S}$ respectively,

Let the positive labeled set, the negative labeled set and the unlabeled set be denoted by L^+, L^-, and U, respectively. Then, the supervised form of (3) is,

$$\begin{aligned} \underset{\varnothing \subset S \subset V}{\text{minimize}} \; & C(S, \bar{S}) - \lambda \sum_{i \in S} d_i \\ \text{s.t. } & L^+ \subseteq S \text{ and } L^- \subseteq \bar{S} \end{aligned} \tag{4}$$

Depending on which class exhibits better intra-similarity, we may have the negative labeled samples be the seeds for the set S. In that case, $L^- \subseteq S$ and $L^+ \subseteq \bar{S}$. For a given value of λ, we solve the minimization problem (3) for the optimal S. We will discuss later how the value of λ is chosen.

It was shown in Hochbaum (2010, 2013) that problem (4) is solved as a minimum (s,t)-cut problem on an associated graph $G_{st}(\lambda)$, described below. The optimal source set S of a minimum (s,t)-cut of $G_{st}(\lambda)$ is an optimal solution of problem (4). Moreover, (4) is solved for *all* values of λ with a parametric minimum cut algorithm (Hochbaum, 2008) in the complexity of a single minimum cut procedure. The parametric cut procedure is applicable whenever the graph $G_{st}(\lambda)$ has the source adjacent capacities, as a function of the parameter λ, monotone non increasing and the sink adjacent capacities monotone non decreasing, or vise versa. Indeed, we make use of the efficient parametric minimum cut algorithm in tuning our implementation, see section 6.2.

The associated graph $G_{st}(\lambda)$ is illustrated in Figure 1. To construct the directed graph $G_{st}(\lambda)$ we replace each edge $[i,j] \in E$ by two directed arcs, (i,j) and (j,i) both carrying the capacity weight w_{ij}. A source node s and a sink node t are added to the set of nodes of G that consists of the positive and negative seed nodes L^+ and L^- and the nodes representing the unlabeled samples U. There are arcs of infinite capacity between s and the nodes of L^+, and arcs of infinite capacity between nodes of L^- and t. For each node $i \in U$, there is an arc (s,i) of capacity λd_i.

The minimum (s,t)-cut of $G_{st}(\lambda)$ partitions nodes into two disjoint sets: $S \cup \{s\}$, the *source set* and $\bar{S} \cup \{t\}$ the *sink set*. Unlabeled samples in S are then classified as positive and those in $\bar{S}$ are classified negative.

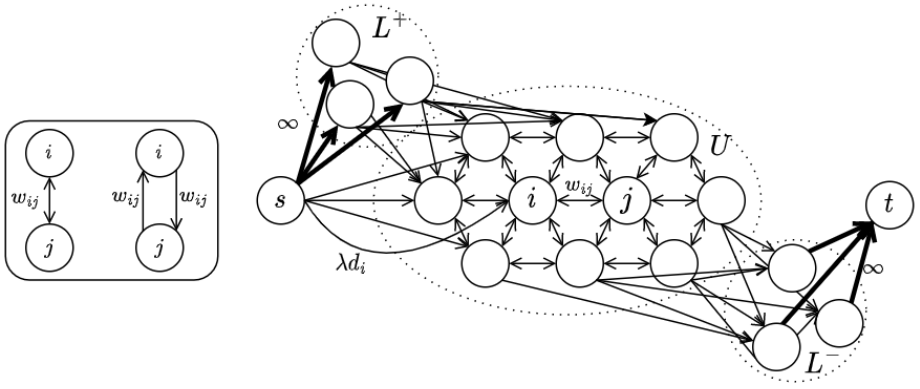

Figure 1: HNC graph or $G_{st}(\lambda)$: Nodes s and t are added to the data graph G along with additional arcs to form G_{st}. The minimum (s,t)-cut on $G_{st}(\lambda)$ provides a classification for the unlabeled samples. Arcs with arrows on both endpoints denote two directed arcs with equal weights as illustrated in the rounded box. Bold arcs are arcs with infinite weights.

4. Confidence HNC (CHNC)

The goal of the proposed method called Confidence HNC, or CHNC, is to enhance the resiliency of HNC to noisy labels. In the graph G_{st} of the HNC problem in Figure 1, positive samples in L^+ and negative samples in L^- are connected to s and t, respectively, with infinite weights on bold arcs. They are forced to be in the partition set of their given labels, either S or $\bar{S}$, in the optimal solution of HNC. However, their labels might be noisy. In CHNC, we replace infinite weights of arcs (bold arcs in Figure 1) that connect labeled samples to their respective source (s) or sink (t) nodes with finite *confidence weights*. The confidence weight c_i of a labeled sample i reflects how *confident* we are in the label quality of sample i. The classification results for the unlabeled samples can be obtained by solving for a minimum (s,t)-cut on the new graph in a similar way to HNC.

Note that by using CHNC, labeled samples are also classified simultaneously with the unlabeled samples. For a labeled sample with a low confidence weight, due to the absence of the infinite weights, it could be placed in the partition set that does not match its label if its similarity to samples with the opposite label is strong enough. Samples like this are considered noisy by the model. Hence, the set of samples that are considered noisy by CHNC can be written as $\{\bar{S} \cap L^+\} \cup \{S \cap L^-\}$ when S is designated for the positive class.

The method to compute the confidence weights is given in Subsection 5.3 of Section 5 where we explain our implementation of CHNC in detail.

5. Implementation of CHNC

In this section, we discuss the input graph G that represents our data to be used in CHNC.

5.1. Graph Sparsification

Regarding the connectivity of the nodes in the graph, we apply the k-nearest neighbor sparsification method by which we connect any two nodes, i and j, if i is among the k nearest neighbors of j, or j is among the k nearest neighbors of i. This technique was used in prior works that also model data as a graph such as Blum and Chawla (2001) and has been shown to yield competitive results compared to other graph construction techniques (Wang et al., 2013). We use $k = 15$ in this work.

5.2. Pairwise Similarities Computation

In terms of pairwise similarities that are used as edge weights, they depend on distances between samples. We use the Gaussian similarity weight, also known as the radial basis function, as the weight for each pair of samples i and j, or edge $[i, j]$, which is given as $w_{ij} = exp(-\frac{dist(i,j)}{2\varepsilon^2})$ where $dist(i, j)$ is the distance between samples i and j. Gaussian weight has been used in many works (Chen et al., 2009; Zhu and Goldberg, 2009; An and Liang, 2013). After a careful inspection, we choose the value of ε to be 1.

Regarding distance computation, we use a weighted Euclidean distance. Features are weighted according to the feature importances obtained from the random forest classifier. This weighted distance is a special case of the Mahalanobis distance when the covariance matrix is a diagonal matrix whose diagonal entries are feature importances. Each feature importance is computed based on the impurity reduction in the random forest model attributed to that feature. We incorporate this step into the distance computation so that this weighted distance function would provide a more accurate representation of how similar the samples are to one another.

5.3. Confidence Weights Computation

We propose a procedure that computes the *confidence weight* of each labeled sample in L using the original HNC. First, we partition the labeled samples $L = L^+ \cup L^-$ into M folds, denoted by $L_1, L_2, ..., L_M$. For each m in $\{1, 2, ..., M\}$, we apply the HNC method on L by using samples in $L \setminus L_m$ as labeled samples and samples in L_m as unlabeled samples. As a result, we obtain the label predictions of sample in L_m, which could be either similar to or different from their given labels.

In each iteration, all samples in L across M folds are treated as unlabeled samples once. We begin the next iteration by repartitioning L into M folds before repeating the same process. This process repeats T times. We denote the prediction of a labeled sample i at the iteration t by y_i^t and the label of i in the given data by y_i. The *confidence score* of the labeled sample i is computed as

$$\frac{\sum_{t=1}^{T} I(y_i^t = y_i)}{T}$$

where $I(\cdot)$ is the indicator function. The confidence score computation procedure is shown in Figure 2. In this work, we use $M = 5$ and $T = 4$.

We multiply the computed confidence scores by $\frac{1}{|E|}\sum_{[i,j]\in E} w_{ij}$, which is the average pairwise similarity weight, to get the *confidence weights* that are of the same magnitude as other weights in the graph. The computed confidence weights replace the infinite weights of G_{st} in Figure 1. The classification result of CHNC is obtained by solving for a minimum (s, t)-cut on the new graph, in a similar way to HNC.

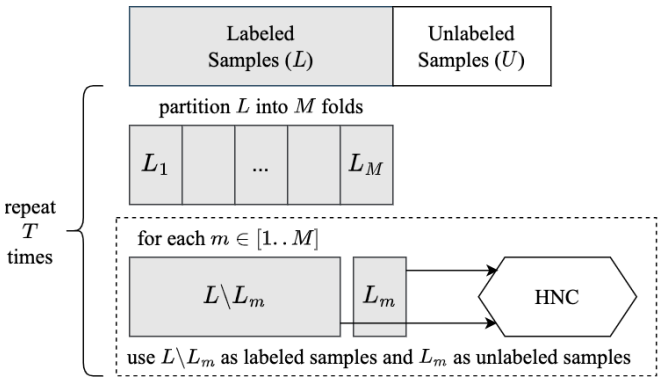

Figure 2: Confidence score computation: Each labeled sample is predicted using other labeled samples and HNC. Confidence score is equal to the proportion of correct predictions.

6. Experiments

To evaluate our model, we compare its classification performance as well as its noise detection capability with several other classification methods on both synthetic datasets and real datasets.

6.1. Baseline Methods

We compare Confidence HNC with three other classifiers that were developed to handle noisy labels. The first method is the Nearest Neighbor Editing Aided by Unlabeled Data (NNEAU) (Guan et al., 2009). The base classifiers used in the editing step of NNEAU are the k nearest neighbor classifier, the naive bayes classifier and the classification tree. The second method is called Co-teaching+ (CT+) (Yu et al., 2019). In this experiment, the training of CT+ is capped at 100 epochs. Early stopping is also included to terminate the training process when the training accuracy does not increase for 5 consecutive epochs. Third, we evaluate the Confident Learning method (CL) by Northcutt et al. (2021). It has been implemented as a Python package called Cleanlab. In this work, we use the k nearest neighbor classifier as the base classifier. We use the implementations of Co-teaching+ and Cleanlab from https://github.com/xingruiyu/coteaching_plus and https://docs.cleanlab.ai/stable/index.html#.

6.2. The hyperparameter for Confidence HNC

The tuning for CHNC involves the choice of the value of λ that delivers the best accuracy in the cross-validation step. We select 20 such candidate values. We also tune on whether s is designated for the positive class or the negative class. This can be done jointly in a single run of the parametric minimum cut with 40 settings for the source and sink adjacent capacities, as follows: The source adjacent capacities are the sequence λd_i for $\lambda \in \{1, 0.8, 0.6, ..., 0.0006, 0.0004, 0.0002\}$, and the corresponding sink adjacent capacities are 0. Then, the source adjacent capacities are 0, and the sink adjacent capacities are λd_i for $\lambda \in \{0.0002, 0.0004, 0.0006, ..., 0.6, 0.8, 1\}$. This sequence of 40 settings of source adjacent and sink adjacent capacities are monotone as required for the setup of the parametric cut network. Therefore, they all run in the complexity of a single minimum (s, t)-cut problem. We use the Pseudoflow Parametric algorithm, which is available at https://riot.

ieor.berkeley.edu/Applications/Pseudoflow/parametric.html, that allows us to tune our hyperparameter via stratified 5-fold cross validation efficiently.

6.3. Datasets

We test the classification methods on both synthetic datasets and real datasets. Synthetic datasets are generated using scikit-learn (Pedregosa et al., 2011). Real datasets are datasets from the UC Irvine Machine Learning Repository (Dua and Graff, 2017).

Regarding synthetic data, each dataset contains 1000 samples, of which each belongs to one of the two classes. Several parameters are involved in the generation of data points including the number of features, the proportion of samples assigned to each class, and the number of clusters per class. We vary their values as listed in Table 1, resulting in 20 configurations of these parameters. For each configuration, we generate 4 different datasets, adding up to 80 datasets.

In terms of real datasets, we experimented the methods on 8 datasets listed in Table 2.

Parameter	**Values**
# of features	10, 20
% of positive samples	30%, 40%, 50%, 60%, 70%
# of clusters per class	2, 4

Table 1: Synthetic Data Configurations

Name	**Size**	**# of Attr**	**%Pos**
Housing (HOU)	506	13	26.09
Energy Efficiency (ENB)	768	8	50.13
German Bank (GER)	1000	24	30.00
Maternal Health (MAT)	1014	6	59.96
Red Wine (WIN)	1599	11	53.47
Obesity (OBE)	2111	19	46.04
Cardiotocograms (CAR)	2126	21	77.85
Letter (LET)	20000	16	49.70

Table 2: Data from the UC Irvine Machine Learning repository

6.4. Experiment Settings

For each dataset, we partition data samples into labeled set (training set), which contains 80% of the samples, and unlabeled set (testing set), which contains the remaining 20%. Since we are evaluating the models' robustness to label noise, we add noise to the data by changing the labels of some labeled samples. Noise levels used in this experiment are 15% and 30% where $x\%$ noise means we change the labels of $x\%$ of samples in each class to the opposite label. Each learning model uses this corrupted labeled set to train the model and predict the labels of unlabeled samples.

To avoid bias due to the way the data is partitioned into labeled and unlabeled sets as well as the way the noise is added, we run 5 experiments for each dataset at each noise level. The difference between these 5 experiments comes from different data partition and label corruption.

6.5. Evaluation Metrics

First, we evaluate the classification performance of CHNC by comparing its classification accuracy to that of other models. For each synthetic data, we compute the relative accuracy improvement, shortened as Rel Acc Imp, of CHNC over each of the remaining models as follows: Rel Acc Imp of CHNC over model M is equal to $((\text{Accuracy of CHNC}/\text{Accuracy of } M) - 1) \times 100\%$. A positive relative accuracy improvement would imply that CHNC achieves a higher classification accuracy than the model that we are comparing it to, on a dataset. The improvement distribution over 400 experiments for each alternative method and each noise level is presented as a histogram plot.

For real datasets, where we run 5 experiments on each of them, we report the average accuracy of each model over the 5 runs as well as the standard deviation.

Second, we examine its noise detection capability by comparing its noise recall and noise precision to other models. Suppose the set of labeled samples determined by a model as noisy is D, and the set of noisy samples is N. Noise recall is computed as the fraction of noisy samples that are detected by the model, or $\frac{|N \cap D|}{|N|}$. Noise precision is the fraction of noisy samples among all samples that are considered noisy by the model, or $\frac{|N \cap D|}{|D|}$.

To test for the significance of the differences between Confidence HNC and other methods, we use the Wilcoxon signed-ranks test, which is commonly used in many works such as Abellán and Masegosa (2012); Sáez et al. (2016); Luengo et al. (2018). It is an appropriate statistical test when we compare the performances of two classifiers over multiple datasets, and when we do not assume that the differences between the scores of two classifiers are normally distributed (Demšar, 2006). The paired t-test, which is another common test but relies on several assumptions, is also used in the evaluation of this work and provides similar results as the Wilcoxon test.

7. Results

7.1. Results on Synthetic Data

With 80 synthetic datasets and 5 different runs on each of them, we obtain 400 classification accuracy scores for each classifier. We compute the relative accuracy improvement given by Confidence HNC compared with other classifiers for each of the 400 experiments as explained in Subsection 6.5. The histograms of the accuracy improvement at 15% and 30% noise levels are shown in Figures 3 and 4.

In each plot, two vertical lines are plot together with the histogram. The dashed line is at zero, indicating experiments where the two classifiers give similar performance (zero improvement). The area of the histogram on the right of this line indicates the proportion of datasets on which CHNC has better performance than the other model that is being compared with, which is when the relative accuracy improvement is positive. The solid line marks the average accuracy improvement. We see in Figure 3 and 4 that for both noise levels and for all three classifiers, the solid line lies to the right of the dashed line, implying that, the average accuracy improvement of CHNC over each classifier is positive in all cases. Compared with NNEAU, CT+ and CL, CHNC outperforms in $80.75\%, 88.75\%$ and 82% of the synthetic datasets at 15% noise level, and $80.25\%, 80.5\%$ and 62.5% at 30% noise level. These percentages are the areas of the histograms that lie to the right of their zero dashed lines, or the areas in orange.

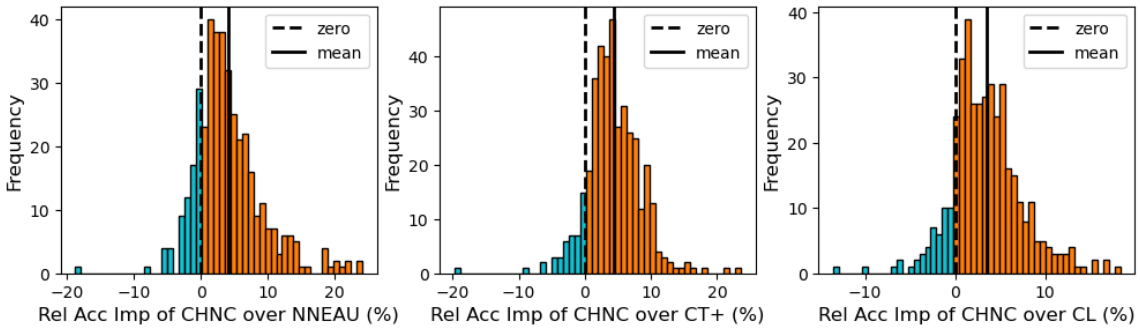

Figure 3: Histograms of accuracy improvement on 400 synthetic datasets yielded by Confidence HNC compared to other classifiers at the 15% noise level: The area to the right of the dashed line in each histogram indicates the number of synthetic datasets where CHNC outperforms the other model.

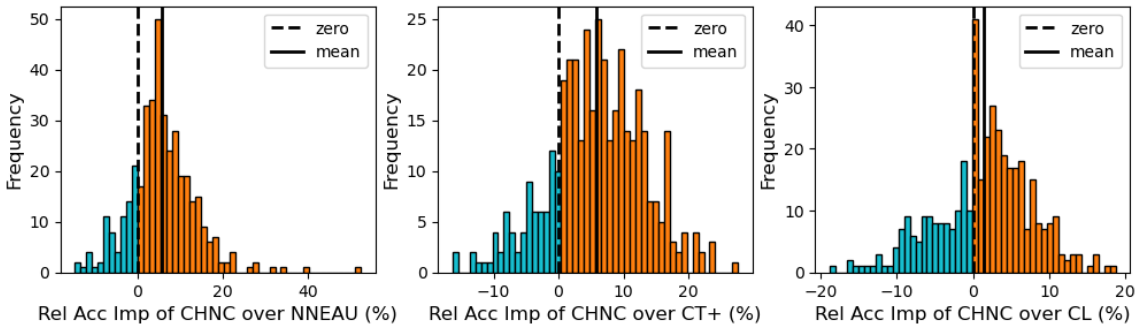

Figure 4: Histograms of accuracy improvement on 400 synthetic datasets yielded by Confidence HNC compared to other classifiers at the 30% noise level: The area to the right of the dashed line in each histogram indicates the number of synthetic datasets where CHNC outperforms the other model.

We apply the Wilcoxon test to evaluate the statistical significance of the difference between the accuracy scores of CHNC and those of other methods. P-values when tested with NNEAU, CT+ and CL are all smaller than 10^{-6} at both noise levels, validating the significance of the results.

Regarding the model's noise detection capability, CHNC yields high noise precision but only moderate noise recall. The model achieves significantly higher noise precision than all other classifiers for both noise levels. When it comes to noise recall, CHNC outperforms CT+ at both noise levels. However, it performs worse than CL at the noise level of 15% and worse than NNEAU at both noise levels. Hence, in terms of noise detection, noise precision is the strength of CHNC. Noise detection results are included in the appendix.

7.2. Results on Real Data

Classification accuracy improvements of CHNC over other classifiers on real data are reported in Table 3 and 4 for the 15% and 30% noise levels. Each entry in the table is the average accuracy over 5 runs along with the standard deviation.

At the 15% noise level, CHNC achieves the highest accuracy on four datasets. CT+ and NNEAU have the best performance on three and one datasets, respectively. There is one dataset, ENB, where all classifiers have the same performance at this noise level. For the data OBE, although CT+ has the highest accuracy, the accuracy given by CHNC is only slightly smaller, and it is higher than that of NNEAU and CL. The standard deviation of the accuracy of CHNC is in most cases close to that of the other models.

P-values given by the Wilcoxon test demonstrate that CHNC performs better than both NNEAU and CT+ with strong statistical significance.

Data	**CHNC**	**NNEAU**	**CT+**	**CL**
HOU	89.02 +/- 3.58	**90.2** +/- 1.96	88.24 +/- 1.75	89.02 +/- 1.69
ENB	97.79 +/- 1.06	97.79 +/- 1.06	97.79 +/- 1.06	97.79 +/- 1.06
GER	72.2 +/- 2.36	73.0 +/- 1.3	**74.5** +/- 1.7	72.9 +/- 2.35
MAT	**77.54** +/- 3.12	73.69 +/- 3.91	76.75 +/- 3.72	75.47 +/- 4.08
WIN	**72.12** +/- 0.85	69.56 +/- 1.39	**72.12** +/- 1.8	71.37 +/- 1.31
OBE	98.25 +/- 0.55	97.68 +/- 0.66	**98.77** +/- 0.53	95.93 +/- 0.46
CAR	**91.31** +/- 1.16	90.7 +/- 2.34	90.66 +/- 1.46	90.52 +/- 1.15
LET	**97.38** +/- 0.22	95.95 +/- 0.34	94.51 +/- 0.16	95.74 +/- 0.35
p-value		0.0027*	0.1893	0.0018*

Table 3: Classification accuracy (mean $\pm$ std err)$(\%)$ of all methods on real data at 15% noise level and p-values of the Wilcoxon test that tests for the statistical significance of the outperformance of CHNC over each alternative method. Boldface numbers are the highest accuracy for each dataset. P-value with (*) indicates strong statistical significance.

Data	**CHNC**	**NNEAU**	**CT+**	**CL**
HOU	83.33 +/- 2.56	85.88 +/- 3.14	84.51 +/- 4.22	**87.84** +/- 3.2
ENB	97.53 +/- 1.12	94.94 +/- 1.56	92.08 +/- 3.75	**97.79** +/- 1.06
GER	**72.1** +/- 2.27	71.5 +/- 2.05	71.0 +/- 3.48	71.4 +/- 0.97
MAT	**70.94** +/- 4.7	65.52 +/- 6.02	68.28 +/- 6.37	69.36 +/- 5.05
WIN	69.38 +/- 2.62	66.12 +/- 2.57	66.88 +/- 1.49	**69.94** +/- 2.2
OBE	95.56 +/- 0.72	89.88 +/- 2.45	**95.6** +/- 1.14	93.85 +/- 0.88
CAR	88.03 +/- 0.8	82.49 +/- 1.63	86.06 +/- 4.05	**88.64** +/- 1.42
LET	**93.74** +/- 0.69	87.7 +/- 0.84	89.72 +/- 0.52	92.96 +/- 0.49
p-value		1.1e-5*	0.0011*	0.4875

Table 4: Classification accuracy (mean $\pm$ std err)$(\%)$ of all methods on real data at 30% noise level and p-values of the Wilcoxon test that tests for the statistical significance of the outperformance of CHNC over each alternative method. Boldface numbers are the highest accuracy for each dataset. P-value with (*) indicates strong statistical significance.

As the noise goes up to 30%, CHNC outperforms CT+ in most cases, and still outperforms NNEAU like when the noise is 15%. Among the four datasets where CL is dominant, the performance of CHNC is fairly close to that of CL on the datasets ENB, WIN and CAR. P-values given by the Wilcoxon test indicate that CHCN gives significantly higher accuracy than NNEAU and CT+.

Moreover, as the noise increases from 15% to 30%, the accuracy of CHNC on datasets such as ENB and GER does not decrease as much as that of some other methods, demonstrating strong robustness against label noise of CHNC on these datasets.

In terms of noise detection, the results on real data are similar to the results on synthetic data. CHNC attains the highest noise precision on almost all datasets. Regarding noise recall, it exhibits only moderate performance. Again, this highlights the stength of CHNC in attaining high noise precision, implying that CHNC does not wrongly identify good samples as noisy as much as other methods. Results on real data regarding noise detection are included in the appendix.

8. Conclusions

We introduce here a new classification method, Confidence Hochbaum's Normalized Cut (CHNC), that is robust to label noise. CHNC varies the model's reliance on the labels of labeled data through the use of confidence weights. Moreover, CHNC allows labeled samples to take on a label that is different from its given label in the graph partition. This works as a noise detection procedure.

From the experiments on both real and synthetic data, we found that CHNC demonstrates competitive results in terms of classification accuracy and noise precision. In terms of these metrics, CHNC outperforms other methods significantly at both 15% and 30% noise levels on synthetic data. On real data, at a lower noise level, it outperforms NNEAU and CL significantly while remains competitive with CT+. At a higher noise level, CHNC outperforms NNEAU and CT+ significantly and performs as well as CL. At both noise levels, it achieves significantly higher noise precision than all three methods. These results substantiate the competency of CHNC in the presence of label noise.

When interpreting results on real data, it is important to note that these real data might already contain noise even before we corrupt them. On the other hand, we know the ground truth of synthetic data with certainty. The interpretation of results on real data is obscured by this problem.

For future work, one direction to explore is the robustness of CHNC or other variants of HNC to asymmetric label noise. In this work, we introduce a symmetric noise corruption to the data with the same percentages of corrupt samples in each class. There are practical situation though in which a certain class might be more susceptible to noise corruption then others. Another direction to investigate is how CHNC can be devised to deal with the limitation on labeled samples quantity. Since CHNC is a semi-supervised learning method which utilizes unsupervised information in the form of similarities between samples, it is particularly suitable for classification scenarios with limited availability of labeled samples.

References

Joaquín Abellán and Andrés R Masegosa. Bagging schemes on the presence of class noise in classification. *Expert Systems with Applications*, 39(8):6827–6837, 2012.

Wenjuan An and Mangui Liang. Fuzzy support vector machine based on within-class scatter for classification problems with outliers or noises. *Neurocomputing*, 110:101–110, 2013.

Philipp Baumann, Dorit S Hochbaum, and Yan T Yang. A comparative study of the leading machine learning techniques and two new optimization algorithms. *European journal of operational research*, 272(3):1041–1057, 2019.

Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. 2001.

Yihua Chen, Eric K Garcia, Maya R Gupta, Ali Rahimi, and Luca Cazzanti. Similarity-based classification: Concepts and algorithms. *Journal of Machine Learning Research*, 10(3), 2009.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30, 2006.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2013.

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. 2016.

Donghai Guan, Weiwei Yuan, Young-Koo Lee, and Sungyoung Lee. Nearest neighbor editing aided by unlabeled data. *Information Sciences*, 179(13):2273–2282, 2009.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

Dorit S Hochbaum. The pseudoflow algorithm: A new algorithm for the maximum-flow problem. *Operations research*, 56(4):992–1009, 2008.

Dorit S Hochbaum. Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):889–898, 2010.

Dorit S Hochbaum. A polynomial time algorithm for rayleigh ratio on discrete variables: Replacing spectral techniques for expander ratio, normalized cut, and cheeger constant. *Operations Research*, 61(1):184–198, 2013.

Chun-fu Lin et al. Training algorithms for fuzzy support vector machines with noisy data. *Pattern recognition letters*, 25(14):1647–1656, 2004.

Julián Luengo, Seong-O Shim, Saleh Alshomrani, Abdulrahman Altalhi, and Francisco Herrera. Cnc-nos: Class noise cleaning by ensemble filtering and noise scoring. *Knowledge-Based Systems*, 140:27–49, 2018.

Curtis Northcutt, Lu Jiang, and Isaac Chuang. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, 2021.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

J. Ross Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27(3): 221–234, 1987.

José A Sáez, Mikel Galar, Julián Luengo, and Francisco Herrera. Inffc: An iterative class noise filter based on the fusion of classifiers with noise sensitivity control. *Information Fusion*, 27: 19–32, 2016.

Eitan Sharon, Meirav Galun, Dahlia Sharon, Ronen Basri, and Achi Brandt. Hierarchy and adaptivity in segmenting visual scenes. *Nature*, 442(7104):810–813, 2006.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

Choh-Man Teng. A comparison of noise handling techniques. In *FLAIRS Conference*, pages 269–273, 2001.

Ivan Tomek. An experiment with the edited nearest-nieghbor rule. 1976.

Jun Wang, Tony Jebara, and Shih-Fu Chang. Semi-supervised learning using greedy max-cut. *The Journal of Machine Learning Research*, 14(1):771–800, 2013.

Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. Robust early-learning: Hindering the memorization of noisy labels. In *International conference on learning representations*, 2020.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR, 2019.

Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

Appendix A. Additional Results

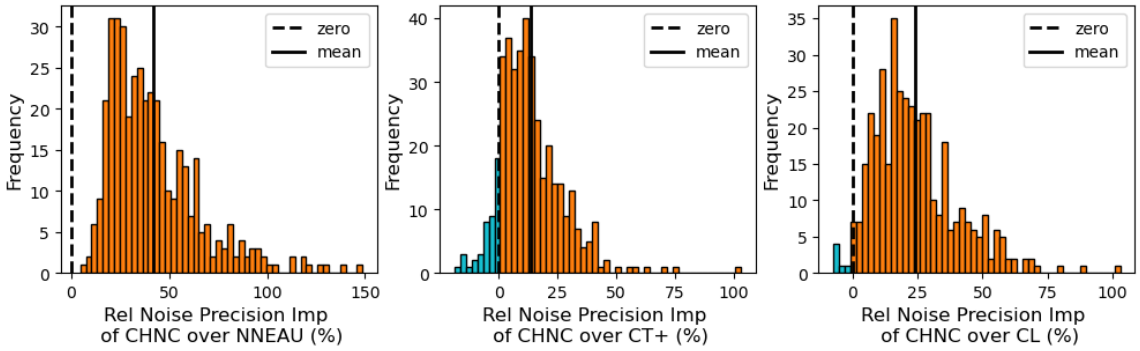

Figure 5: Histograms of noise precision improvement on synthetic datasets yielded by CHNC compared to other classifiers at the 15% noise level

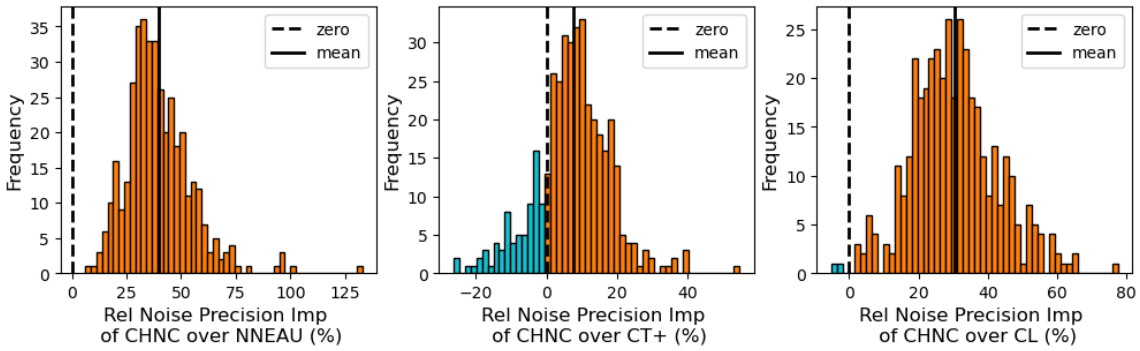

Figure 6: Histograms of noise precision improvement on synthetic datasets yielded by CHNC compared to other classifiers at the 30% noise level

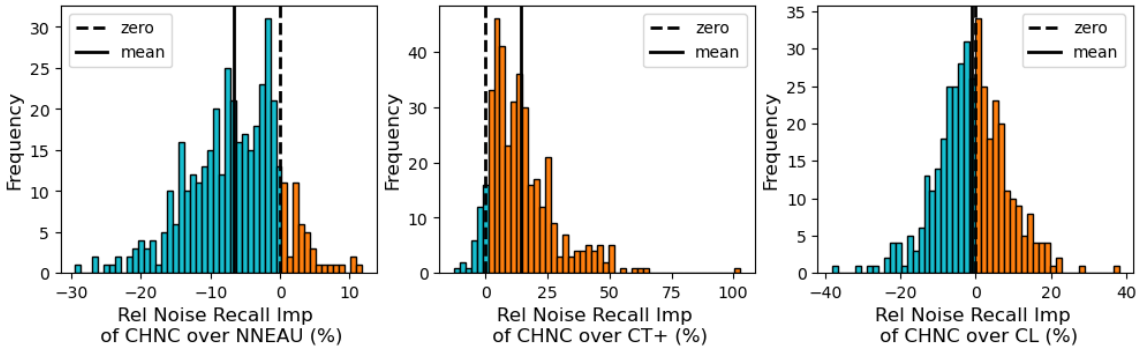

Figure 7: Histograms of noise recall improvement on synthetic datasets yielded by CHNC compared to other classifiers at the 15% noise level

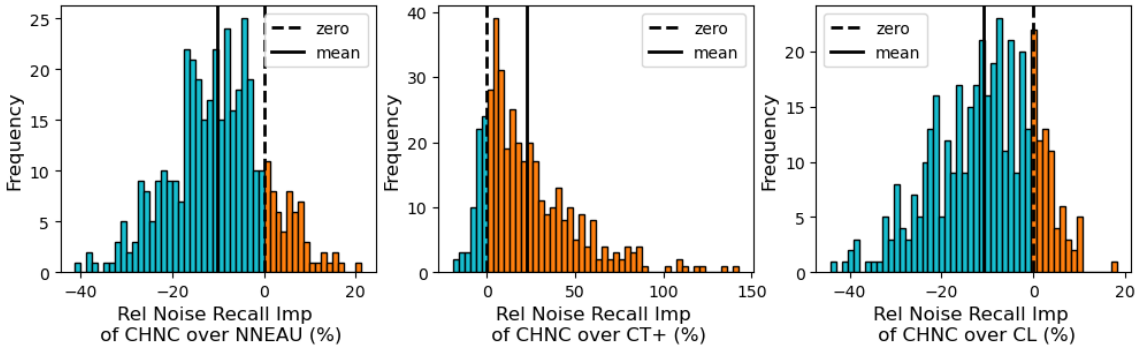

Figure 8: Histograms of noise recall improvement on synthetic datasets yielded by CHNC compared to other classifiers at the 30% noise level

Data	CHNC	NNEAU	CT+	CL
HOU	**69.67** +/- 3.13	59.75 +/- 1.86	66.48 +/- 1.51	68.75 +/- 2.49
ENB	90.2 +/- 0.94	86.48 +/- 2.46	90.2 +/- 0.94	**90.8** +/- 1.39
GER	**41.24** +/- 2.57	31.98 +/- 2.02	40.66 +/- 1.89	38.26 +/- 2.26
MAT	**41.69** +/- 1.35	35.95 +/- 1.72	35.47 +/- 1.65	37.21 +/- 1.52
WIN	**39.49** +/- 1.54	31.7 +/- 1.14	35.81 +/- 2.48	32.66 +/- 1.29
OBE	96.68 +/- 2.31	87.43 +/- 2.68	**97.94** +/- 1.04	88.17 +/- 2.94
CAR	**76.64** +/- 2.27	65.31 +/- 1.68	66.26 +/- 2.59	72.57 +/- 0.65
LET	**94.75** +/- 0.53	80.6 +/- 1.15	84.49 +/- 0.74	85.79 +/- 1.64
p-value		9.09e-13*	7.11e-6*	4.90e-7*

Table 5: Noise precision (mean $\pm$ std err)(%) on real data at 15% noise level and p-values of the Wilcoxon test that tests for the statistical significance of the outperformance of CHNC over each alternative method. Boldface numbers are the highest accuracy for each dataset. P-value with (*) indicates strong statistical significance.

Data	CHNC	NNEAU	CT+	CL
HOU	**79.79** +/- 3.97	70.32 +/- 3.31	76.98 +/- 7.27	70.59 +/- 2.9
ENB	**95.7** +/- 0.92	78.96 +/- 1.78	88.51 +/- 7.65	85.06 +/- 3.19
GER	58.76 +/- 4.83	49.06 +/- 1.31	**61.93** +/- 6.5	46.84 +/- 2.05
MAT	**59.34** +/- 2.91	45.65 +/- 3.6	47.9 +/- 5.17	49.29 +/- 2.29
WIN	**56.64** +/- 2.57	44.19 +/- 1.25	50.36 +/- 3.15	45.27 +/- 1.52
OBE	**95.85** +/- 1.15	73.29 +/- 3.73	92.9 +/- 3.43	78.84 +/- 2.64
CAR	**85.21** +/- 2.51	62.13 +/- 3.52	75.8 +/- 7.69	73.54 +/- 2.66
LET	**94.43** +/- 0.75	71.23 +/- 0.64	84.29 +/- 1.3	77.38 +/- 0.5
p-value		9.09e-13*	1.70e-5*	9.9e-13*

Table 6: Noise precision (mean $\pm$ std err)(%) on real data at 30% noise level and p-values of the Wilcoxon test that tests for the statistical significance of the outperformance of CHNC over each alternative method. Boldface numbers are the highest accuracy for each dataset. P-value with (*) indicates strong statistical significance.

Data	CHNC	NNEAU	CT+	CL
HOU	86.23 +/- 1.67	**88.52** +/- 4.15	84.59 +/- 3.04	81.64 +/- 9.12
ENB	98.04 +/- 0.81	**98.26** +/- 0.87	98.04 +/- 0.81	95.87 +/- 3.46
GER	61.33 +/- 3.67	**72.5** +/- 2.98	62.5 +/- 1.75	61.5 +/- 3.27
MAT	67.05 +/- 2.86	66.89 +/- 5.33	72.79 +/- 3.53	**73.11** +/- 4.29
WIN	59.17 +/- 1.02	65.94 +/- 3.71	73.33 +/- 1.52	69.17 +/- 3.04
OBE	95.51 +/- 1.82	94.8 +/- 1.79	**96.77** +/- 0.76	93.86 +/- 2.26
CAR	87.29 +/- 2.11	86.98 +/- 3.26	**88.08** +/- 2.02	84.63 +/- 2.36
LET	**92.91** +/- 0.67	92.18 +/- 0.66	89.28 +/- 0.59	92.64 +/- 0.39
p-value		0.9904	0.9356	0.7651

Table 7: Noise recall (mean $\pm$ std err)($\%$) on real data at 15% noise level and p-values of the Wilcoxon test that tests for the statistical significance of the outperformance of CHNC over each alternative method. Boldface numbers are the highest accuracy for each dataset. P-value with (*) indicates strong statistical significance.

Data	CHNC	NNEAU	CT+	CL
HOU	78.03 +/- 1.59	**82.62** +/- 3.0	84.43 +/- 3.4	74.59 +/- 3.63
ENB	**94.24** +/- 2.24	87.17 +/- 2.63	90.33 +/- 5.85	93.37 +/- 2.6
GER	59.92 +/- 4.61	**71.33** +/- 4.57	54.33 +/- 7.14	53.83 +/- 3.42
MAT	55.9 +/- 5.16	56.39 +/- 1.69	65.08 +/- 4.8	**70.08** +/- 2.79
WIN	50.65 +/- 3.7	60.78 +/- 1.92	67.31 +/- 2.12	**67.47** +/- 1.58
OBE	86.28 +/- 2.95	82.37 +/- 4.75	**90.99** +/- 2.45	90.83 +/- 0.89
CAR	79.06 +/- 2.85	69.76 +/- 5.22	**82.47** +/- 5.26	77.73 +/- 1.62
LET	83.44 +/- 0.88	79.09 +/- 0.97	84.31 +/- 0.72	**88.8** +/- 0.48
p-value		0.6045	0.9983	0.9929

Table 8: Noise recall (mean $\pm$ std err)($\%$) on real data at 30% noise level and p-values of the Wilcoxon test that tests for the statistical significance of the outperformance of CHNC over each alternative method. Boldface numbers are the highest accuracy for each dataset. P-value with (*) indicates strong statistical significance.