Confidence HNC : A Network-Flow Based Classifier Resilient to Label Noise

Tor Nitayanont[*] Dorit S. Hochbaum[†]

Abstract

The quality of labeled data is a major challenge to machine learning methods. Obtaining high-quality training data is costly, and the use of historical labeled data can introduce noise. HNC (Hochbaum's Normalized Cut) is a binary classification method that identifies a homogeneous cluster, maximizing pairwise similarities between samples within the cluster, whether labeled or not, while minimizing the similarity between the cluster and its complement. This is done by applying an efficient minimum cut procedure. As a supervised technique, the labeled samples are preassigned either to the cluster or its complement. It was shown to be competitive with many classification methods. We introduce an extension of this method called *Confidence HNC* or CHNC to cope with training labels that might be noisy or biased. In this case, the labeled samples are assumed to belong to their label classes with limited confidence levels. CHNC permits labeled samples to 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 experimental study comparing CHNC with leading algorithms for coping with noisy labels, on both real and synthetic data, and show that CHNC achieves better performance in terms of accuracy and noise precision.

1 Introduction

The performance of machine learning models depend to a great extent on the data quality and, in particular, the reliability of the labels. Label noise 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.

To combat the issue of label noise, we inspect semi-supervised learning methods, which utilize additional information from unlabeled data on top of labeled data that traditional supervised methods have access to. While semi-supervised methods can come in handy when labeled data is scarce or costly [34], we believe that another advantage that this class of methods provides is how the effect of possibly noisy labeled data is counterbalanced by that of unlabeled data, making them suitable for learning with label noise.

A particular class of semi-supervised methods that we are interested in is the class of network-flow based, or graph based, methods in which minimum cut solution of a graph representation of the data provides label prediction of unlabeled samples. Unlabeled samples assist the method through their connectivity with labeled samples, as well as that among themselves. Examples of methods in this class are [3], [4], [14]. While these methods already control the effect of the labeled samples, in this work, we regulate their influence further by modifying the graph representation. We incorporate this modification into a network-flow based method called Hochbaum's Normalized Cut (HNC) [14].

In HNC, samples are partitioned into two sets in a way that maximizes pairwise similarities in the same group while minimizing similarities between groups by solving a single minimum cut problem on a respective graph. It was shown in [2], via an extensive experimental study, that HNC is competitive and improves on leading classifiers.

The variant of HNC that we devise here to handle noisy labels is called *Confidence HNC* (CHNC). Instead of requiring that labeled samples fall in their respective labeled class, we assign *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. 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 classification methods designed to deal with noisy labels: the *Nearest Neighbor Editing Aided by Unlabeled Data* [9] that utilizes pairwise similarity like HNC; the *Confident Learning* [23] filtering method that can be coupled with any classifier; and a deep learning method called *Co-teaching+* [33]. These methods are selected as baselines

[*] Industrial Engineering and Operations Research Department, University of California, Berkeley. tor_n@berkeley.edu

[†] Industrial Engineering and Operations Research Department, University of California, Berkeley. hochbaum@ieor.berkeley.edu

in this work as they have been shown to be robust to varying levels of label noise.

2 Related works

This section overviews classification methods that handle noisy labels. Methods that we mention here either (i) detect noisy samples by using the model trained with other labeled samples, or (ii) alter the influence of each labeled sample. Our method, Confidence HNC, shares these ideas with the works mentioned below.

Preprocessing methods: Editing methods such as Editing nearest neighbors and Repeated edited nearest neighbors [30] are among the first filtering methods used with the kNN classifier. These methods remove labeled instances that are incorrectly classified by a model trained on the labeled data. Another work is Nearest Neighbor Editing Aided by Unlabeled Data (NNEAU) [9], where the training set is augmented by applying co-training on the unlabeled set before filtering. A more recent method is Confident Learning by [23] in which the joint distribution between noisy labels and true labels is estimated. Probabilistic thresholds and ranking method are then used to filter data. There are also other works such as [16, 21, 22, 24] that use thresholds on certain metrics to filter labeled samples.

Methods that control the influence of labeled samples: To limit the impact of noisy samples, the concept of fuzzy membership is introduced to the SVM method, called Fuzzy SVM [18]. The likelihood that each label is correct is computed and used to scale the penalty weight in the objective function. Many deep learning methods also adjust the impact of labeled samples depending on how likely they are noisy. Co-teaching [10] 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. Hence, two networks are trained on the small-loss samples of the other network. Co-teaching+ [33] further incorporates the disagreement between the two networks by training only on small-loss samples where the two networks disagree. Other works include [19, 20, 29].

Other deep learning methods Aside from Co-teaching+, there are also other prominent methods such as DivideMix [17]. However, many of them have been applied exclusively on image data. Their implementations that are available online are also tailored for image data specifically. Hence, we do not include them in our experiments as our focus is on tabular data.

Confidence HNC is similar to filtering methods in the way that it evaluates the quality of labeled samples based on the prediction from the model trained on labeled samples. It also allows labeled samples to exert different influences depending on their estimated quality, similar to Fuzzy SVM and Co-teaching. In the experiments, we compare Confidence HNC with NNEAU, Co-teaching+ and Confident Learning.

3 Problem Statement and Notation

Given a data that consists of labeled samples $L = \{x_i, y_i\}_{i=1}^{|L|}$, where y_i indicates the class or label of sample i, and unlabeled samples $U = \{x_i\}_{i=|L|+1}^{|L|+|U|}$, our goal is to predict the labels of unlabeled samples.

The labeled set L consists of the positive set L^+ and the negative set L^-. In the context of this work, some labels in L could possibly be corrupted. Some positive samples might have their labels recorded as negative and are incorrectly included in the set L^-, and vice versa.

Let each sample be represented by a node in a graph, $G = (V, E)$, where $V = L \cup U$ and E is a set of edges connecting pairs of nodes in V with similarity weight w_{ij} for $[i, j] \in E$.

For two subsets of nodes $A, B \subseteq V$, we denote $C(A, B) = \sum_{[i,j]\in E, i\in A, j\in B} w_{ij}$. That is, the sum of edge weights that have one endpoint in A and the other in B. With this notation, $C(B, B)$ is the total sum of similarities within the set B, which is desired to be maximized, and $C(B, \bar{B})$ is the total similarity between B and its complement $\bar{B}$, which is aimed to be minimized.

In the next section we formally introduce Confidence HNC. Firstly, in Subsection 4.1, we provide a summary of HNC. Then, in Subsection 4.2, we describe the modification, CHNC, that handles noisy labels.

4 Confidence HNC

4.1 Hochbaum's Normalized Cut (HNC)

HNC selects a cluster of samples that balances two different objectives: maximizing the sum of similarities within the cluster, and minimizing the similarity between elements of the cluster and its complement. With the notation above, one way of combining these two objectives is to minimize their ratio

$$\min_{\emptyset \subset S \subset V} \frac{C(S, \bar{S})}{C(S, S)} \tag{4.1}$$

This problem was mistakenly assumed in [27] to be equivalent to the NP-hard Normalized Cut (NC) problem introduced by [28]. However, problem (4.1) and several variants of the problem were shown in [14, 15] to be solvable in polynomial time by solving a minimum s,t-cut problem on an associated graph.

For $d_i = \sum_{[i,j]\in E, j\in V} w_{ij}$, the *weighted degree* of i, and $d(S) = \sum_{i\in S} d_i$, the sum of weighted degrees of nodes in S or the *volume* of S, problem (4.1) was proved in [14] to be equivalent to

$$\min_{\emptyset \subset S \subset V} \frac{C(S,\bar{S})}{d(S)} \tag{4.2}$$

This ratio problem can be solved by "linearizing" it,

$$\underset{\varnothing \subset S \subset V}{\text{minimize}} \; C(S,\bar{S}) - \lambda \sum_{i \in S} d_i \tag{4.3}$$

and finding the smallest positive λ for which the optimal objective function of the linearized problem (4.3) is non-positive. This was shown in [14] to be solved with a parametric cut procedure in the complexity of a single minimum cut procedure. The linearized problem (4.3) is another form of trading off the two objectives.

We refer to these three versions of the same problem, (4.1), (4.2) and (4.3), as HNC.

In the context of binary classification, we designate the set S for the positive class and $\bar{S}$ for the negative class. We use the positive and negative labeled samples as seed nodes to *supervise* the partition by forcing them to belong to S and $\bar{S}$, respectively. That is, we require L^+ in S and L^- in $\bar{S}$.

The supervised form of (4.3) is then written as

$$\underset{L^+ \subseteq S \subseteq V \setminus L^-}{\text{minimize}} \; C(S,\bar{S}) - \lambda \sum_{i \in S} d_i \tag{4.4}$$

Once the optimal S^* is solved, unlabeled samples that belong to the optimal set S^* (or $S^* \cap U$) are predicted as positive whereas other unlabeled samples are predicted negative. This form of the problem has been used for various machine learning tasks under the label "supervised normalized cut" [2, 32].

(Note that we may designate the set S for the negative class instead, and require $L^- \in S$ and $L^+ \in \bar{S}$, depending on which class exhibits better intra-similarity. This designation is a tunable parameter.)

Lemma 4.1. *Problem (4.4) can be rewritten as*

$$\underset{L^+ \subseteq S \subseteq V \setminus L^-}{\text{minimize}} \; C(S,\bar{S}) - \lambda \sum_{i \in S \cap U} d_i \tag{4.5}$$

Proof. S consists of $S \cap U$ and L^+, which are disjoint. Thus, $\sum_{i \in S} d_i = \sum_{i \in S \cap U} d_i + \sum_{i \in L^+} d_i$. As L^+ is given, $\sum_{i \in L^+} d_i$ is a constant. Hence, minimizing $-\lambda \sum_{i \in S} d_i$ is equivalent to minimizing $-\lambda \sum_{i \in S \cap U} d_i$. □

Problems (4.3) and (4.4) are solvable, in polynomial time, as a minimum s,t-cut problem on an associated graph $G_{st}(\lambda)$, described in the next paragraph. The minimum cut algorithm for these problems is immediately implied since these problems are monotone IP3 problems, [11, 12], and any such problem is solved with a minimum s,t-cut procedure on the associated graph. A *Monotone IP3* problems are integer programming problems on at most 3 variables per constraint, where two of the variables appear with opposite coefficients, and a third variable, if included, can appear in at most one constraint. The coefficient of the third variable in the objective function must be non-negative for minimization problems, or non-positive for maximization problems.

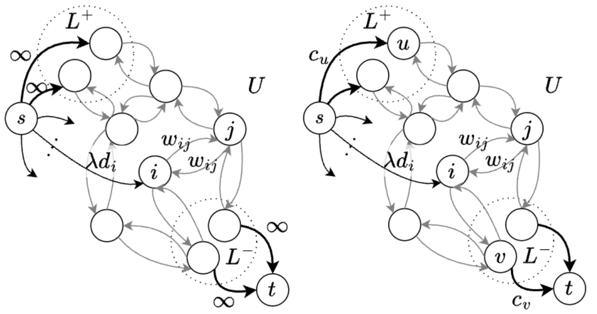
Not only are problems (4.3) and (4.4) solved efficiently for a given value of λ, but they were also shown in [14, 15] to be solved for *all* values of λ, using the parametric minimum cut algorithm of [13], in the complexity of a single minimum cut procedure on the associated graph. Therefore, the respective ratio problems (4.1) and (4.2) are solvable as well in the complexity of a minimum cut on the graph. This efficient parametric cut procedure is used here for tuning our implementation and selecting the "best" value of λ, see Section 6.2.

The construction of the associated directed graph, $G_{st}(\lambda)$, given in [11, 12] and applied for problem (4.4), is as follows: In the graph representation of the data, $G = (V, E)$, each edge $[i,j] \in E$ is replaced by two directed arcs, (i,j) and (j,i) both carrying the same capacity weight w_{ij} that reflects their pairwise similarity. A source node s and a sink node t are added to the set of nodes V. There are arcs of infinite capacity between s and the nodes of L^+, as well as between nodes of L^- and t. These infinite capacity arcs guarantee that all nodes of L^+ are included in S with the source node and all nodes of L^- are included in $\bar{S}$ with the sink. For each node $i \in U$, there is an arc (s,i) of capacity λd_i. The graph $G_{st}(\lambda)$ is illustrated in Figure 1a.

It was proved in [14, 15] that for a minimum cut partition $(S^* \cup \{s\}, \bar{S}^* \cup \{t\})$ into a *source set* $S^* \cup \{s\}$ and a *sink set* $\bar{S}^* \cup \{t\}$, S^* is the optimal solution to (4.4). That means that unlabeled samples in S^* and $\bar{S}^*$ are classified as positive and negative, respectively.

4.2 Confidence HNC (CHNC)

Instead of forcing the labeled nodes to belong to the source or sink sets according to their labels, like how it was done in HNC, Confidence HNC permits labeled samples to belong to the opposite set with penalty, or confidence weight, c_i, for labeled sample i. The confidence weight c_i reflects how *confident* we are in the label quality of sample i. The confidence weights computation is explained in Section 5. Relying on the HNC formulation (4.5), we write CHNC as,



(a) Supervised HNC (4.4)   (b) Confidence HNC

Figure 1: (a) HNC graph or $G_{st}(\lambda)$: Nodes s and t are added to the data graph G along with additional arcs to form $G_{st}(\lambda)$. Bold arcs are arcs with infinite weights. (b) CHNC graph, or $G'_{st}(\lambda)$, where infinite arcs are replaced by arcs with finite confidence weights

$$\text{(4.6)} \quad \underset{\emptyset \subset S \subset V}{\text{minimize}} \; C(S,\bar{S}) - \lambda \sum_{i \in S \cap U} d_i + \sum_{i \in \bar{S} \cap L^+} c_i + \sum_{j \in S \cap L^-} c_j.$$

Clearly, HNC is a special case of CHNC where the penalty weights c_i are equal to ∞ and both problems are monotone IP3. We show next that (4.6) is a *Monotone IP3* problem.

Lemma 4.2. *Problem (4.6) is a Monotone IP3 problem.*

Proof. To prove the statement of the Lemma, we provide the respective formulation (4.7). Let x_i be a binary variable equal to 1 if node i belongs to S, and 0 otherwise. Let z_{ij} be a binary variable equal to 1 if (i,j) is in the cut $(S,\bar{S})$ i.e. $i \in S$ and $j \in \bar{S}$.

$$\text{(4.7)} \quad \begin{aligned} \min \sum_{[i,j] \in E} & w_{ij}(z_{ij} + z_{ji}) + \sum_{i \in L^+} c_i(1 - x_i) \\ & + \sum_{i \in L^-} c_i x_i - \lambda \sum_{i \in U} d_i x_i \end{aligned}$$

$$\begin{aligned} \text{subject to} \quad & x_i - x_j \leq z_{ij} && \forall [i,j] \in E \\ & x_j - x_i \leq z_{ji} && \forall [i,j] \in E \\ & x_i \in \{0,1\} && \forall i \in V \\ & z_{ij}, z_{ji} \in \{0,1\} && \forall [i,j] \in E \end{aligned}$$

This formulation is indeed monotone IP3 since each z-variable (third variable) appears in one constraint only, and its objective coefficient is positive. The other variables, the x-variables, appear at most twice per constraint, and with opposite sign coefficients, 1 and -1 in this case. □

The construction of the associated graph for (4.7), where a minimum cut partition provides the optimal solution is a special case of the generic construction described in [11, 12]. For CHNC, that graph is given in Figure 1b. The difference between the graph for HNC (1a) and CHNC (1b) is in replacing the infinite weight arcs between s to the positive labeled nodes and those between the negative labeled nodes to t by arcs of capacity equal to the confidence weights. We next show that, indeed, the minimum s,t-cut in the graph 1b implies the optimal solution to problem CHNC (4.6).

Theorem 4.1. *Let $(S^* \cup \{s\}, \bar{S}^* \cup \{t\})$ be a minimum s,t-cut for the Confidence HNC graph in Figure 1b. Then, S^* is an optimal solution to CHNC (4.6).*

Proof. Let $(s \cup S, t \cup \bar{S})$ be a finite (s,t) cut on $G'_{st}(\lambda)$. The capacity of this cut is given by

$$\begin{aligned} & C(s \cup S, t \cup \bar{S}) \\ & \quad = \sum_{i \in S, j \in \bar{S}} w_{ij} + \sum_{j \in U \cap \bar{S}} \lambda d_j + \sum_{i \in L^+ \cap \bar{S}} c_i + \sum_{k \in L^- \cap S} c_k \end{aligned}$$

Let D denote the volume of the set of unlabeled nodes, $D = \sum_{i \in U} d_i$, which is a constant since U is given. Therefore, $C(s \cup S, t \cup \bar{S})$ can be rewritten as

$$\sum_{i \in S, j \in \bar{S}} w_{ij} + \lambda D - \sum_{j \in U \cap S} \lambda d_j + \sum_{i \in L^+ \cap \bar{S}} c_i + \sum_{k \in L^- \cap S} c_k$$

Since λD is a constant, and noting that $C(S,\bar{S}) = \sum_{i \in S, j \in \bar{S}} w_{ij}$, it follows that minimizing $C\left(s \cup S, t \cup \bar{S}\right)$ is attained for the same set S that minimizes

$$C(S,\bar{S}) - \lambda \sum_{j \in U \cap S} d_j + \sum_{i \in L^+ \cap \bar{S}} c_i + \sum_{k \in L^- \cap S} c_k$$

which is the CHNC problem. □

Therefore, we solve for the optimal S^* in (4.6) by solving for the minimum (s,t)-cut of the CHNC graph (Figure 1b). Unlabeled samples that are in S^* are predicted positive and other unlabeled samples, or those in $\bar{S}^*$, are predicted negative.

Noise detection 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 $\{\bar{S} \cap L^+\} \cup \{S \cap L^-\}$ is the set of samples that are considered noisy by CHNC.

5 Implementation of CHNC

5.1 Graph sparsification The a-priori setting of the graph may include similarity comparisons between all pairs of samples in $L \cup U$. This leads to quadratic increase in the size of the graph, which can lead to computational burden. To reduce the number of edges in the graph, we apply the k-nearest neighbor sparsification. Nodes i and j are connected if i is among the k nearest neighbors of j, or j is among those of i. This technique was used in prior works such as [3, 4] and yielded competitive results compared to other graph construction methods [31]. Here, we set $k = 15$.

5.2 Pairwise similarities computation The pairwise similarities used as edge weights are set to depend on the distance between the respective samples. Given the distance between samples i and j, $dist(i, j)$, we use the Gaussian weight $w_{ij} = exp(-\frac{dist(i,j)}{2\varepsilon^2})$. Gaussian weight is commonly used, e.g. [6, 34]. After a careful inspection, we choose the value of ε to be 1.

To compute the distance, we use the weighted Euclidean distance where features are weighted according to the features' importance obtained from the random forest classifier. The importance of each feature is computed in the random forest based on the impurity reduction attributed to that feature. This step is incorporated so that the computed pairwise similarity reflects relationship between samples more accurately.

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 into M folds, denoted by $L_1, L_2, ..., L_M$. To compute confidence scores of samples in each fold L_m, we train HNC using $L \setminus L_m$ as labeled set, and L_m as unlabeled set. As a result, we obtain the predictions of samples in L_m, which could be either similar to or different from their given labels. Our hypothesis is that noisy samples would be predicted incorrectly.

We shuffle samples and repeat this process T times. Denote the prediction of a labeled sample i at iteration t by y_i^t, and the given label of i by y_i. The *confidence score* of the labeled sample i is computed as

$$\frac{\sum_{t=1}^{T} I(y_i^t = y_i)}{T} \tag{5.8}$$

$I(\cdot)$ is the indicator function. Samples that are always predicted correctly get scores of 1 while those that always have incorrect label predictions have scores of 0. In this work, we use $M = 5$ and $T = 4$.

We scale the confidence scores by multiplying them by the average pairwise similarity weight, $\frac{1}{|E|}\sum_{[i,j]\in E} w_{ij}$. This results in *confidence weights* that are of the same magnitude as pairwise similarity weights. A schematic description of the computation of the confidence weights is shown in Figure 2.

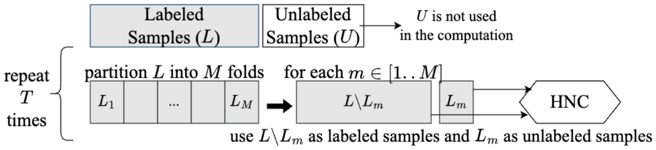

Labeled Samples (L)
Unlabeled Samples (U)
U is not used in the computation
repeat T times
partition L into M folds
for each $m \in [1..M]$
L_1
...
L_M
$L \backslash L_m$
L_m
HNC
use $L \backslash L_m$ as labeled samples and L_m as unlabeled samples

Figure 2: Confidence score computation: Confidence score of each labeled sample is equal to the proportion of correct predictions yielded by HNC trained on other samples.

6 Experiments

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

6.1 Baseline methods We compare Confidence HNC with the following methods. The first method is the Nearest Neighbor Editing Aided by Unlabeled Data (NNEAU) [9]. The base classifiers used in the editing step of NNEAU are the kNN classifier, the naive Bayes classifier and the classification tree. The second method is Co-teaching+ (CT+) [33]. We capped the training of CT+ at 100 epochs and included early stopping to terminate the training process when the training accuracy does not increase for 5 consecutive epochs. Third, we evaluate the Confident Learning method (CL) [23], that has been implemented as a Python package called Cleanlab. We use the implementations of Co-teaching+ and Confident Learning from `https://github.com/xingruiyu/coteaching_plus` and `https://docs.cleanlab.ai/stable/index.html#`.

As mentioned in Section 2, there are other leading noise-resilient models such as DivideMix [17]. However, available implementation of these models are customized for tasks on image data, and they have been evaluated mostly on image data. We focus on tabular data and do not compare these models here.

6.2 Hyperparameters of Confidence HNC The tuning for CHNC involves selecting λ that delivers the best 5-fold cross-validation accuracy, selected from 20 candidate values: $0.02, 0.04, 0.06, ..., 60, 80, 100$. We also tune on whether the set S in (4.6) is designated for the positive class or the negative class. The tuning of the two parameters can be done jointly in a single run of

Parameter	Values
# of samples	1000, 5000
# of features	5, 10, 20
% of positive samples	30%, 40%, 50%, 60%, 70%
# of clusters per class	2, 4
class separation	0.5, 1, 2
centroids on vertices of ..	hypercube, polytope

Table 1: Synthetic Data Configurations

Name	Size	# Attr	%Pos
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 UCI ML repository

the parametric minimum cut. Therefore, the evaluation of all 40 pairs of parameter values altogether runs in the complexity of a single minimum cut. We use the Pseudoflow Parametric algorithm, which is available at `https://riot.ieor.berkeley.edu/Applications/Pseudoflow/parametric.html` [5], that allows us to tune the hyperparameters efficiently.

6.3 Datasets We test the classification methods on both synthetic and real datasets. Synthetic datasets are generated using scikit-learn [26]. The generation of synthetic data involves a number of parameters as indicated in in Table 1 along with the values that we used. See [26] for more details of these parameters. There are 360 configurations of these parameters. For each configuration, we generate 4 different datasets, adding up to 1440 synthetic datasets. In terms of real datasets, we experimented on 6 datasets from the UCI Machine Learning Repository [8] listed in Table 2.

6.4 Experiment settings For each dataset, we partition data samples into labeled set, which contains 80% of the samples, and unlabeled set, which contains the remaining 20%. To evaluate the models' robustness to label noise, we add noise to 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 model uses this corrupted labeled set to train the model and predict the labels of unlabeled samples. For real data, we run 5 experiments for each of them at each noise level using different randomization of label corruption and different partitions into labeled-unlabeled samples.

6.5 Evaluation metrics First, we evaluate the classification performance using the accuracy improvement (Acc Imp). Acc Imp of CHNC over model M is equal to $((Acc\, of\, CHNC\ /\ Acc\, of\, M) - 1) \times 100\%$. A positive Acc Imp implies a higher accuracy of CHNC over the model M.

Second, we examine noise detection capability through noise recall and precision. 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 the fraction of noisy samples detected by the model, or $\frac{|N \cap D|}{|N|}$. Noise precision is the fraction of noisy samples among samples that are considered noisy by the model, or $\frac{|N \cap D|}{|D|}$.

To test for the significance of the results, we use the Wilcoxon signed-ranks test, which is commonly used in many works such as [1, 25, 21]. It is an appropriate statistical test when we compare two classifiers over multiple datasets, and do not assume that the differences are normally distributed [7].

Moreover, we also inspect the confidence scores of clean and noisy samples to evaluate our proposed confidence weights computation.

7 Results

7.1 Results on synthetic data From experiments on synthetic datasets, we obtain 1440 classification accuracy scores for each classifier. We compute the accuracy improvement given by Confidence HNC at both 15% and 30% noise levels, and plot the histograms as shown in Figures 3 and 4.

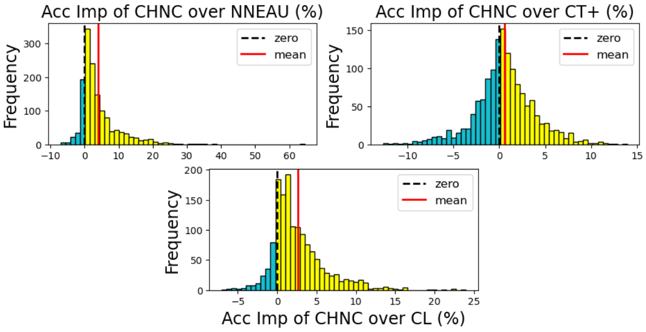


Figure 3: Histograms of accuracy improvement on 1440 synthetic datasets, with 15% noise, yielded by CHNC. Area to the right of the dashed line indicates the number of datasets where CHNC outperforms. The red solid line indicates the mean of improvement.

In each plot, the dashed vertical line is at zero improvement. The area in yellow on the right of this line indicates the proportion of datasets on which CHNC has positive accuracy improvement. The red solid line marks the average accuracy improvement. In Figures 3 and 4, for both noise levels and for all three classifiers, the red line lies to the right of the dashed line, implying positive average improvement. Compared with NNEAU, CT+ and CL, CHNC outperforms in $81.94\%, 57.29\%$ and 81.32% of the synthetic datasets at 15% noise level, and $83.60\%, 64.21\%$ and 63.45% at 30% noise level. These percentages are the areas of the histograms that lie to the right of their zero dashed lines.

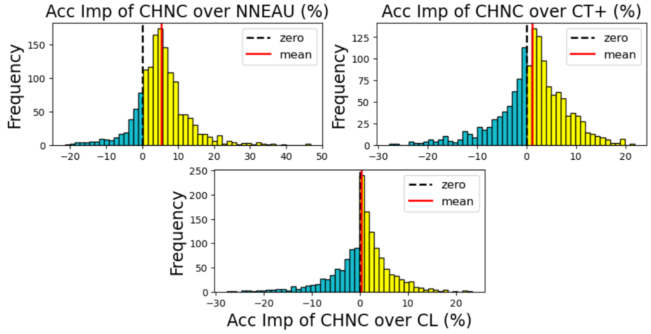


Figure 4: Histograms of accuracy improvement on 1440 synthetic datasets, with 30% noise, yielded by CHNC. Area to the right of the dashed line indicates the number of datasets where CHNC outperforms. The red solid line indicates the mean of improvement.

We apply the Wilcoxon test to evaluate the statistical significance of the difference between the accuracy scores of CHNC and other methods. P-values when tested with NNEAU, CT+ and CL are all smaller than 0.05, or even 10^{-5} to be precise, at both 15% and 30% noise levels, validating the significance of the results. We also inspect t-test p-values, which are smaller than 10^{-9} in all cases, except for when compared to CL at 30% noise where t-test p-value is 0.078.

Regarding the noise detection capability, compared to NNEAU and CL, CHNC achieves significantly higher noise precision for both noise levels, but it did not improve on noise recall. Compared to CT+, CHNC outperforms on both metrics for 30% noise level and improves only on noise recall for 15% noise. Hence, for most part, noise precision is the strength of CHNC. High noise precision of CHNC implies that CHNC does not wrongly identify good samples as noisy as much as other methods.

7.2 Results on real data Classification accuracy improvements of CHNC over baseline methods 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 all datasets except for GER and OBE. For the OBE data, the accuracy of CHNC is only slightly smaller than CT+, and it is higher than that of NNEAU and CL. The standard deviation of the accuracy of CHNC is close to that of other models in most cases. P-values given by the Wilcoxon test demonstrate that CHNC performs better than both NNEAU and CL with strong statistical significance.

Data	**CHNC**	**NNEAU**	**CT+**	**CL**
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		0.0005*	0.2760	0.0011*

Table 3: Classification accuracy (mean ± std err)(%) on real data with 15% noise and p-values of the outperformance of CHNC. The highest accuracy is in bold. (*) marks strong statistical significance.

Data	**CHNC**	**NNEAU**	**CT+**	**CL**
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		4.42e-6*	0.0021*	0.0409*

Table 4: Classification accuracy (mean ± std err)(%) on real data with 30% noise and p-values of the outperformance of CHNC. The highest accuracy is in bold. (*) marks 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%. For the two datasets, WIN and CAR, where CL is dominant, the performance of CHNC is fairly close to that of CL. P-values given by the Wilcoxon test indicate that CHCN gives significantly higher accuracy than all three methods.

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, except for OBE at 15% noise level and GER at 30% noise level.

Furthermore, we assess our confidence weight computation method (Equation 5.8) by examining the confidence scores of labeled samples. Score distribution of labeled samples in two datasets, MAT and OBE, for the noise level of 15%, are shown in Figure 5. Above the zero line are histograms of scores of uncorrupted labeled

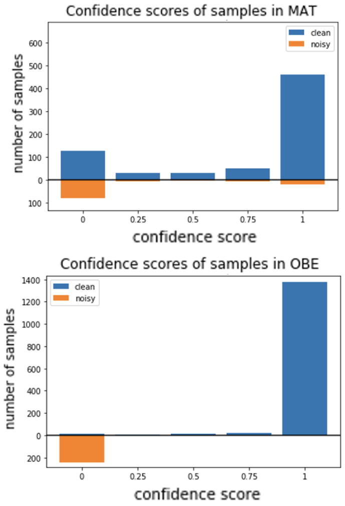


Figure 5: Distribution of confidence scores of clean (in blue) and noise (in orange) samples for datasets MAT and OBE at 15% noise level

samples. In both datasets, most clean samples have perfect confidence scores. Similarly, as seen below the horizontal line, most noisy labeled samples have confidence scores of zero as desired.

Last but not least, note that, when interpreting results on real data, there might be noise contained in the data even before we corrupt them, unlike synthetic data. The interpretation of results on real data can be obscured by this issue.

8 Conclusions

We introduce a new classification method, Confidence Hochbaum's Normalized Cut (CHNC), that varies its reliance on the labeled data through the use of confidence weights. CHNC allows labeled samples to take on labels that are different from their given labels in the graph partition. If a label is reversed then it is considered to be noisy since it is "more similar" to other samples that have another label. Another contribution is a procedure to detect noisy labels, or to compute the confidence weights of labeled samples based on HNC.

Our experiments demonstrate that CHNC is competitive in terms of classification accuracy and noise precision. CHNC outperforms all baseline models in most cases and remains competitive with Co-teaching+ at 15% noise level on real data. These results substantiate the viability of CHNC in the presence of label noise.

Additionally, we would like to mention the flexibility of our CHNC implementation. When it is suitable to use pairwise similarity measure and confidence weights that are appropriate for a particular data domain instead of the ones proposed in this work, or when one has prior information about these values, one can easily supply the preferred pairwise similarity weights and confidence weights to the model.

While this work exhibits favorable results of Confidence HNC, it is also important that we discuss one of its limitations here. CHNC only solves a binary classification task. Hence, extending the use of CHNC to multiclass classification can be an interesting direction for future works.

For other future works, we plan to explore the robustness of CHNC or other variants of HNC to asymmetric label noise or instance-dependent label noise since there are situations where samples are more susceptible to noise corruption than others. Another direction is to investigate how CHNC can be devised to deal with limited number of labeled samples. Since CHNC is a semi-supervised learning method which utilizes unsupervised information in the form of similarities between samples, it is particularly suitable for scenarios with limited availability of labeled samples. Finally, note that the performance of our method, as well as methods like k nearest neighbor, depends heavily on the pairwise similarity measure choice, whether it reflects the true proximity between samples. Appropriate selection of similarity measure can be conditional on data domain, as well as on the classifiers in the experiments. Learning appropriate similarity measure is indeed another direction to pursue.

ACKNOWLEDGEMENTS

This research was supported in part by AI Institute NSF Award 2112533.

References

[1] Abellán, J. & Masegosa, A. Bagging schemes on the presence of class noise in classification. *Expert Systems With Applications*. **39**, 6827-6837 (2012)

[2] Baumann, P., Hochbaum, D. & Yang, Y. A comparative study of the leading machine learning techniques and two new optimization algorithms. *European Journal Of Operational Research*. **272**, 1041-1057 (2019)

[3] Blum, A. & Chawla, S. Learning from labeled and unlabeled data using graph mincuts. (Carnegie Mellon University,2001)

[4] Blum, A., Lafferty, J., Rwebangira, M. & Reddy, R. Semi-supervised learning using randomized mincuts. *Proceedings Of The Twenty-first International Conference On Machine Learning*. pp. 13 (2004)

[5] Chandran, B. & Hochbaum, D. Pseudoflow Parametric Maximum Flow Solver Version 1.0. *Pseudoflow Parametric Maximum Flow Solver Version 1.0.*, https://riot.ieor.berkeley.edu/Applications/Pseudoflow/parametric.html

[6] Chen, Y., Garcia, E., Gupta, M., Rahimi, A. & Cazzanti, L. Similarity-based classification: Concepts and algorithms.. *Journal Of Machine Learning Research.* **10** (2009)

[7] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *The Journal Of Machine Learning Research.* **7** pp. 1-30 (2006)

[8] Dua, D. & Graff, C. UCI Machine Learning Repository. (University of California, Irvine, School of Information,2017), http://archive.ics.uci.edu/ml

[9] Guan, D., Yuan, W., Lee, Y. & Lee, S. Nearest neighbor editing aided by unlabeled data. *Information Sciences.* **179**, 2273-2282 (2009)

[10] Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I. & Sugiyama, M. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances In Neural Information Processing Systems.* **31** (2018)

[11] Hochbaum, D. Solving integer programs over monotone inequalities in three variables: A framework for half integrality and good approximations. *European Journal Of Operational Research.* **140**, 291-321 (2002)

[12] Hochbaum, D. Applications and efficient algorithms for integer programming problems on monotone constraints. *Networks.* **77**, 21-49 (2021)

[13] Hochbaum, D. The pseudoflow algorithm: A new algorithm for the maximum-flow problem. *Operations Research.* **56**, 992-1009 (2008)

[14] Hochbaum, D. Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE Transactions On Pattern Analysis And Machine Intelligence.* **32**, 889-898 (2010)

[15] Hochbaum, D. A polynomial time algorithm for rayleigh ratio on discrete variables: Replacing spectral techniques for expander ratio, normalized cut, and cheeger constant. *Operations Research.* **61**, 184-198 (2013)

[16] Kim, T., Ko, J., Choi, J., Yun, S. & Others Fine samples for learning with noisy labels. *Advances In Neural Information Processing Systems.* **34** pp. 24137-24149 (2021)

[17] Li, J., Socher, R. & Hoi, S. Dividemix: Learning with noisy labels as semi-supervised learning. *ArXiv Preprint ArXiv:2002.07394.* (2020)

[18] Lin, C. & Others Training algorithms for fuzzy support vector machines with noisy data. *Pattern Recognition Letters.* **25**, 1647-1656 (2004)

[19] Liu, T. & Tao, D. Classification with noisy labels by importance reweighting. *IEEE Transactions On Pattern Analysis And Machine Intelligence.* **38**, 447-461 (2015)

[20] Liu, S., Niles-Weed, J., Razavian, N. & Fernandez-Granda, C. Early-learning regularization prevents memorization of noisy labels. *Advances In Neural Information Processing Systems.* **33** pp. 20331-20342 (2020)

[21] Luengo, J., Shim, S., Alshomrani, S., Altalhi, A. & Herrera, F. CNC-NOS: Class noise cleaning by ensemble filtering and noise scoring. *Knowledge-Based Systems.* **140** pp. 27-49 (2018)

[22] Nguyen, D., Mummadi, C., Ngo, T., Nguyen, T., Beggel, L. & Brox, T. Self: Learning to filter noisy labels with self-ensembling. *ArXiv Preprint ArXiv:1910.01842.* (2019)

[23] Northcutt, C., Jiang, L. & Chuang, I. Confident learning: Estimating uncertainty in dataset labels. *Journal Of Artificial Intelligence Research.* **70** pp. 1373-1411 (2021)

[24] Pleiss, G., Zhang, T., Elenberg, E. & Weinberger, K. Identifying mislabeled data using the area under the margin ranking. *Advances In Neural Information Processing Systems.* **33** pp. 17044-17056 (2020)

[25] Sáez, J., Galar, M., Luengo, J. & Herrera, F. INFFC: An iterative class noise filter based on the fusion of classifiers with noise sensitivity control. *Information Fusion.* **27** pp. 19-32 (2016)

[26] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. Scikit-learn: Machine Learning in Python. *Journal Of Machine Learning Research.* **12** pp. 2825-2830 (2011)

[27] Sharon, E., Galun, M., Sharon, D., Basri, R. & Brandt, A. Hierarchy and adaptivity in segmenting visual scenes. *Nature.* **442**, 810-813 (2006)

[28] Shi, J. & Malik, J. Normalized cuts and image segmentation. *IEEE Transactions On Pattern Analysis And Machine Intelligence.* **22**, 888-905 (2000)

[29] Thulasidasan, S., Bhattacharya, T., Bilmes, J., Chennupati, G. & Mohd-Yusof, J. Combating label noise in deep learning using abstention. *ArXiv Preprint ArXiv:1905.10964.* (2019)

[30] Tomek, I. AN EXPERIMENT WITH THE EDITED NEAREST-NIEGHBOR RULE.. (1976)

[31] Wang, J., Jebara, T. & Chang, S. Semi-supervised learning using greedy max-cut. *The Journal Of Machine Learning Research.* **14**, 771-800 (2013)

[32] Yang, Y., Fishbain, B., Hochbaum, D., Norman, E. & Swanberg, E. The supervised normalized cut method for detecting, classifying, and identifying special nuclear materials. *INFORMS Journal On Computing.* **26**, 45-58 (2014)

[33] Yu, X., Han, B., Yao, J., Niu, G., Tsang, I. & Sugiyama, M. How does disagreement help generalization against label corruption?. *International Conference On Machine Learning.* pp. 7164-7173 (2019)

[34] Zhu, X. & Goldberg, A. Introduction to semi-supervised learning. *Synthesis Lectures On Artificial Intelligence And Machine Learning.* **3**, 1-130 (2009)