IE 506: Machine Learning - Principles and Techniques **Jan-May 2024**

Project Mid-term Report : **Harnessing Side Information for Classification Under Label Noise**

Team Name: Optimistic Learners *Team Members: 23N0457, 23N0452*

Abstract

This project report contains the details on our project which aims to solve a classification problem while being robust to the label noise treating feature matrix as side information. We describe the optimisation algorithm required for this method and why this is superior to the other conventional works, also we explain the necessary assumptions taken and why this is necessary. In particular, we explain about the ML method used for the project, the details of training procedure, details of inference and discuss about experiments conducted. We have tried to validate the algorithm using the toy dataset and then performed experiments on one of the dataset and compared the performance with the other algorithm.

1 Introduction

The classification problem is really important in the world of learning. It's all about sorting things into different groups. Think of it like sorting your toys into different boxes based on what they are. This idea is used in lots of everyday things, like sorting emails into spam or not spam, or figuring out if a picture has a cat or a dog in it. People are always working to make these sorting systems better, so they can help with more and more things in our lives. When the problem is Classification the most off the shelf methods are used like logistic regression , support vector Machines, Decision trees, K-nearest neibours, but this algorithm runs with the assumption that the data is correctly labeled but sometimes, it's not perfect and contains errors or irrelevant information, which we call noise. This noise can confuse classification systems, making it harder for them to sort things accurately. Handling noise is a big challenge in classification research., often the labels gets noise due to various reasons namely: Human Error, Crowdsourcing, Automation Errors, Data Quality, Concept Drift, Adversarial Attacks, Biases and Stereotypes, Incomplete Information. In this regard we need to bring in some methods which are robust to such label noise and are able to solve classification problem both in noisy and no noise cases. To deal with this Patrini [1] tried to analyse the empirical risk minimisation method to make the off-the shelf methods be more robust to the label noise. However, they are only applicable to binary classification and the extension to multi-class is non-trivial[3]. Moreover these methods require the estimation of class prior, which is actually quite difficult in the presence of corrupted data.

Therefore our project is based on dealing with the label noise problem from a different viewpoint of side information.Specifically,the method formulate label correction as a label matrix recovery problem and treat the example features as side information to aid the recovery process. Therefore, the method is named as "label noise handling via side information" (LNSI). The paradigm of this method is shown in Fig. 1, which intuitively explains how to transform the noisy label removing problem to the label matrix recovery task by exploiting the side information.

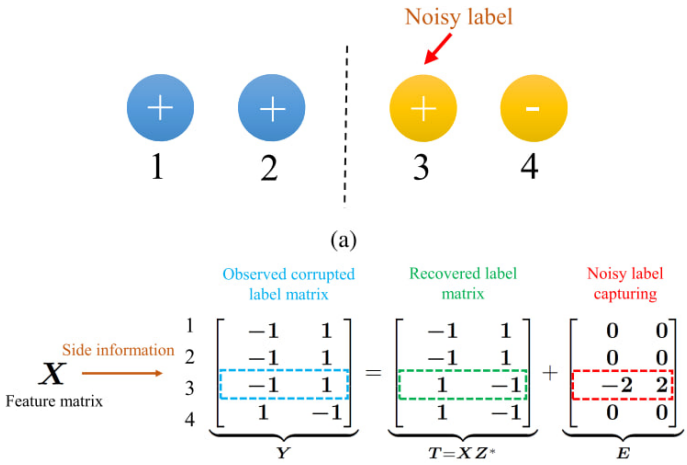

Figure 1: Decomposition of the label matrix into ground truth matrix and the error labeling matrix

2 Contributions

1. **Suitable for Multi-Class Classification**: Our method, known as LNSI, excels in handling scenarios where there are many different categories or classes that need to be sorted out. While some other techniques might require complex procedures to deal with numerous classes, LNSI simplifies the process by seamlessly integrating all the classes together. This means you don't have to worry about complicated operations when dealing with multiple categories – LNSI takes care of it for you.

2. **Reliability Supported by Theoretical Analysis**: Researchers have extensively analyzed how well LNSI performs under various circumstances. What they've discovered is quite reassuring: under certain favorable conditions, LNSI has the remarkable ability to accurately identify the true labels of the data. This means you can have confidence in the results provided by LNSI, knowing that it's built on solid theoretical foundations.

3. **Unified Solution for Noise Removal and Parameter Optimization**: LNSI doesn't just stop at removing noisy labels from the data. It goes a step further by also optimizing the settings of the classifier to ensure optimal performance. What's particularly convenient is that all of this is done within a single framework. By integrating both noise removal and parameter optimization, LNSI simplifies the workflow and makes the process more efficient. This unified approach also ensures that the classifier works at its best, even when dealing with noisy labels.

3 Literature Survey

Supervised learning is one of the major application based field in the Machine learning community, but the very off the shelf algorithms for the classification problems like SVMs, logistic regression, decision trees aren not that robust to label noise, which seems to be the case in real life, to incorporate the robustness against noise there have been several methods, we have encountered two of such methods in this regard.
In the first method, the label having noise can be considered being not labeled and thus it makes a weak supervised learning problem (WSL).Traditionally, WSL problems are attacked by designing ad-hoc loss functions and optimization algorithms tied to the particular learning setting. One of this method in explained by Patrini[1] in his paper, he sufficiency advocates for a more principled two-step strategy: (1) estimating

the mean operator μ from weakly supervised data, and (2) integrating it into any Linear Odd Loss and employing established procedures for empirical risk minimization. Consequently, (1) emerges as the sole technical in adapting this algorithm, though often easily surmounted.(2) is being illustrated by modifying stochastic gradient descent (SGD) for WSL. We simply need to convert the input into a "double sample" S_{2x} — each instance provided twice, once labeled with each label — and incorporate μ into the model update. For each instance (x_i, y_i) in S_{2x}, with a constant determined by the choice of λ and η, the learning rate, we obtain:

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla \ell(y_i, h_{\theta_t}(x_i)) - \frac{1}{2}\eta\lambda\mu.$$

We then turn our attention to learning with asymmetric label noise, characterized by noise rates (p^+, p^-), where p^+, p^- is the rate at which the +1 label data is hampered and -1 label data is hampered repectively, We extend existing methods by devising an unbiased estimator of μ:

$$\hat{\mu} = \mathbb{E}(x,y)\left[y - \frac{(p^- - p^+)}{1 - p^- - p^+}x\right].$$

Here, the difference would be we will not be using the empirical estimation rather we will get this robust estimation.
Algorithm 1, μSGD, is a method that tweaks SGD for situations where supervision is weak. Imagine we're using a simple version of SGD that uses subgradient descent with L2 regularization. To adapt it for μ, we make simple changes: (i) create a new set called S_{2x} based on the existing set $\widetilde{S}$, and (ii) add $\frac{a\mu}{2}$ to the subgradients of each example in S_{2x}. Later on, in Section 6, we'll talk about how we can use this same simple method to improve other algorithms.

Input:

- S_{2x}: Dataset
- μ: Learning rate
- $\lambda > 0$: Regularization parameter
- $T > 0$: Number of iterations

Initialization:

- $m_0 = |S_{2x}|$
- $\theta_0 = 0$

Iteration:

- For $t = 1, \ldots, T$:
  - Pick $i_t \in [m_0]$ uniformly at random
  - $\eta_t = (\lambda t)^{-1}$
  - Pick any $v \in \partial`(y_i h_{\theta_t}, x_i)$

- $\theta_{t+1} = (1 - \eta_t\lambda)\theta_t - \eta_t(v + a\mu/2)$
- $\theta_{t+1} = \min_n \left(\theta_{t+1}, \frac{\theta_{t+1}}{\sqrt{\lambda^{-1}\|\theta_{t+1}\|_2^2}}\right)$

Output:

- θ_{T+1}

Algorithm2, stays true no matter which algorithm or type of loss function you use. Specifically, we show that every learning method has a property that makes it more resistant to uneven label errors, depending on the data distribution which is estimated through a unbiaset estimator

Input Parameters:

- $\tilde{S}$: Dataset with instances and their associated noisy labels.
- λ: Regularization parameter.
- T: Number of iterations.

Step 1: Estimate μ:

- Compute an estimate of the mean operator μ from $\tilde{S}$.

Step 2: Apply μ-SGD:

- Initialize θ.
- Use μ-SGD on a modified dataset S_{2x}.
- Incorporate μ estimate, λ, and T.
- Update θ iteratively.

Output:

- Return learned θ.

Thus we will get a learned θ, which can be further used for inference.Although this method seems to be fine for the binary class classification problem wheras it is non-trivial for the multi-class problems [3]. To incorporate the idea of multiclass classification problem, another view point of side information came into picture. If considered that the erroneous labels are missing values and we have to find the missing values in the label matrix, then this problem can be considered as a matrix completion matrix [2]. Let $\hat{R}$ be a matrix in $\mathbb{R}^{n_1 \times n_2}$ with rank r that requires recovery. Additionally, let $\hat{X}$ and $\hat{Y}$ be matrices in $\mathbb{R}^{n_1 \times d_1}$ and $\mathbb{R}^{n_2 \times d_2}$ (where d_1 and d_2 denote the feature dimensions), respectively, recording the row and column features of $\hat{R}$. These matrices serve as side information for recovering $\hat{R}$. The recovery model proposed by Chiang et al. is formulated as follows:

$$\min_{M,N} \sum_{(i,j)\in\Omega} \left((\hat{X}M\hat{Y}^T + N)_{ij} - \hat{R}_{ij}\right)^2 + \lambda_M\|M\|_* + \lambda_N\|N\|_*$$

Here, $\|\cdot\|_*$ represents the nuclear norm, and Ω denotes the index set containing all observed entries of $\hat{R}$. The observed $\hat{R}$ can be decomposed into two parts: one is the low-rank matrix estimated from the feature space $\hat{X}M\hat{Y}^T$, where $M \in \mathbb{R}^{d_1 \times d_2}$, a co-embedding matrix comes from the features $\hat{X}$ and $\hat{Y}$, while another part, N, holds information that the noisy features (represented by $\hat{X}$ and $\hat{Y}$) cannot fully describe. The loss function ensures that the recovered matrix matches $\hat{R}$ where we have observations. Naturally, we want both $\hat{X}M\hat{Y}^T$ and N to be simple since they help us estimate a simpler matrix $\hat{R}$, which encourages M to also be simple. The parameters λ_N and λ_M control the balance between simplicity and accuracy. We can estimate the original matrix $\hat{R}$ using $\hat{X}M^*\hat{Y}^T + N^*$, where M^* and N^* are found by optimizing the objective function. This approach has been proven to work well in both theory and practice.

Inspired by this approach, the problem of removing noisy labels as a task of recovering a label matrix, where the example features describe the rows has been taken forward. So, our LNSI method benefits from the strengths of this model, ensuring good results.

4 Methods and Approaches

Here in our project we have been using LNSI model as stated in the introduction. In this method we consider the feature matrix $X \in \mathbb{R}^{n \times d}$ (d denotes the feature dimensionality and n denotes the number of samples) is considered as the side information which is used to recover the label matrix and the label matrix $Y \in \mathbb{R}^{n \times c}$ (c denotes the number of classes in the label) is the observed label matrix with corrupted entries.

Let X_i represent the ith row of matrix X, which records the features of the ith example. The function Y_{ij} can be defined as follows:

$$Y_{ij} = \begin{cases} 1 & \textbf{if the } i\textbf{th example has label } j, \\ -1 & \textbf{otherwise.} \end{cases}$$

Here, the main step involves decomposing the matrix Y into two parts: XZ and E, where:

- XZ represents the true label matrix, where Z is the projection matrix of the feature vectors into the actual label martrix.
- E captures the error in prediction, the entries in E are such that it has non-zero entries -2 and 2 only in those sample positions where their is misclassification.

Thus the errors in labels can be effectively captured by a row-sparse matrix E since label noise in training examples tends to be sparse. Consequently, the number of corresponding nonzero rows in E should be small. Additionally, a Frobenius norm is imposed on Z to prevent overfitting.By adding the Frobenius norm regularization term to the loss function, the model is encouraged to have smaller parameter values and thus help in preventing the overfitting which is the case when the model tries to mimic the behaviour of the train data.Also the problem has some nuclear norm on the Z matrix. The nuclear norm provides a measure of the "rank" of a matrix. A matrix with a low nuclear norm typically has a low-rank structure. This is because the nuclear norm is the sum of the singular values, and the number of nonzero singular values corresponds to the rank of the matrix.Matrices with low-rank structures tend to be more robust to noise and outliers. The nuclear norm regularization can help in denoising or robustifying optimization problems by promoting solutions with lower-rank structures. Combining these considerations,the following optimization problem is formulated:

$$\begin{aligned} &\min_{Z,E} \|Z\|_* + \lambda_1 \|Z\|_F^2 + \lambda_3 \|E\|_{2,1} \\ &\textbf{s.t. } Y = XZ + E \textbf{ and } XZ \in \{-1, 1\}^{n \times c} \end{aligned} \tag{1}$$

(2)

Now, as in the introduction we have mentioned the use of feature matrix as the side information, thus to exploit the side information we furthur use the Laplacian regularizer based on graph embedding.Graph embedding techniques play a vital role in many machine learning tasks, where data points are represented as nodes in a graph, with edges denoting relationships or similarities between them. These techniques aim to learn low-dimensional embeddings of nodes that capture essential structural information and relationships within the graph. Central to this concept is the Laplacian matrix, a mathematical representation that encapsulates the graph's structure and quantifies its smoothness or connectivity. Different types of Laplacian matrices, including the unnormalized, symmetric normalized, and random walk Laplacian, each capture distinct aspects of graph structure. In machine learning tasks involving graph data, the Laplacian regularizer is employed to ensure the smoothness and consistency of node embeddings with respect to the graph structure. By penalizing deviations between the embeddings of neighboring nodes, this regularizer promotes similar embeddings for connected nodes, effectively exploiting side information encoded in the graph structure. This exploitation proves particularly beneficial in tasks such as node classification, link prediction, and graph-based semi-supervised learning, where leveraging relational information among data points is crucial for achieving high performance.
Let $G = \{V, E\}$ be an undirected weighted graph with vertex set V consisting of all n examples, and E is the edge set encoding the similarity between these examples. The symmetric adjacency matrix $\hat{W} \in \mathbb{R}^{n\times n}$ is utilized to quantify the graph G, where $\hat{W}_{ij} = \exp\left(-\frac{\|X_i - X_j\|_2^2}{2\sigma_k^2}\right)$ (where σ_k is the kernel width) measures the similarity between examples X_i and X_j $(i, j = 1, 2, \ldots, n)$. The diagonal matrix D and the Laplacian matrix L of the graph G are, respectively, defined as

$$D_{ii} = \sum_j \hat{W}_{ij} \quad \textbf{and} \quad L = D - \hat{W}.$$

Therefore, we have the Laplacian regularizer that is derived as

$$\sum_{i,j} \hat{W}_{ij} \|X_i Z - X_j Z\|_2^2 = \mathbf{tr}((XZ)^T L(XZ)) \quad (2)$$

in which tr$(\cdot)$ computes the trace of the corresponding matrix.
Thus by combinning (1) and (2) we have the obtimisation problem framed as:

$$\begin{aligned} &\min_{Z,E} \|Z\|_* + \lambda_1 \|Z\|_F^2 + \lambda_2 \mathbf{tr}((XZ)^T L(XZ)) + \lambda_3 \|E\|_{2,1} \\ &\textbf{s.t. } Y = XZ + E, \quad XZ \in \{-1, 1\}^{n\times c} \end{aligned} \quad (3)$$

where λ_1, λ_2, and λ_3 are the nonnegative tradeoff parameters.
The above problem falls in the category of the integer programming problem, which is generally NP-hard.To make this problem, we relax the discrete constrainsts $XZ \in \{-1, 1\}^{n\times c}$ to a continuous convex set $XZ \in [-1, 1]^{n\times c}$.Thus the problems turns out to be:

$$\begin{aligned} &\min_{Z,E} \|Z\|_* + \lambda_1 \|Z\|_F^2 + \lambda_2 \mathbf{tr}((XZ)^T L(XZ)) + \lambda_3 \|E\|_{2,1} \\ &\textbf{s.t. } Y = XZ + E, \quad -1 \leq XZ \leq 1 \end{aligned} \quad (4)$$

We then itroduce two auxiliary variable so that we can use the method of Alternating direction method of multipliers(ADMMs) [], which optimizes the related variables in an iterative method breaking the problem with respect to separate optimisation problem for the separate variables.
The augmented Lagrangian function, with the continuous convex constraint can be written as:

$$L(Z,E,B,J,M_1,M_2,M_3) = \|Z\|_* + \lambda_1\|Z\|_F^2 + \lambda_2 \text{tr}((XJ)^T L(XJ)) + \lambda_3\|E\|_{2,1} + \text{tr}\left(M_1^T(Y-B-E)\right) + \text{tr}\left(M_2^T(B-XJ)\right) + \text{tr}\left(M_3^T(Z-J)\right) + \frac{\mu}{2}\left(\|Y-B-E\|_F^2 + \|B-XJ\|_F^2 + \|Z-J\|_F^2\right)$$

Where M_1,M_2 and M_3 are the Lagrangian multipliers and $\mu > 0$ is the penalty coefficient. Thus we will first minimize $L(Z,E,B,J,M_1,M_2,M_3)$ then maximize with respect to M_1,M_2 and M_3.This problem can be broken into 4 subproblems. Since there are 4 variables we can initiate 3 of the variables and solve a separate optimization for the remaining variable and iterate over with the same technique. The optimisation for each variable update is given below:

$$\textbf{Update Z:} \min_Z \|Z\|_* + \lambda_1\|Z\|_F^2 + \text{tr}\left(M_3^T(Z-J)\right) + \frac{\mu}{2}\|Z-J\|_F^2$$

This update can be solved by some manupulation and then it has a standard solution of the form:

$$Z = U\text{diag}(\max\{\lambda_{ii} - \tau, 0\})V^T \quad \forall i = 1,2,\ldots,\min(d,c) \tag{5}$$

where U and V are obtained by conducting the singular value decomposition (SVD) on $\hat{T}$ (i.e., $\hat{T} = U\Sigma V^T$), and λ_{ii} is the ith diagonal element of the singular value matrix Σ.

$$\begin{aligned}\textbf{Update E:} \min_E \quad & E\lambda_3\|E\|_{2,1} + \text{tr}(M^T(Y-B-E)) \\ & + \frac{\mu_2}{2}\|Y-B-E\|_F^2\end{aligned} \tag{6}$$

after manipulation and solving the udate can be given as :

$$E_i = \begin{cases} \|M_i\|_2^2 - \eta\|M_i\|_2 M_i & \text{if } \|M_i\|_2^2 > \eta \\ 0 & \text{otherwise} \end{cases} \quad (14) \tag{7}$$

where E_i and M_i represent the ith row of the related matrices, respectively. Here, $M = Y - B + \left(\frac{M_1}{\mu}\right)$ and $\eta = \frac{\lambda_3}{\mu}$. M is defined as the matrix obtained by subtracting B from Y and adding the scaled version of M_1 by $\frac{1}{\mu}$.
The subproblem corresponding J is:

$$\begin{aligned}\textbf{Update J:} \min_J \quad & \lambda_2 \text{tr}((XJ)^T L(XJ)) + \text{tr}(M_2(B-XJ)) \\ & + \text{tr}(M_3(Z-J)) + \frac{\mu_2}{2}\|B-XJ\|_F^2 + \|Z-J\|_F^2\end{aligned} \tag{8}$$

$$J = \left(2\lambda_2 X^T L X + \mu X^T X + \mu I\right)^{-1}\left(X^T M_2 + M_3 + \mu X^T B + \mu Z\right) \tag{9}$$

This update rule for J is derived by computing the derivative of the expression in equation (8) with respect to J, setting it to zero, and solving for J.

$$\begin{aligned}\textbf{Update B :} \min_B \quad & \text{tr}(M_1(Y-B-E)) + \text{tr}(M_2(B-XJ)) \\ & + \frac{\mu_2}{2}(\|Y-B-E\|_F^2 + \|B-XJ\|_F^2) \\ \text{s.t.} \quad & B \in [-1,1]^{n\times c}\end{aligned} \tag{10}$$

By first computing the derivative of (10) with respect to B and setting it to zero, the optimal $\hat{B}$ can be represented as:

$$\hat{B} = \mu(Y - E + XJ) + \frac{M_1 - M_2}{2\mu} \tag{11}$$

To restrict $\hat{B}$ to the feasible region, we further project all its elements to $[-1, 1]$ as:

$$B_{ij} = \Pi(\hat{B}_{ij}) \tag{12}$$

where the projection $\Pi(x)$ is defined as:

$$\Pi(x) = \begin{cases} 1, & \text{if } x > 1 \\ x, & \text{if } x \in [-1, 1] \\ -1, & \text{if } x < -1 \end{cases}$$

Thus, we obtain the required matrices Z and E after the iteration. The iteration continues based on the following conditions: $\|Y - B - E\|_F \leq \delta$, $\|B - XJ\|_F \leq \delta$, $\|Z - J\|_F \leq \delta$, or iter $>$ iter_max.

5 Data set Details

The project utilizes a subset of the MNIST dataset, comprising 2000 training and 2000 testing images. Each image, representing a handwritten digit from 0 to 9, has a resolution of 28×28 pixels. To ensure balanced representation, 200 images per digit category are selected for both training and testing. The data, primarily grayscale images, undergo preprocessing steps such as feature extraction using the first fully connected layer of VGGNet16, resulting in 4096-dimensional feature vectors. The dataset is obtained from the TensorFlow library.To ensure equitable representation, we select 200 images per digit category for both training and testing phases. The dataset primarily comprises grayscale images, which undergo preprocessing steps, including feature extraction via the initial fully connected layer of VGGNet16. This process yields 4096-dimensional feature vectors. The dataset originates from the TensorFlow library.

6 Experiments

Experiment Setup: For our experiments, we implemented our algorithm from scratch after gaining a thorough understanding of the paper. To validate our algorithm, we utilized the validation dataset consisting of 8 data points provided in the paper. Subsequently, we employed the MNIST dataset for testing purposes. Our experiment involved the following steps:

Loading and Preparing the Data: We loaded the MNIST dataset, which contains handwritten digit images and corresponding labels for training and testing. To ensure balanced representation across digit categories, we selected a subset of 2000 images for both training and testing. Specifically, we randomly chose 200 images for each digit from 0 to 9 for training and 2000 random images for testing.

VGG16 Model Initialization: We initialized a VGG16 model pretrained on the ImageNet dataset, leveraging its ability to extract hierarchical features from natural images. Omitting the top layers (`include_top=False`), we aimed for feature extraction rather than classification. The `input_shape` parameter was set to `(32, 32, 3)`, requiring resizing and stacking of MNIST grayscale images to match the expected three-channel input shape. The output from the base model was flattened using the `Flatten` layer, converting 3D feature maps into a 1D vector. Subsequently, we created a feature matrix using the vectors and generated a corresponding label matrix by encoding the label matrix, placing 1 in the position of the corresponding class and -1 elsewhere. Thus, the feature matrix X is of size 2000×4096 and the label matrix Y is of size 2000×10.

We then implemented functions to update Z, E, B, and J along with the update of the multipliers. Additionally, we enforced a 10 nearest neighbor condition for the adjacency matrix and determined the optimal values for the hyperparameters through grid search, with $\lambda_1 = 10000$, $\lambda_2 = 1000$, $\lambda_3 = 1$, and a kernel width of 1.

We investigated noise levels of 0.0, 0.2, 0.4, and 0.6 and compared the accuracy for the test and training sets for the LNSI algorithm. Furthermore, we evaluated the accuracy of SVM with an "rbf" kernel versus LNSI. We conducted this alongside LNSI to compare the accuracy of the two models in this context.

Note: The MNIST dataset consists of 10 classes, and we randomly sampled 200 datapoints from each class for our experiments.

7 Results

In this section, we present the outcomes of our experimental investigations aimed at assessing the accuracy of different algorithms under varying levels of noise. Table 1 displays the accuracy scores acquired across different noise rates.

Noise Rate	Train Accuracy	Test Accuracy
0.2	0.9675	0.932
0.6	0.814	0.798
0.4	0.931	0.8895
0	0.9815	0.953

Table 1: Accuracy Scores for Different Noise Levels for LNSI

As illustrated in Table 1, the noise rates range from 0 to 0.6, with each row representing a specific noise rate. The columns delineate the corresponding training and test accuracies. We observe that higher noise rates tend to coincide with diminished accuracy scores across both training and test datasets. This underscores the criticality of noise mitigation strategies in enhancing algorithmic robustness across various real-world scenarios.

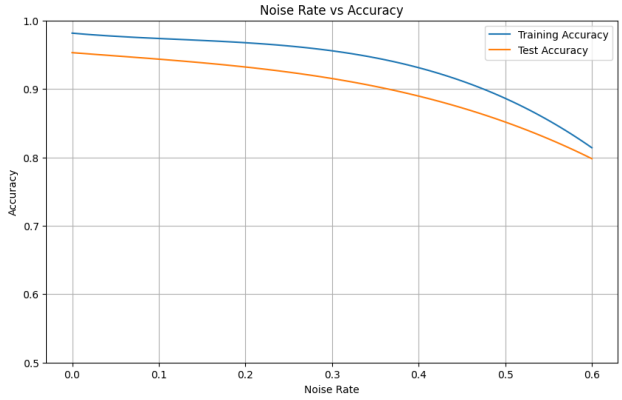

Figure 2: LNSI Performance on test and train noisy data

For demonstration purpose we have done a gridsearchCV and got the best hyperparameters here and tried to get the overview on the train and test data in the noisy condition using SVM with 'rbf' kerne;.

Noise Rate	Test Set Accuracy	Train Set Accuracy
0.2	0.2255	0.947
0.4	0.096	0.904
0.6	0.199	0.831

Table 2: Accuracy Scores for SVM with Different Noise Levels

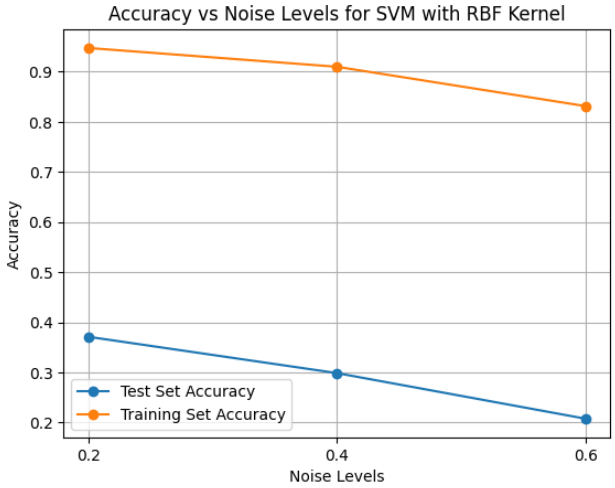

Figure 3: SVM with 'rbf' kernel Performance on test and train noisy data

8 Future Work

- **Firstly,** our primary focus will be on updating the current codebase to enhance its performance and compare the outcomes with those obtained from the μSGD model. This iterative process of refinement aims to bolster the accuracy and efficiency of our classification system, ensuring that it meets the highest standards of quality and reliability.

- **Secondly,** our paper relies on an assumption when defining the Laplacian regularizer, asserting that similar data points will exhibit similar labels. We intend to rigorously evaluate this assumption by exploring alternative methods for constructing the adjacency matrix, thereby gaining deeper insights into the underlying data structure.

- **Thirdly,** we plan to explore the impact of varying norms employed throughout the project. By experimenting with different norm functions, we aim to uncover potential avenues for improving the overall performance of our classification models and achieving better results.

9 Conclusion

As our project progressed until the mid-semester review, we encountered significant hurdles with traditional classification algorithms in handling label noise effectively. Despite their widespread use, these classic methods often falter when confronted with noisy labels, leading to suboptimal performance and unreliable results. Recognizing the limitations of these approaches, we embarked on a journey to explore alternative methodologies that could better address the challenges posed by label noise.One avenue we pursued was the adoption of μSGD for binary classification tasks. This novel approach involves the estimation of an unbiased estimator, representing a departure from conventional techniques. By incorporating this innovative method into our workflow, we aimed to overcome the shortcomings of traditional algorithms and enhance the robustness of our classification models.Furthermore, our exploration led us to investigate the concept of LNSI, which stands for Label Noise with Side Information. This technique capitalizes on the feature matrix as supplementary information to improve classification accuracy, particularly in scenarios involving multi-class classification tasks. By leveraging the additional insights provided by the feature matrix, LNSI offers a unique perspective on addressing label noise challenges, opening up new avenues for more accurate and reliable inference.Through our experimentation with these alternative methods, we have not only broadened our understanding of classification techniques but also gained valuable insights into the intricacies of handling label noise in real-world datasets. Moving forward, we are committed to further refining and optimizing these methodologies to develop robust and dependable classification systems capable of tackling the diverse challenges encountered in practical applications.

10 References

[**1**] Giorgio Patrini, Frank Nielsen, Richard Nock, Marcello Carioni. Loss Factorization, Weakly Supervised Learning and Label Noise Robustness. The 33rd International Conference on Machine Learning, in Proceedings of Machine Learning Research, 2016
(chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://proceedings.mlr.press/v48/patrini16.pdf)

[**2**] K.-Y. Chiang, C.-J. Hsieh, and I. S. Dhillon. Matrix completion with noisy side information. In Proc. Adv. Neural Inf. Process. Syst., 2015
(https://proceedings.neurips.cc/paper/2015/file/0609154fa35b3194026346c9cac2a248-Paper.pdf)

[**3**] R. Wang, T. Liu, and D. Tao. Multiclass learning with partially corrupted labels. IEEE Trans. Neural Netw. Learn. Syst., 2018
(https://ieeexplore.ieee.org/abstract/document/7929355)

[**4**] Kuan - Min Huang, Hooman samani, chang-yun yang yang and Jie - shengchen. Alternating Direction Method of Multipliers for Convex optimization in Machine Learning - interpretation and Implementation,2nd International Conference on Image processing and Robotics,2022.
(https://ieeexplore.ieee.org/document/9798720)

[**5**] Prof won Math ($https://youtu.be/M9F7zT9Gg-k?si=BQIkRM2batjPYVmc$) Accessed on 11th Feb,2024

[**6**] datacamp (https://www.datacamp.com/blog/classification-machine-learning) Accessed on 11th Feb,2024

[**7**] iMerit (https://imerit.net/blog/how-noisy-labels-impact-machine-learning-models/) Accessed on 11th Feb,2024

[**8**] Prof. Yi Ma (https://people.eecs.berkeley.edu/ yima/matrix-rank/home.html) Accessed on 12th Feb,2024

[**9**] Sciencedirect (https://www.sciencedirect.com/topics/mathematics/frobenius-norm) Accessed on 12th Feb,2024

[**10**] Wikipedia (https://en.wikipedia.org/wiki/Laplacian matrix) Accessed on 13th Feb,2024

[**11**] Yang Liu, Assistant Professor, Computer Science and Engineering UC Santa Cruz (https://www.youtube.com/watch?v=6tAHaO5GGqI) Accessed on 14th march,2024