Assignment 2

Tutors: Siyu Xu
Group members: Kaiyue Wang (ID: 530402325, UniKey: kwan5221), Peng Huang (ID: 530203700, UniKey: phua0038), Shoudi Huang (ID: 500478204, Unikey: shua9875)

Abstract

Convolutional neural networks (CNN) have made groundbreaking advances in the field of image recognition, improving the performance of classifiers effectively by learning large numbers of data. However, large-scale datasets hold the potential problem of label noise, which drastically degrade the performance of CNNs and results in a worse generalisation of the model. Learning with noisy labels is a challenging problem for current research. In this study, we aim to implement robust classifiers to learning on label noise datasets via existing label noise robust methods, Importance Reweighting and T-Reversion. We first designed a transition matrix estimator based on CNN classifier. Next, we evaluated the performance of the robust classifiers with known flip rates on the FashionMNIST0.5 and FashionMNIST0.6 datasets, and with unknown flip rates on the CIFAR dataset. The experimental results show that the performance of the robust classifiers consistently outperforms the baseline on all datasets. For datasets with unknown flip rates, the T-Revision method obtains better performance than the Importance Reweighting method. These findings highlight the importance of method selection for learning with noisy labels based on specific label noise conditions , and can be informative for future studies.

1 Introduction

In image recognition, the high cost of accurately annotating large datasets often leads to the prevalence of more affordable but noisily labeled datasets. This label noise is a significant concern in machine learning, dramatically affecting the accuracy and reliability of classifiers. As classifiers rely on correct labels to learn and make predictions, noisy labels can lead to incorrect learning and poor model performance. Therefore, investigating and assessing robust methods to combat label noise is crucial for enhancing the learning capabilities of machine learning models for image recognition.

The aim of this study is to develop classifiers robust to label noise and to assess their performance across various datasets contaminated by label noise. This study helps to explore and compare existing methods for their resilience to label noise, thereby advancing our understanding of how to maintain high performance in the presence of imperfect data labeling.

Firstly, we review the related work and methods for dealing with label noise. Then we decided that our study intends to evaluate two label noise robust classifiers, which use Importance Reweighting and T-Revision techniques, on three datasets with label noise: FashionMINIST0.5 and FashionMINIST0.6, which have known transition matrices, and CIFAR, which has an unknown transition matrix. Learning from noisy labeled data entails training within a noise domain and testing in a clean data domain, exemplifying domain adaptation, which Importance Reweighting is proven highly effective. Besides, T-Revision is selected over other loss correction methods, such as Backward or

Forward Correction, due to its demonstrated superiority in previous studies. We developed an estimator to determine the transition matrix for CIFAR. The effectiveness of this estimator was then assessed, and the performance of the label noise robust classifiers across datasets was compared using a range of metrics.

2 Related work

Various methodologies existed to mitigate the label noise, including Robust Architecture, Robust Regularization, Loss Adjustment, Sample Selection, or Robust Loss Function [11]. Loss Adjustment encompasses categories such as Loss Correction and Loss Reweighting. Loss Correction modifies each sample's loss by applying the estimated label transition probabilities to the model's predicted class probabilities. Loss Reweighting adjusts the objective function by assigning tailored weights to sub-objectives for each training sample, thereby addressing label noise.

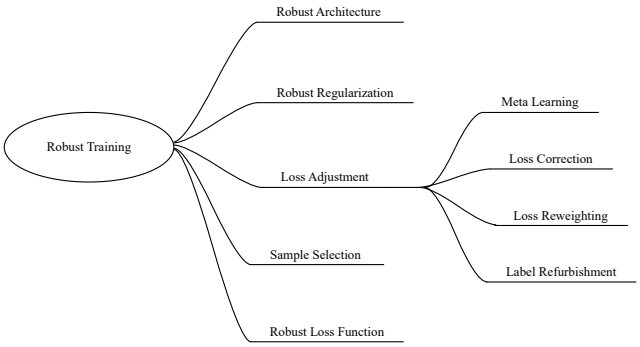

Figure 1: Related work

Backward Correction [7], a loss correction method, corrects model predictions by adjusting the output probabilities using the inverse transition matrix T^{-1} during the prediction phase. This technique directly transforms the model's output, aligning the predicted results with the actual label distribution.

Similarly, Forward Correction [7] applies the inverse transition matrix T^{-1} for the loss correction during the training phase, instead of adjusting the predicted probabilities output. The core idea is to adjust the loss function to account for the label noise, effectively training the model on what the noisy distribution of labels should look like if it were clean.

When severe noise exists, the transition matrix can not be accurately estimated from noisy data. Building on Forward Correction, Gold Loss Correction [3] was proposed for handling severe noise, assuming a small trusted data subset is available.

One key disadvantage of these methods is their reliance on accurately estimating the transition matrix. Acquiring this matrix typically necessitates clean validation data or anchor points, which are often unavailable in real-world problem.

Hence, T-Revision [12] has been introduced to learn the noise transition matrix without needing anchor points. It begins by initializing a transition matrix using examples with high noisy class posterior probabilities and iteratively refines this matrix by incorporating a slack variable.

In addition to Loss Correction techniques for label noise robustness, Loss Reweighting methods also offer a viable approach. For instance, DualGraph [13] leverages graph neural networks to reweight examples based on structural label relations, effectively isolating and removing abnormal noise examples.

Besides, Chang et al. introduced Active Bias [1], which focuses on examples with uncertain and inconsistent label predictions by using their prediction variances as weights during training.

Similarly, Liu et al. introduced Importance Reweighting [6], the weight of each noisy example in the loss function is determined by a ratio (Eq.2), which represents the relative distribution of joint data under clean and noisy conditions.

3 Method

3.1 Pre-processing

This study employs Min-Max Scaling to normalize image data, a process that involves first determining the minimum $\min(x)$ and maximum $\max(x)$ values in the dataset. Subsequently, the data are rescaled based on these identified extreme values, as shown in Eq.1.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{1}$$

For the image data used in this study, pixel values are limited to a discrete spectrum from 0 to 255. Within this scope, Min-Max Scaling is streamlined to divide each pixel's intensity by 255, thereby rescaling these values to fall within a $[0, 1]$ interval.

This normalization improves the classifier's numerical stability, computational efficiency and accelerates algorithmic convergence.

3.2 Classification Model

CNN is a type of deep learning model for processing data especially for the image data. Its architecture consists of convolutional, pooling and fully connected layers. There are many widely used CNN architectures designed, such as AlexNet, VGG and ResNet [4, 10, 2], which have achieved good accuracy in many practices. However, these models use a large number of parameters, which leads to high computational costs.

Therefore, considering the size and characteristics of the dataset images used for the experiments. We prefer to reference the architecture of LeNet-5 [5] and design model architectures suitable for the FashionMNIST dataset and the CIFAR dataset, as shown in Figures 2, 3. We validated the capability of these architectures by cross-validation with the FashionMNIST and CIFAR training sets.

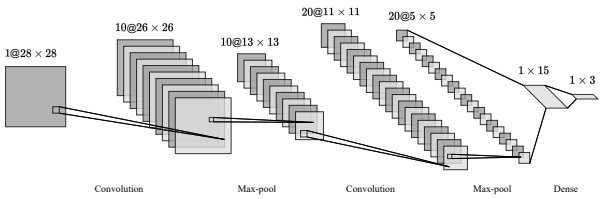

Figure 2: The CNN architecture used for the FashionMNIST dataset

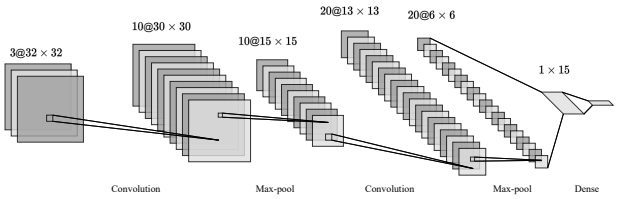

Figure 3: The CNN architecture used for the CIFAR dataset

The CNN model extracting spatial features through convolutional layers by using the 3×3 size convolutional kernel with stride of 1 and no padding. Reducing and controlling overfitting problem through max pooling layers by using the 2×2 filter, with stride of 2. Performing advanced

combination of features through fully connected layers, and get the finally output prediction. The difference between the two classifier models is basically the number of input channels and the size of the images. By using fewer model parameters, the possibility of overfitting can be reduced, and the model can also be trained faster.

3.3 Label Noise Robust Methods

This section presents two methods for label noise robustness integrated into the CNN model to enhance its resistance to label noise.

3.3.1 Importance Reweighting

Importance Reweighting [6] is a technique for learning with label noise, drawing from domain adaptation principles, viewing noisy training data as the source domain ($\overline{D}$), and clean test data as the target domain (D). This method reweights the loss corresponding to each noise sample so that the model biases the sub-objectives to overcome label noise. The weight given to a noisy example $(X, \overline{Y}) \sim \overline{D}$ can be derived by exploiting the inversed noise rates, as shown in Eq.2, where T is the transition matrix

$$\beta(X, \overline{Y}) = \frac{P_D(Y|X)}{P_{\overline{D}}(\overline{Y}|X)} \approx \frac{P_{\overline{D}}(\overline{Y}|X)}{T^\top P_{\overline{D}}(\overline{Y}|X)} \tag{2}$$

Then the classifier can therefore be learned by minimizing the following objective function (Eq. 3),

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \beta(X_i, \overline{Y}_i) \ell(f(X_i), \overline{Y}_i) \tag{3}$$

where ℓ is the Cross-Entropy loss function (Eq.4).

$$\ell(f(X), \bar{Y}) = -\sum_{i=1}^{C} 1_{\bar{Y}=i} \log(P_{\overline{D}}(\bar{Y}|X)) \tag{4}$$

The training procedure for the classifier using Importance Reweighting is outlined in Algorithm 1 and Figure 4. Note that when training on datasets with a known transition matrix T, such as the FashionMNIST datasets, Stage 1 is omitted. Instead, the classifier is directly initialized and trained using the weighted loss as specified in Eq.3.

Algorithm 1: Traning process for classifier employing Importance Reweighting

Result: f
Input: Noisy training sample $\overline{D}_t$; Noisy validation set $\overline{D}_v$.
Stage 1: Initialize the classifier f and estimate $\hat{T}$
1. Initialize the CNN with unweighted loss;
2. Estimate the transition matrix $\hat{T}$ as described in Section 3.4;
Stage 2: Learn the classifier f
3. Continue to Train f by minimizing the weighted loss (Eq.3) using $\hat{T}$;

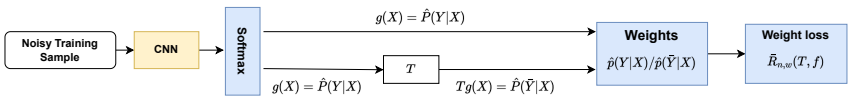

Figure 4: Importance Reweighting method

3.3.2 T-revision

T-Revision [12] is an effective method to learn transition matrix without known anchor points. It initiated the transition matrix $\hat{T}$ using data points resembling anchor points with high noisy class

posterior probabilities, treating this as prior knowledge. Subsequently, it refined $\hat{T}$ by integrating a learned slack variable ΔT, validated alongside the classifier with noisy data.

The T-Revision method works because it learns ΔT by the objective function for minimizing the risk-consistent estimator (Eq.5), which is asymptotically equal to the expected risk with respect to clean data [12].

$$\arg\min_{\hat{T}+\Delta T, f} \frac{1}{n} \sum_{i=1}^{n} \frac{g_{\bar{Y}_i}(X_i)}{((\hat{T}+\Delta T)^\top g)_{\bar{Y}_i}(X_i)} \cdot \ell(f(X_i), \bar{Y}_i) \tag{5}$$

Where $g(x) = \hat{P}(Y|X = x)$ is used for approximating $P(Y|X = x)$ and $(\hat{T} + \Delta T)^\top g(x) = \hat{P}(\bar{Y}|X = x)$ is used for approximating $P(\bar{Y}|X = x)$. The weight $\hat{P}(Y|X = x)/\hat{P}(\bar{Y}|X = x)$ is for the Cross-Entropy loss function ℓ (Eq.4).

The training process for the classifier using T-Revision is outlined in Algorithm 2 and Figure 5. It is segmented into three stages, enhancing the interpretability of the report. This differs from the original two-stage training described by [12]. Note that when training on datasets with a known transition matrix T, the FashionMNIST datasets, the initial stage is bypassed. Training commences directly from stage 2, with the classifier being initialized and trained accordingly.

Algorithm 2: Traning process for classifier employing T-Revision

Result: $\hat{T}$, ΔT, and f.
Input: Noisy training sample $\overline{D}_t$; Noisy validation set $\overline{D}_v$.
Stage 1: Initialize the classifier f and estimate $\hat{T}$
1. Initialize the CNN with unweighted loss;
2. Estimate the transition matrix $\hat{T}$ as described in Section 3.4;
Stage 2: Learn the classifier f
3. Learn f by minimizing the weighted loss (Eq.3) with a noisy adaption layer $\hat{T}^\top$;
Stage 3: Refine the classifier f and learn ΔT
4. Minimize the T-Revision weighted loss (Eq.5) to learn f and ΔT with a noisy adaption layer $(\hat{T} + \Delta T)^\top$;

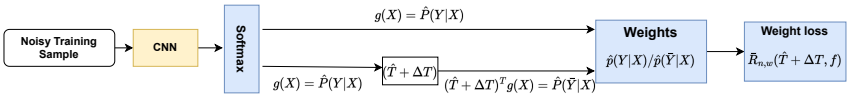

Figure 5: T-Revision method

3.4 Noise Rate Estimation Method

The transition matrix serves as a link between the class posterior probabilities of noisy and clean data, enabling the construction of a risk-consistent estimator through loss correction. An estimator is deemed risk-consistent when, as the size of noisy samples increase, the empirical risk derived from these noisy samples and a modified loss function converges to the expected risk derived by clean samples with the unaltered loss function [12].

The effectiveness of consistent algorithms hinges on the accurate learning of transition matrices. To learn the transition matrix, anchor points within the clean data domain are estimated using the noisy data. An instance x qualifies as an anchor point for class i if $P(Y = i|x)$ approximates one, indicating a high likelihood that Y is in the i^{th} class. When $P(Y = i|x) = 1$ for a given x, it implies $P(Y = k|x) = 0$ for all $k \neq i$. Thus, the transition matrix can be formalized as shown in Eq.6.

$$P(\bar{Y} = j|x) = \sum_{k=1}^{C} T_{kj} P(Y = k|X = x) = T_{ij}. \tag{6}$$

In our experiment, T is formulated as a 3x3 matrix (Eq.7), corresponding to the three categories, 0, 1, and 2, present in each dataset. Each element $P(\bar{Y} = j|Y = i)$ denotes the probability that a true label i is observed as a noisy label j in the dataset.

$$T = \begin{bmatrix} P(\bar{Y}=0|Y=0) & P(\bar{Y}=0|Y=1) & P(\bar{Y}=0|Y=2) \\ P(\bar{Y}=1|Y=0) & P(\bar{Y}=1|Y=1) & P(\bar{Y}=1|Y=2) \\ P(\bar{Y}=2|Y=0) & P(\bar{Y}=2|Y=1) & P(\bar{Y}=2|Y=2) \end{bmatrix} \tag{7}$$

The training process for the transition matrix estimator is outlined in Algorithm 3.

Algorithm 3: Training Procedure for estimating transition matrix

Result: T and f
Input: Noisy training sample D_t.
1. Minimize the unweighted loss to learn $P(\bar{Y}|X = x)$ with no noise adaption layer;
2. Initialize T according to Eq.6, utilizing instances that exhibit the greatest $P(\bar{Y} = i|X = x)$ to define anchor points for the i^{th} class.

4 Experiments and Discussion

In the previous work, some experiments have presented the performance of the specific method on certain noise label datasets. However, due to differences in datasets, levels of flip rates, and different classifier models used in the various researches, it is hard to determine which method is superior compared to the others.

Therefore, we train the robustness classifiers with Importance Reweighting and T-Revision using three datasets with known and unknown flip rates. After that, we will perform a side-by-side comparison using four evaluation metrics and analyse the robustness of the different methods.

4.1 Experimental setup

We designed an experimental process consisting of 5 phase, as shown in Figure 6.

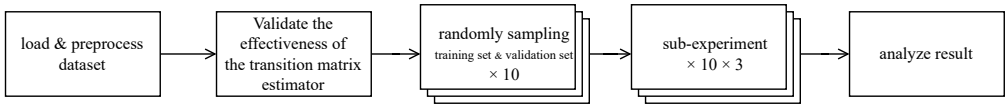

Figure 6: The phases of the experiment

In the first phase, we load three datasets with integer data type, and since floating point numbers are helpful for the training of neural networks, we pre-process the datasets using the normalisation method (Eq 1) to obtain floating point type datasets. The experiment uses the following datasets.

- The FashionMNIST datasets are grey-scale image datasets with an image size of 28×28 and contain 3 categories. All samples are well pre-processed, centred and have white background. Both of FashionMNIST0.5 and FashionMNIST0.6 have a training set (Figure 7) of 18000 samples and their labels have been introduced with noise based on a known flip rate. However, FashionMNIST0.6 has a higher probability of flip rate than FashionMNIST0.5, which means that the labels in the training set are more likely to be wrong. Both of their test sets have 3000 samples with clean labels.

Figure 7: Samples of FashionMNIST

- The CIFAR dataset is a 3-channel colour image dataset with an image size of 32×32 and 3 categories. Compared to FashionMNIST, the targets appear in the images with diverse positions, viewpoints and sizes and with various backgrounds. Its training set (Figure 8) has 15000 samples with noisy labels of unknown flip rate. Its test set has 3000 samples with clean labels.

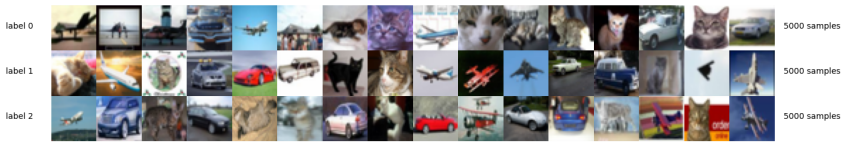

Figure 8: Samples of CIFAR

In the second phase, we validate the effectiveness of the transition matrix estimator by comparing the estimated transition matrices and the corresponding transition matrices with the known flip rate, and calculating the gap between them by mean absolute error (MAE) as in Eq 8. Where t is the known flip rate, $\hat{t}$ is the estimated flip rate and c is the number of classes. We use the training sets of FashionMNIST0.5 and FashionMNIST0.6 to learn the transition matrix estimators respectively, recording the validation accuracy and model parameters during each epoch. We then select the model parameter with the highest validation accuracy and use it to estimate the transition matrices.

$$\text{MAE} = \frac{1}{c \times c} \sum_{i=1}^{c} \sum_{j=1}^{c} |t_{i,j} - \hat{t}_{i,j}| \tag{8}$$

In the third phase, for a rigorous experiment, we shuffled the entire training set and randomly selected 80% of the samples as the training set and 20% of the samples as the validation set, making 10 combinations in total. By using different combinations of training and validation sets, we can observe the stability of the performance of classifiers trained with different robustness methods.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

$$\text{Top 1 accuracy} = \frac{TP + TN}{\text{test sample size}} \tag{12}$$

In the fourth phase, we performed the following two types of sub-experiments based on a specific dataset. All combinations of training and validation sets obtained in the previous stage are individually input to the sub-experiments for training and to generate a model for predicting the labels of the test set. We compare the predicted labels with the true labels and record the results of the sub-experiments using 4 evaluation metrics: Recall (Eq. 9) indicates the ability of the model to identify all samples in a specific category, Precision (Eq. 10) indicates the ability of the model to return only the correct samples in a specific category, F1 score (Eq. 11) is the combination of the precision and recall with harmonic mean and Top 1 accuracy (Eq. 12) indicates the ability of the accuracy of the highest probability of the prediction by the model. Where True Positive (TP) is the category predicted correct with the corresponding positive label, False Positive (FP) is the category predicted incorrect with the corresponding positive label, True Negative (TN) is the category predicted correct with the corresponding negative label, False Negative (FN) is the category predicted incorrect with the corresponding negative label.

For the sub-experiments with known noise label flip rate, based on FashionMNIST0.5 dataset and FashionMNIST0.6 dataset. We directly use the known transition matrix (Figures 12, 13) to train the robust classifier models, and evaluate by the test set, as shown in Figure 9.

- For using the T-Revision method to train the model, there are 2 stages: 1st, train with Importance Reweighting loss, record the model parameters and validation accuracy for each epoch. 2nd, choose the model parameters with the highest validation accuracy, then train with T-Revision loss and output the final model.

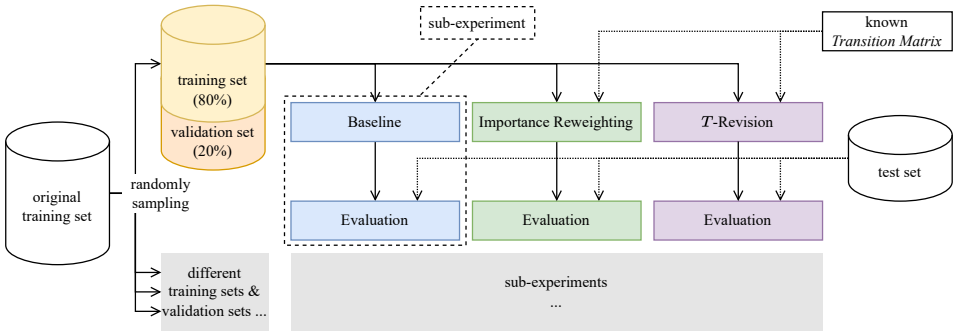

Figure 9: Sub-experiments on FashionMNIST

For the sub-experiments with unknown noise label flip rate, based on CIFAR dataset. We first trained an estimator model and use the training set data to estimate the transition matrix by using the same approach as in the second phase. The estimated transition matrix is then used in robust learning methods to train classifier models, and evaluate by the test set, as shown in Figure 10.

- For using the Importance Reweighting method to train the model, there are 2 stages: 1st, obtain the estimator model and the transition matrix. 2nd, train with Importance Reweighting loss with the estimated transition matrix and output the final model.
- For using the T-Revision method to train the model, there are 3 stages: 1st, obtain the estimator model and the transition matrix. 2nd, train with Importance Reweighting loss with the estimated transition matrix, record the model parameters and validation accuracy for each epoch. 3rd, choose the model parameters with the highest validation accuracy, then train with T-Revision loss with the estimated transition matrix and output the final model.

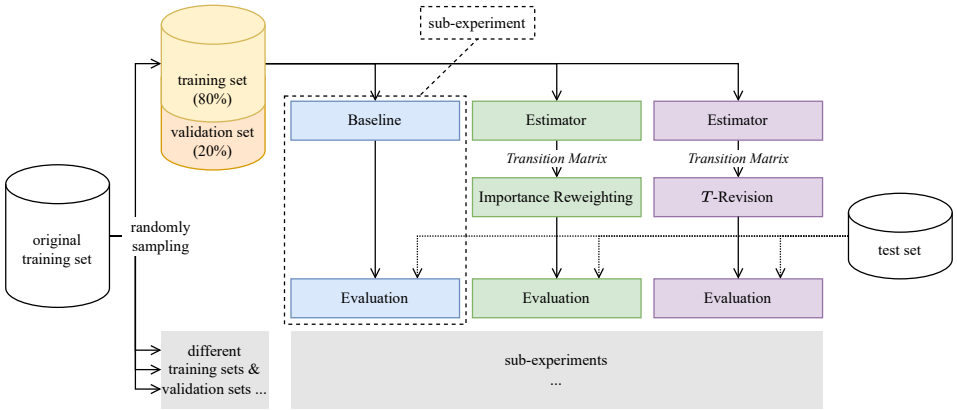

Figure 10: Sub-experiments on CIFAR

In the fifth phase, we collect the results of all the sub-experiments including Recall, Precision, F1 score and Top 1 accuracy scores of each robust classifier on different training and validation sets and calculate their standard deviation and mean (Figure 11).

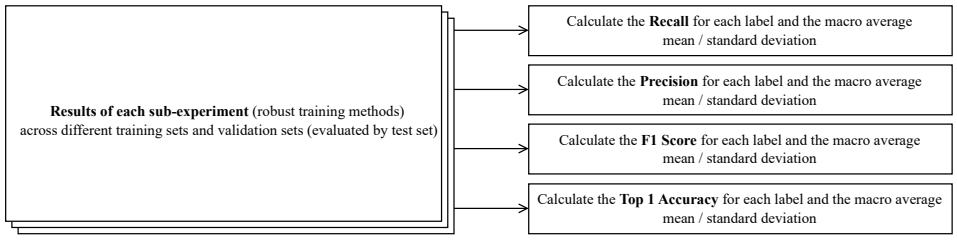

Figure 11: Analyze results

4.2 Results on Transition Matrix Estimator

We compare the given transition matrix and the transition matrix we estimated. The hyperparameters of the Baseline classifier we used to estimate transition matrix can be found in Table 1. We evaluate the transition matrix on FashionMINIST0.5 and FashionMINIST0.6 and evaluate the performance by mean absolute error (MAE). The MAE on the FashionMINIST0.5 dataset is 0.0837, on the FashionMINIST0.6 dataset is 0.0143. The error is relatively small which means the estimation of the classifier is close to the known transition matrix. In Figure 12 and Figure 13, T is the known transition matrix, $\tilde{T}$ is the estimated transition matrix and $T - \tilde{T}$ is the error between them.

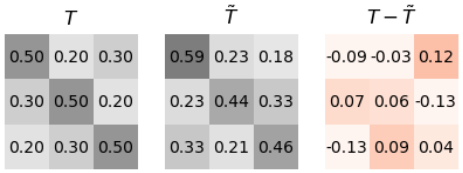

Figure 12: Estimated Transition matrix on FashionMINIST 0.5

Figure 13: Estimated Transition matrix on FashionMINIST 0.6

4.3 Results on FashionMINIST0.5

The configuration settings for two label noise-robust classifiers, a CNN model with Importance Reweighting and a CNN model with T-Revision, alongside a baseline CNN model on the FashionMNIST0.5 dataset, are presented in Table 1. The hyperparameters applied to these classifiers were fine-tuned.

The results obtained by the classifiers on the FashionMNIST0.5 dataset have been summarized in Figure 14, which exhibits the Top 1 accuracy, and in Table 2, which presents the Recall, Precision, and F1 Score. These results have been compared between two label noise robust classifiers and a baseline classifier.

Figure 14 illustrates that the label noise robust classifiers outperform the baseline in Top 1 accuracy with means above 90% and notably lower variance, compared to the baseline's 87.5%. The

	Baseline	**Importance Reweighting**	T**-Revision**	
		Stage 2	Stage 2	Stage 3
Epoch	20	20	20	20
Learning Rate	0.1	0.01	0.01	0.000001
Momentum	0.9	0.9	0.9	-
Weight Decay	0.0001	0.0001	0.0001	0.0001
Loss Function	Cross-Entropy	Importance Reweighting	Importance Reweighting	T-Revision
Optimizer	SGD	SGD	SGD	Adam

Table 1: Configuration settings for all classifiers on FashionMINIST0.5 and FashionMINIST0.6

T-Revision classifier displayed a marginal advantage over the Importance Reweighting classifier, indicated by a marginally greater average accuracy and a more constrained variation range despite a single outlier.

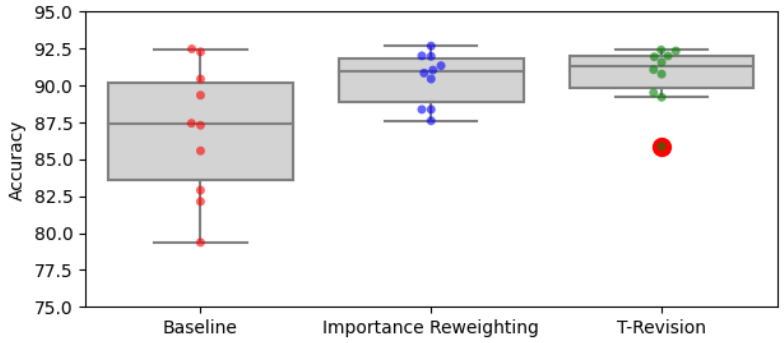

Figure 14: Top 1 accuracy for all classifiers on FashionMNIST0.5

Table 2 demonstrates that the label noise-robust classifiers markedly exceeded the baseline classifier's performance across all metrics, Recall, Precision, and F1 Score, for individual classes and the macro average, considering both mean values and standard deviations. Furthermore, the T-Revision classifier slightly surpassed the Importance Reweighting classifier in these metrics, particularly in terms of the mean F1 scores for individual classes and the macro averages across all metrics. However, the Importance Reweighting classifier demonstrated superior performance regarding the standard deviations across all metrics.

Table 2: Recall, Precision and F1 score for all classifiers on FashionMINIST0.5

	Baseline		**Importance Reweighting**		T**-Revision**	
	mean	std	mean	std	mean	std
Recall 0	0.8484	0.0917	0.8615	0.0364	0.8790	0.0602
Recall 1	0.9100	0.0734	0.9897	0.0057	0.9830	0.0148
Recall 2	0.8864	0.0552	0.8771	0.0231	0.8747	0.0338
Precision 0	0.8773	0.1084	0.9571	0.0091	0.9351	0.0331
Precision 1	0.9308	0.0261	0.9017	0.0275	0.9165	0.0366
Precision 2	0.7995	0.1464	0.8550	0.0442	0.8683	0.0725
F1 score 0	0.8527	0.0529	0.9062	0.0168	0.9039	0.0207
F1 score 1	0.9188	0.0436	0.9434	0.0137	0.9479	0.0160
F1 score 2	0.8284	0.0820	0.8653	0.0267	0.8690	0.0320
Recall macro average	0.8816	0.0314	0.9094	0.0140	0.9122	0.0162
Precision macro average	0.8692	0.0419	0.9046	0.0166	0.9066	0.0192
F1 score macro average	0.8666	0.0453	0.9049	0.0167	0.9069	0.0194
Top 1 accuracy macro average	0.8692	0.0419	0.9046	0.0166	0.9118	0.0152

4.4 Results on FashionMINIST0.6

The classifiers' configuration settings for the FashionMNIST0.6 dataset were inherited from those of the FashionMNIST0.5 dataset, as detailed in Table 1. Figure 15 and Table 3 summarized the results obtained by the classifiers on the FashionMNIST0.6 dataset.

Regarding Top 1 accuracy, the label noise robust classifiers continue to outperform the baseline classifier on the FashionMNIST0.6 dataset in both mean values and standard deviations, as depicted in Figure 15. The label noise robust classifiers maintained a mean accuracy of around 90%, whereas the baseline classifier's mean accuracy declined to around 75%. The Importance Reweighting and T-Revision classifiers exhibited comparable performances in terms of mean values and standard deviations of Top 1 accuracy, where T-Revision had a slightly wider variation span.

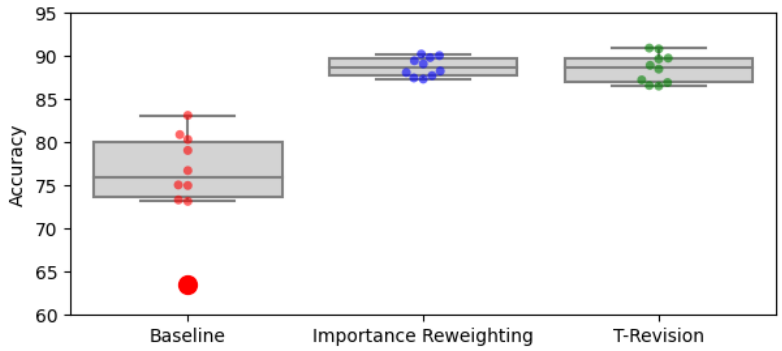

Figure 15: Top 1 accuracy for all classifiers on FashionMNIST0.6

Similar to their performance on the FashionMNIST0.5 dataset, the label noise robust classifiers notably outperformed the baseline classifier in Recall, Precision, and F1 score on the Fashion-MNIST0.6 dataset. The label noise robust classifiers displayed comparable performance, with the Importance Reweighting classifier slightly edged in Precision and F1 score, while T-Revision marginally led in Recall, considering both means and standard deviations. Details are presented in Table 3.

Table 3: Recall, Precision and F1 score for all classifiers on FashionMINIST0.6

	Baseline		**Importance Reweighting**		**T-Revision**	
	mean	std	mean	std	mean	std
Recall 0	0.8605	0.0776	0.8975	0.0351	0.8935	0.0538
Recall 1	0.8835	0.1135	0.9670	0.0299	0.9804	0.0100
Recall 2	0.6749	0.1054	0.8159	0.0318	0.8143	0.0600
Precision 0	0.6350	0.1724	0.8575	0.0434	0.8565	0.0670
Precision 1	0.8169	0.1147	0.9198	0.0222	0.9128	0.0217
Precision 2	0.8291	0.1517	0.8855	0.0543	0.8885	0.0725
F1 score 0	0.7114	0.1101	0.8755	0.0144	0.8707	0.0210
F1 score 1	0.8356	0.0633	0.9422	0.0087	0.9452	0.0107
F1 score 2	0.7229	0.0459	0.8473	0.0164	0.8449	0.0232
Recall macro average	0.8063	0.0306	0.8935	0.0120	0.8961	0.0105
Precision macro average	0.7603	0.0524	0.8876	0.0105	0.8859	0.0161
F1 score macro average	0.7566	0.0559	0.8883	0.0110	0.8869	0.0159
Top 1 accuracy macro average	0.7603	0.0524	0.8876	0.0105	0.8859	0.0161

4.5 Results on CIFAR

Table 4 presents the configuration settings for two label noise robust classifiers, a CNN model with Importance Reweighting and a CNN model with T-Revision, as well as a baseline CNN model as applied to the CIFAR dataset.

	Baseline	**Importance Reweighting**		**T-Revision**		
		Stage 1	Stage 2	Stage 1	Stage 2	Stage 3
Epoch	20	20	20	20	20	20
Learning Rate	0.1	0.1	0.01	0.1	0.01	0.000001
Momentum	0.9	0.9	0.9	0.9	0.9	-
Weight Decay	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Loss Function	Cross-Entropy	Cross-Entropy	Importance Reweighting	Cross-Entropy	Importance Reweighting	T-Revision
Optimizer	SGD	SGD	SGD	SGD	SGD	Adam

Table 4: Configuration settings for all classifiers on CIFAR

Note that when applying Importance Reweighting or T-Revision classifiers to the CIFAR dataset, which has an unknown transition matrix, each run commences at Stage 1 to estimate the transition matrix as described in Algorithm 1 or 2, respectively. Thus, 10 iterations of each classifier will yield 20 estimated transition matrices. The averaged values and standard deviations (std) of these matrices are depicted in Figure 16, with comprehensive data on each provided in Appendix: Figures.

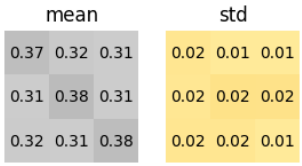

Figure 16: Mean values of estimated transition matrixs on CIFAR

Figure 17 shows that on the CIFAR dataset with unknown noise label flip rates, the label noise robust classifiers continue to outperform the baseline in Top 1 accuracy, maintaining a mean accuracy over 55% with a narrower variance compared to the baseline's around 50% with broader variance. The T-Revision classifier, with a marginally higher mean accuracy and similar variance than the Importance Reweighting classifier, outperforms all in Top 1 accuracy.

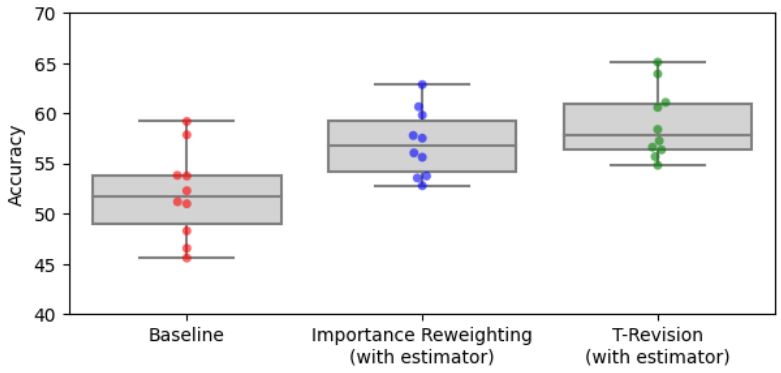

Figure 17: Top 1 accuracy for all classifiers on CIFAR

The label noise robust classifiers surpass the baseline across all other metrics as well, including Recall, Precision, and F1 score, when considering both means and standard deviations. The T-Revision classifier outperformed the Importance Reweighting classifier in the macro averages of these metrics. Notably, the Importance Reweighting classifier exhibits marginally lower standard deviations for Precision and F1 score. The specifics are detailed in Table 5.

Table 5: Recall, Precision and F1 score for all classifiers on CIFAR

	Baseline		**Importance Reweighting**		**T-Revision**	
	mean	std	mean	std	mean	std
Recall 0	0.5193	0.0716	0.6569	0.1144	0.6723	0.1199
Recall 1	0.5527	0.0685	0.5610	0.0657	0.6263	0.1022
Recall 2	0.5322	0.0786	0.5563	0.0519	0.5546	0.0477
Precision 0	0.5546	0.1218	0.5625	0.1413	0.6449	0.1347
Precision 1	0.5132	0.1992	0.4894	0.1493	0.3966	0.1519
Precision 2	0.4901	0.1588	0.6587	0.1760	0.7273	0.1333
F1 score 0	0.5278	0.0751	0.5806	0.0820	0.6352	0.0344
F1 score 1	0.4974	0.1398	0.5093	0.0922	0.4628	0.1174
F1 score 2	0.4898	0.0880	0.5855	0.0776	0.6217	0.0538
Recall macro average	0.5348	0.0463	0.5914	0.0401	0.6177	0.0380
Precision macro average	0.5193	0.0423	0.5702	0.0315	0.5896	0.0336
F1 score macro average	0.5050	0.0445	0.5584	0.0376	0.5732	0.0443
Top 1 accuracy macro average	0.5193	0.0423	0.5702	0.0315	0.5896	0.0336

4.6 Discussion

The label noise robust classifiers using Importance Reweighting or T-Revision consistently outperform the baseline on all datasets, FashionMNIST0.5, FashionMNIST0.6, and CIFAR. With higher noise rates in the datasets, the advantage of the noise-robust classifiers becomes increasingly pronounced. This aligns with expectations that classifiers employing the label noise robust methods would exhibit greater resilience to such noise.

Moreover, the Importance Reweighting and T-Revision classifiers exhibit similar performance on the FashionMNIST0.5 and FashionMNIST0.6 datasets. However, on the CIFAR dataset, with unknown label noise flip rates, the T-Revision classifier surpasses the Importance Reweighting classifier. This enhanced performance can likely be attributed to T-Revision's mechanism for refining the initial noise transition matrix by introducing a slack variable (ΔT), which is optimized alongside the classifier using noisy data. This adaptability allows T-Revision to effectively learn the transition matrix without anchor points, offering a significant edge on datasets where the noise transition matrix is unknown.

The transition matrix estimator's efficacy is evidenced by the low MAE between the estimated and actual transition matrices for FashionMINST0.5 and FashionMINST0.6 datasets, along with the low standard deviations observed for the estimated transition matrices on the CIFAR dataset. However, despite FashionMINST0.6's higher noise rate, its estimated transition matrix exhibits a lower MAE compared to that of FashionMINST0.5. This may result from FashionMINST0.5's broader diversity in noise flip rates, 0.2, 0.3, and 0.5, as opposed to FashionMINST0.6's narrower span, 0.3 and 0.4.

5 Conclusion and Future work

5.1 Conclusion

This study designed and implemented two CNN models tailored for the FashionMNIST and CIFAR datasets. The study's objective was to assess the performance of label noise robustness of each model by integrating two distinct label noise robust methodologies, namely Importance Reweighting and T-Revision. These methods were benchmarked against a baseline classifier, a model that did not incorporate any label noise robust method. The comparative analysis assessed label noise robust classifiers across datasets, two with known transition matrices and one with an unknown transition matrix, to compare their resistance to label noise. Additionally, the study assessed the transition matrix estimator's effectiveness and subsequently applied the estimator to the dataset with the unknown transition matrix.

Experimental results suggest that label noise robust classifiers consistently outperformed the baseline across datasets. The T-Revision classifier excels over Importance Reweighting on the dataset with an unknown noise transition matrix and matches its performance when the transition matrix is known. Besides, the transition matrix estimator, utilizing the baseline CNN model, proved effective, as evidenced by the comparative analysis of estimated versus actual transition matrices within the FashionMNIST datasets.

Overall, the two label noise robust classifiers developed in this study and the transition matrix estimator demonstrated resilience to label noise and effectiveness, respectively.

5.2 Future work

Considering the inherent limitation of Importance Reweighting and T-Revision dealing with label noise, in future we plan to make improvement on constructing label noise robust classifiers, studing on alternative loss correction or reweighting techniques like Gold Loss Correction [3] and Active Bias [1]. Additionally, Loss Adjustment strategies from the domains of label Refurbishment or Meta-Learning, such as Bootstrapping [8] or Automatic Reweighting [9], can be applied. Beyond Loss Adjustment, there are options like Robust Regularization as well. It is meaningful for us to investigate and discuss the performance of those method between the experiment we have already done, so it will help us to uncover which method has the better performance dealing with the label noise.

References

[1] Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. Active bias: Training more accurate neural networks by emphasizing high variance samples. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[3] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[5] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998.

[6] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):447–461, 2016.

[7] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[8] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. 12 2014.

[9] Mengye Ren, Wenyuan Zeng, Binh Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, 2018.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[11] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11):8135–8153, 2023.

[12] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[13] HaiYang Zhang, XiMing Xing, and Liang Liu. Dualgraph: A graph-based method for reasoning about label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9654–9663, June 2021.

Appendix: Figures

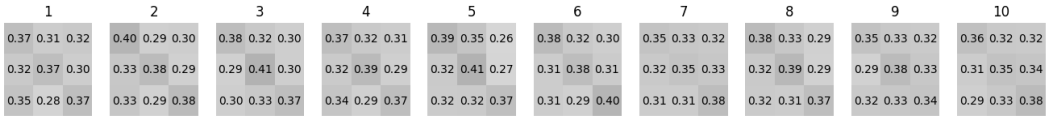

Figure 18: Importance Reweighting Classfier's estimated transition matrixs on CIFAR

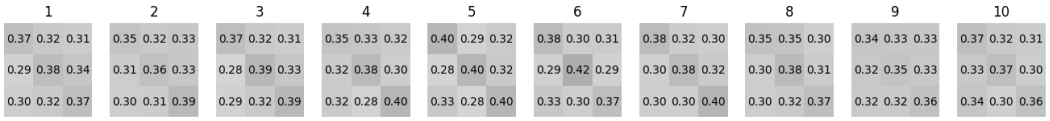

Figure 19: T-Revision Classfier's estimated transition matrixs on CIFAR

Appendix: Running the Code

The experiment code is compatible for CPU and GPU environments, and it is also supported to run on Colab with Google Drive mounted. For easy presentation of the output results, we employed the Jupyter Notebook format. The `main.ipynb` contains all experimental content from this report, segmented into 6 sections.

1. Initialization
2. Prepare Datasets
   (a) Load dataset
   (b) Display datasets samples
   (c) Pre-processing
   (d) Function for split datasets into training and validation sets
3. Classifier Models
   (a) CNN model for FashionMNIST0.5 and FashionMNIST0.6
   (b) CNN model for CIFAR
4. Transition Matrix Estimator
   (a) Function for estimate transition matrix
   (b) FashionMNIST0.5 dataset's transition matrix
   (c) FashionMNIST0.6 dataset's transition matrix
   (d) CIFAR dataset's transition matrix
5. Label noise robust methods
   (a) Importance Reweighting
   (b) T-Revision
6. Experiments
   (a) Function for running experiments
   (b) Experiments on FashionMNIST0.5 Dataset
   (c) Experiments on FashionMNIST0.6 Dataset
   (d) Experiments on CIFAR Dataset

Setting `ENABLE_xxx=True` will enable the code to be output for each section.

Please load this file into your Jupyter Notebook environment and execute the `run all` command to run the complete experiment.

Using the default parameters, it will take about 6 hours (based on a laptop with an AMD Ryzen 4800u CPU).