A Theoretical Analysis

In this section, we provide a theoretical analysis of our proposed FasTEN. First, we formally establish the need for label correction (Appendix A.1). Then, we provide a formal background for a statistically consistent classifier (Appendix A.2) and show the detailed calculation on the estimation of transition matrix $\boldsymbol{T}$ through forward propagation (Appendix A.3). Finally, we prove that $\boldsymbol{T}$ estimation error of our method are upper bounded (Appendix A.4).

A.1 Motivation: Need for Label Correction

Reducing the noise level of a dataset is crucial in learning with noisy labels both empirically [19, 121, 105, 40] and theoretically [14, 12]. Following [14, 80, 44, 64, 33], the true transition matrix $\boldsymbol{T}$ in *symmetric noise* with noise level $\gamma\,(<1)$ is defined as follows:

$$T_{ij} = \begin{cases} 1 - (^{(N-1)}/_{N})\,\gamma, & \text{for } i = j. \\ \gamma/_{N}, & \text{otherwise.} \end{cases} \tag{8}$$

Correspondingly, the true transition matrix $\boldsymbol{T}$ in *asymmetric noise* with noise level $\gamma\,(<{}^{1}/_{2})$ is defined as follows:

$$T_{ij} = \begin{cases} 1-\gamma, & \text{for } i = j. \\ \gamma, & \text{for some } i \neq j. \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

We employ these noise schemes in our CIFAR-10 experiments. Under the assumption that the label class is balanced, [14] prove that the upper bound of test accuracy for the symmetric and asymmetric noise is as follows:

$$\begin{cases} (^{(N-1)}/_{N})\,\gamma^2 - 2\,(^{(N-1)}/_{N})\,\gamma + 1, & \text{for } \textit{symmetric noise.} \\ 2\gamma^2 - 2\gamma + 1, & \text{for } \textit{asymmetric noise.} \end{cases} \tag{10}$$

Eq. 10 shows quadratic (convex) functions of γ. In the case of the symmetric noise, test accuracy is minimized at the $\gamma = 1$. Similarly, with asymmetric noise, test accuracy is minimized at the $\gamma = {}^{1}/_{2}$. Without additional assumptions, asymmetric noise of $\gamma > {}^{1}/_{2}$ cannot be learned as over half of the training data have wrong labels [33]. Hence, the test accuracy always decreases as γ increases in the feasible bound of γ. Therefore, reducing the noise level of a dataset plays a vital role in increasing the achievable test accuracy.

How to Reduce the Noise Level of a Dataset Two approaches are commonly used to reduce the noise level of a dataset: re-weighting samples and label correction. Sample re-weighting reduces the noise level by eliminating noisy samples during the model training, whereas label correction directly cleans up the dataset. Recently, label correction methods have shown notable results compared to sample re-weighting methods. [86, 105, 121, 67, 11, 58, 84] claim that re-weighting might show sub-optimal performance by filtering out noisy samples, which might aid in training feature extractors.

Theoretical Inspired Explanation of the Superiority of Label Correction Here, we provide a more theoretically motivated explanation of the above claim. We explore the reason behind the superior performance of label correction compared to sample re-weighting. [14] considers only the upper bound of test accuracy according to the noise level while ignoring the effect on the number of samples on a generalization error while [12] does not consider deep networks. We aim to exhibit the superiority of label correction by presenting the generalization error considering both the number of samples and the noise level. We argue that both the noise level and the number of training samples are critical in determining the generalization error.

For simplicity, our explanation assumes binary classification with asymmetric noise with a level γ. We employ the VC dimension framework [96, 95, 15, 116] to describe the various methods for learning with noisy labels, although the framework provides a loose bound. Further investigation on a tighter bound using the Rademacher complexity or considering the multi-class classification is suggested for future research. Under clean training data distribution $\mathcal{D}$ and clean true data distribution $\mathcal{D}^*$, the VC dimension framework presents the following bound.

$$p\left(|\mathcal{E}_{\mathcal{D}}(f) - \mathcal{E}_{\mathcal{D}^*}(f)| > \epsilon\right) \leq 4\left(2|\mathcal{D}|\right)^{\mathbf{d}_{VC}} \exp\left(-\frac{1}{8}\epsilon^2|\mathcal{D}|\right), \tag{11}$$

where $\mathbf{d}_{VC}$ is the VC dimension and $\mathcal{E}_{\mathcal{D}}(f)$ is the expectation of error for function f regarding the data distribution $\mathcal{D}$. If the VC dimension $\mathbf{d}_{VC}$ is bounded (or finite), convergence is guaranteed because the upper bound decreases exponentially as the size of the dataset increases. Now, we observe a noisy dataset $\bar{\mathcal{D}}$ rather than a clean dataset $\mathcal{D}$. With the triangular inequality and the definition of γ, the following inequalities hold.

$$\begin{aligned}
& p\left(|\mathcal{E}_{\bar{\mathcal{D}}}(f) - \mathcal{E}_{\mathcal{D}^*}(f)| > \epsilon\right) && (12)\\
&= p\left(|\mathcal{E}_{\bar{\mathcal{D}}}(f) - \mathcal{E}_{\mathcal{D}}(f) + \mathcal{E}_{\mathcal{D}}(f) - \mathcal{E}_{\mathcal{D}^*}(f)| > \epsilon\right) && (13)\\
&\leq p\left(|\mathcal{E}_{\bar{\mathcal{D}}}(f) - \mathcal{E}_{\mathcal{D}}(f)| > \epsilon\right) + p\left(|\mathcal{E}_{\mathcal{D}}(f) - \mathcal{E}_{\mathcal{D}^*}(f)| > \epsilon\right) && (14)\\
&\leq \gamma + 4\left(2|\mathcal{D}|\right)^{\mathbf{d}_{VC}} \exp\left(-\frac{1}{8}\epsilon^2|\mathcal{D}|\right) && (15)
\end{aligned}$$

Even if this theoretical bound is loose, we argue that label correction shows better performance than re-weighting samples. As the dataset gets noisier, the number of filtered out samples by the re-weighting methods will also increase, resulting in a drastic reduction of the number of training samples. However, as aforementioned in Section 1, label correction holds an inherent problem of error propagation. We now explain the theoretical background on how the transition matrix acts as a safeguard for the label correction on our FasTEN.

A.2 Background: Statistically Consistent Classification

It is well known that the label transition matrix T can be used to train *statistically consistent classifiers* in the presence of noisy labels [67, 108, 113]. A

statistically consistent classifier is a classifier which guarantees the convergence to an optimal classifier when the number of data samples increases indefinitely. Following [31, 108, 113, 67], we describe the consistency of empirical risk to yield the consistency of the classifier.

Statistically Consistent Empirical Risk Multi-class classification aims to train the hypothesis $\mathcal{H}$, which estimates a label y given an input x. Given the deep neural network $f_{\phi,\theta}$, a hypothesis $\mathcal{H}$ is commonly defined as follows:

$$\mathcal{H}(x) = \arg \max_{n \in \{1,\dots,N\}} f_{\phi,\theta}(x)|_n. \tag{16}$$

With the true sample distribution $\mathcal{D}^*$, the *expected risk* $\mathcal{R}$ for $\mathcal{H}$ is defined as follows:

$$\mathcal{R}(\mathcal{H}) = \min_{\mathcal{H}} \mathbb{E}_{(x,y)\sim\mathcal{D}^*} \left[\mathcal{L}(\mathcal{H}(x), y)\right] \tag{17}$$

Under the distribution $\mathcal{D}^*$ is unknown, the optimal hypothesis $\mathcal{H}$ should minimize $\mathcal{R}$. Since the risk of the optimal hypothesis is difficult to calculate, the empirical risk is usually used for approximation via training dataset $\mathcal{D}$. The definition of empirical risk is as follows:

$$\begin{aligned} \mathcal{R}_{|\mathcal{D}|}(\mathcal{H}) &= \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}(\mathcal{H}(x), y)\right] \\ &= \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathcal{L}(\mathcal{H}(x), y). \end{aligned} \tag{18}$$

Following equation holds for the *statistically consistent empirical risk*:

$$\mathcal{R}(\mathcal{H}) = \lim_{|\mathcal{D}|\to\infty} \mathcal{R}_{|\mathcal{D}|}(\mathcal{H}), \tag{19}$$

where it is common to assume that $\mathcal{D}$ is sampled from $\mathcal{D}^*$ as independent and identically distributed (i.i.d) random variables [108, 14, 15, 31, 17].

Statistically Consistent Classifier Suppose an ideal zero-one loss function $\mathcal{L}^*$ (where it cannot be used in reality because its differentiation is impossible) [5]:

$$\mathcal{L}^*(\mathcal{H}(x) = y) = \mathbf{1}_{\{\mathcal{H}(x)\neq y\}}. \tag{20}$$

$\mathbf{1}_{\{\cdot\}}$ is an indicator function that outputs 1 if $\mathcal{H}(x) \neq y$ and 0 otherwise. If the class of the hypothesis $\mathcal{H}$ is large enough [69], the optimal hypothesis to minimize the expected risk $\mathcal{R}(\mathcal{H})$ corresponds to the Bayes classifier [5] as follows:

$$\mathcal{H}(x) = \arg \max_{n \in \{1,\dots,N\}} p(y = n|x) \tag{21}$$

Many classification loss functions in modern machine learning are proven to be *classification-calibrated* [5, 83], i.e., the classification-calibrated loss function leads to a similar prediction to that of $\mathcal{L}^*$ when $|\mathcal{D}|$ is sufficiently large [69, 95]. For example, the hinge loss is proven to be classification calibrated [110],

and the cross-entropy loss with softmax function is empirically classification-calibrated [29]. The classifier $f_{\phi,\theta}(x)$ is said to be statistically consistent when the classifier converges to the probability $p(y|x)$ by minimizing the empirical risk $\mathcal{R}_{|\mathcal{D}|}(\mathcal{H})$. Note that being risk consistent makes classifier consistent, but not vice versa [108].

Statistically Consistent Classifier in Noisy Labels The empirical risk $\mathcal{R}_{|\bar{\mathcal{D}}|}(\mathcal{H})$ of a noisy dataset $\bar{\mathcal{D}}$ is as follows.

$$\begin{aligned}\mathcal{R}_{|\bar{\mathcal{D}}|}(\mathcal{H}) &= \mathbb{E}_{(x,y)\sim\bar{\mathcal{D}}}\left[\mathcal{L}(\mathcal{H}(x),y)\right] \\ &= \frac{1}{|\bar{\mathcal{D}}|}\sum_{(x,y)\in\bar{\mathcal{D}}}\mathcal{L}(\mathcal{H}(x),y)\end{aligned} \tag{22}$$

Since the statistically consistent classifier $f_{\phi,\theta}(x)$ converges to $p(y|x)$, we can accept $f_{\phi,\theta}(x)$ to approximate $p(y|x)$. Given the definition of the transition matrix $p(\bar{y}|x) = T^\top p(y|x)$, a hypothesis with a noisy dataset $\bar{\mathcal{H}}$ is defined as follows:

$$\bar{\mathcal{H}}(x) = \arg\max_{n\in\{1,\dots,N\}} T^\top f_{\phi,\theta}(x)|_n \tag{23}$$

Hence, minimizing the following empirical risk $\mathcal{R}_{|\bar{\mathcal{D}}|}(\bar{\mathcal{H}})$ using only the noisy dataset $\bar{\mathcal{D}}$ leads to a consistent classifier $f_{\phi,\theta}(x)$ [108].

$$\begin{aligned}\mathcal{R}_{|\bar{\mathcal{D}}|}(\bar{\mathcal{H}}) &= \mathbb{E}_{(x,y)\sim\bar{\mathcal{D}}}\left[\mathcal{L}(\bar{\mathcal{H}}(x),y)\right] \\ &= \frac{1}{|\bar{\mathcal{D}}|}\sum_{(x,y)\in\bar{\mathcal{D}}}\mathcal{L}(\bar{\mathcal{H}}(x),y)\end{aligned} \tag{24}$$

In other words, $f_{\phi,\theta}$ converges to the optimal classifier for the clean data when the sample size of the noisy dataset becomes infinitely large. Although other lines of research guarantee that maximizing accuracy in noisy data distribution maximizes accuracy in clean data distribution even without the transition matrix [15], loss correction via the transition matrix is still an effective consistent classifier training scheme. For this reason, a line of work in learning with noisy labels via the transition matrix attempts to train a statistically consistent classifier by an additional layer modeling the transition matrix preceded by the softmax layer [25, 76, 92, 115, 68, 79, 88]. Incidentally, it is known that modifying the loss function using the transition matrix has a degree of handling instance-dependent label corruption [66, 40].

Statistically Consistent Classifier in Noisy Labels with Small Clean Dataset We exploit a small number of clean data as in [20, 97, 50, 44, 80, 54, 40, 85, 3, 118, 121, 105, 101] while disjointing the clean $\mathcal{D}$ and noisy dataset $\bar{\mathcal{D}}$. It is trivial that a statistically consistent classifier in exploiting a clean set can be obtained

by minimizing the following empirical risk:

$$\begin{aligned}\mathcal{R}_{|\mathcal{D}|}(\mathcal{H}) + \mathcal{R}_{|\bar{\mathcal{D}}|}(\bar{\mathcal{H}}) \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{L}(\mathcal{H}(x),y)\right] + \mathbb{E}_{(x,y)\sim\bar{\mathcal{D}}}\left[\mathcal{L}(\bar{\mathcal{H}}(x),y)\right] \\ &= \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}}\mathcal{L}(\mathcal{H}(x),y) + \frac{1}{|\bar{\mathcal{D}}|}\sum_{(x,y)\in\bar{\mathcal{D}}}\mathcal{L}(\bar{\mathcal{H}}(x),y).\end{aligned} \tag{25}$$

Since the cross-entropy loss surrogates the ideal zero-one loss function $\mathcal{L}^*$ [29], minimizing the empirical risk $\mathcal{R}_{|\mathcal{D}|+|\bar{\mathcal{D}}|}(\mathcal{H},\bar{\mathcal{H}})$ is equivalent to following optimization problem.

$$\arg\min_{\phi,\theta}\sum_{(x,y)\in\mathcal{D}}\mathcal{L}\left(f_{\phi,\theta}(x),y\right) + \sum_{(x,\bar{y})\in\bar{\mathcal{D}}}\mathcal{L}\left(\widehat{\boldsymbol{T}}^\top f_{\phi,\theta}(x),\bar{y}\right). \tag{26}$$

Without loss of generalization, the optimization problem can be rewritten by introducing an episodic batch formation in Section 3.1:

$$\arg\min_{\phi,\theta}\sum_{(x,y)\in d}\mathcal{L}\left(f_{\phi,\theta}(x),y\right) + \sum_{(x,\bar{y})\in\bar{d}}\mathcal{L}\left(\widehat{\boldsymbol{T}}^\top f_{\phi,\theta}(x),\bar{y}\right). \tag{27}$$

A.3 Calculation of the Estimated Transition Matrix $\widehat{T}$ of FasTEN

GLC [40] presents a method to estimate the transition matrix through a small clean dataset similar to our FasTEN. GLC adopts the slow calculation method via a `FOR` or `WHILE` loop since [40] only requires to obtain the transition matrix once in the entire training process. However, our FasTEN needs to estimate the transition matrix for every iteration as we correct the labels on the fly, ending up altering the ideal transition matrix. We speed up the estimation with a single forward propagation by using only matrix operations, avoiding the sluggish `FOR` or `WHILE` loop. Here, we show the derivation of Eq. 2. Let $d_i = \{(x,y)\in d | y_i = 1\}$. Then,

$$\widehat{T}_{ij} = p(\bar{y}=j|y=i) \tag{28}$$

$$= \frac{1}{|d_i|}\sum_{(x,y)\in d_i} p(\bar{y}=j|y=i,x) \tag{29}$$

$$= \frac{1}{|d_i|}\sum_{(x,y)\in d_i} f_{\phi,\theta}(x)\Big|_j \tag{30}$$

$$= \frac{1}{|d_i|}\left(\sum_{(x,y)\in d_i} y f_{\bar{\phi},\bar{\theta}}(x)^\top\Big|_{(i,j)}\right) \tag{31}$$

$$= \left(\sum_{(x,y)\in d_i} y f_{\bar{\phi},\bar{\theta}}(x)^\top\Big|_{(i,j)}\right)\frac{1}{|d_i|} \tag{32}$$

$$= \left(\sum_{(x,y)\in d_i} y f_{\bar{\phi},\bar{\theta}}(x)^\top \Big|_{(i,j)} \right) \left(\operatorname{diag}^{-1} \left(\sum_{(x,y)\in d} y \right) \Big|_{(i,j)} \right) \tag{33}$$

Without loss of generality, $\widehat{\boldsymbol{T}}$ can be written as follows:

$$\widehat{\boldsymbol{T}} = \left(\sum_{(x,y)\in d} y f_{\bar{\phi},\bar{\theta}}(x)^\top \right) \operatorname{diag}^{-1} \left(\sum_{(x,y)\in d} y \right). \tag{34}$$

A.4 Proof of Theorem 1

In this section, we prove under strong assumptions (Theorem 2) followed by milder assumptions (Theorem 1). Theorem 2 estimates the upper bound of the error on transition matrix T, assuming the ideal situation where $p(\bar{y}|x)$ is perfectly parameterized to $f_{\bar{\phi},\bar{\theta}}(x)$.

Theorem 2. *Assuming* $p(\bar{y}|x) = f_{\bar{\phi},\bar{\theta}}(x)$, *for* $\epsilon \geq 0$,

$$p\left(\left|\widehat{T}_{ij} - T_{ij}\right| > \epsilon\right) \leq 2\exp\left(-2\epsilon^2 K\right). \tag{35}$$

Proof. If $p(\bar{y}|x) = f_{\bar{\phi},\bar{\theta}}(x)$, then $p\left(\left|\mathbb{E}\left[\widehat{\boldsymbol{T}}\right] - \boldsymbol{T}\right| > \epsilon\right) = 0$. With the triangular and Hoeffding inequality, the following holds:

$$\begin{aligned}
&p\left(\left|\widehat{T}_{ij} - T_{ij}\right| > \epsilon\right) && (36)\\
&= p\left(\left|\widehat{T}_{ij} - \mathbb{E}\left[\widehat{T}_{ij}\right] + \mathbb{E}\left[\widehat{T}_{ij}\right] - T_{ij}\right| > \epsilon\right) && (37)\\
&\leq p\left(\left|\mathbb{E}\left[\widehat{T}_{ij}\right] - T_{ij}\right| > \epsilon\right) + p\left(\left|\widehat{T}_{ij} - \mathbb{E}\left[\widehat{T}_{ij}\right]\right| > \epsilon\right) && (38)\\
&= p\left(\left|\mathbb{E}\left[\widehat{T}_{ij}\right] - T_{ij}\right| > \epsilon\right) + 2\exp\left(-2\epsilon^2 K\right) && (39)\\
&= 2\exp\left(-2\epsilon^2 K\right). && (40)
\end{aligned}$$

□

With Theorem 2 alone, we can see that the estimation error of transition matrix $\boldsymbol{T}$ decreases exponentially as K (the number of samples per class) increases, as we mentioned in Theorem 1 of Section 3.2.

We assume the hypothetical case that $p(\bar{y}|x)$ could flawlessly model $f_{\bar{\phi},\bar{\theta}}(x)$, but the assumption does not hold in practice. Several lemmas are established in order to prove Theorem 1 under more relaxed assumptions. If $p(\bar{y}|x) \neq f_{\bar{\phi},\bar{\theta}}(x)$, then $p(|\mathbb{E}[\widehat{\boldsymbol{T}}] - \boldsymbol{T}| > \epsilon) \neq 0$. We focus on examining the upper bound of $p(|\mathbb{E}[\widehat{\boldsymbol{T}}] - \boldsymbol{T}| > \epsilon)$ under the relaxed assumption. The upper bound of $\widehat{\boldsymbol{T}}$ (which

is equivalent to $f_{\bar{\phi},\bar{\theta}}(x)$) is strictly 1 since it is a probability. By applying McDiarmid's concentration inequality [9], the following inequality is established:

$$\begin{aligned} & p\left(\left|\mathbb{E}\left[\widehat{T}_{ij}\right]-T_{ij}\right|>\epsilon\right) \\ & \leq \mathbb{E}_{\sigma}\left[\sup_{\mathcal{H}} \frac{1}{|\bar{\mathcal{D}}|} \sum_{(x,\bar{y})\in\bar{\mathcal{D}}} \sigma_x \mathcal{L}(\mathcal{H}(x),\bar{y})\right]+\sqrt{\frac{\log(1/\epsilon)}{2|\bar{\mathcal{D}}|}} \end{aligned} \tag{41}$$

where σ is an i.i.d Rademacher random variable [70] and $\mathcal{H}$ is a hypothesis.

We estimate the upper bound of the estimation error of $\boldsymbol{T}$ by assuming $\mathcal{H}$ is constructed using deep neural networks. A deep neural networks hypothesis $\mathcal{H}'$ is defined as follows.

$$\mathcal{H}'(x)=\bar{\theta}\mathcal{A}_{H-1}(\bar{\phi}_{H-1}\mathcal{A}_{H-2}(\ldots\,\mathcal{A}_1(\bar{\phi}_1 x)))\in\mathbb{R}^N \tag{42}$$

where H is the depth of deep neural networks and $\mathcal{A}_i$ is the i-th activation function. When the function class is limited with deep neural networks, the following lemma holds by borrowing the results of [108].

Lemma 1. *Suppose σ is an i.i.d Rademacher random variable and $\mathcal{L}$ is the cross-entropy loss function which is L-Lipschitz continuous with respect to $\mathcal{H}'$,*

$$\mathbb{E}_{\sigma}\left[\sup_{\mathcal{H}} \frac{1}{|\bar{\mathcal{D}}|} \sum_{(x,\bar{y})\in\bar{\mathcal{D}}} \sigma_x \mathcal{L}(\mathcal{H}(x),\bar{y})\right] \leq NL\mathbb{E}_{\sigma}\left[\sup_{\mathcal{H}'} \frac{1}{|\bar{\mathcal{D}}|} \sum_{(x,\bar{y})\in\bar{\mathcal{D}}} \sigma_x \mathcal{H}'(x)\right] \tag{43}$$

where $\mathcal{H}'$ is a hypothesis belonging to the function class of deep neural networks.

As opposed to using the VC dimension framework in Section 1, this section uses the Rademacher complexity framework [6] to assess the upper bounds of our method. Hypothesis complexity of deep neural networks via Rademacher complexity is broadly studied in [108, 4, 26, 72]. In particular, [26] proves the following lemma:

Lemma 2. *Assume the Frobenius norm of the weight matrices $\bar{\phi}_1,\ldots,\bar{\phi}_{H-1},\bar{\theta}$ are at most $\bar{\Phi}_1,\ldots,\bar{\Phi}_{H-1},\bar{\Theta}$ for H-layer neural networks $f_{\bar{\phi},\bar{\theta}}$. Let the activation functions be 1-Lipschitz, positive-homogeneous, and applied element-wise (such as the ReLU). Let σ is an i.i.d Rademacher random variable. Let x is upper bounded by B, i.e., for any $x\in\mathcal{X}$, $\|x\|\leq B$. Then, for $\epsilon\geq 0$*

$$\mathbb{E}_{\sigma}\left[\sup_{\mathcal{H}'} \frac{1}{|\bar{\mathcal{D}}|} \sum_{(x,\bar{y})\in\bar{\mathcal{D}}} \sigma_x \mathcal{H}'(x)\right] \leq \frac{B(\sqrt{2H\log 2}+1)\bar{\Theta}\Pi_{h=1}^{H-1}\bar{\Phi}_i}{\sqrt{|\bar{\mathcal{D}}|}}. \tag{44}$$

Now, we can complete the proof of Theorem 1.

Proof. With the triangular inequality, Hoeffding inequality, Theorem 2, Lemma 1, and 2, the following holds.

$$
\begin{aligned}
& p\left(\left|\widehat{T}_{ij}-T_{ij}\right|>\epsilon\right) && (45)\\
&= p\left(\left|\widehat{T}_{ij}-\mathbb{E}\left[\widehat{T}_{ij}\right]+\mathbb{E}\left[\widehat{T}_{ij}\right]-T_{ij}\right|>\epsilon\right) && (46)\\
&\leq p\left(\left|\mathbb{E}\left[\widehat{T}_{ij}\right]-T_{ij}\right|>\epsilon\right)+p\left(\left|\widehat{T}_{ij}-\mathbb{E}\left[\widehat{T}_{ij}\right]\right|>\epsilon\right) && (47)\\
&= p\left(\left|\mathbb{E}\left[\widehat{T}_{ij}\right]-T_{ij}\right|>\epsilon\right)+2\exp\left(-2\epsilon^2 K\right) && (48)\\
&\leq \mathbb{E}_\sigma\left[\sup_{\mathcal{H}}\frac{1}{|\bar{\mathcal{D}}|}\sum_{(x,\bar{y})\in\bar{\mathcal{D}}}\sigma_x\mathcal{L}(\mathcal{H}(x),\bar{y})\right]+\sqrt{\frac{\log(1/\epsilon)}{2|\bar{\mathcal{D}}|}}+2\exp\left(-2\epsilon^2 K\right) && (49)\\
&\leq NL\mathbb{E}_\sigma\left[\sup_{\mathcal{H}'}\frac{1}{|\bar{\mathcal{D}}|}\sum_{(x,\bar{y})\in\bar{\mathcal{D}}}\sigma_x\mathcal{H}'(x)\right]+\sqrt{\frac{\log(1/\epsilon)}{2|\bar{\mathcal{D}}|}}+2\exp\left(-2\epsilon^2 K\right) && (50)\\
&\leq \frac{NLB(\sqrt{2H\log 2}+1)\bar{\Theta}\Pi_{h=1}^{H-1}\bar{\Phi}_i}{\sqrt{|\bar{\mathcal{D}}|}}+\sqrt{\frac{\log(1/\epsilon)}{2|\bar{\mathcal{D}}|}}+2\exp\left(-2\epsilon^2 K\right) && (51)
\end{aligned}
$$

□

Theorem 1 and 2 state that the estimation error of transition matrix $\boldsymbol{T}$ is reduced with a larger K. However, we experimentally verify that $K=1$ is enough for achieving comparable performance (See Appendix C.7).

Table 5: Data split composition of dataset used in our experiments.

Dataset	Noisy-train	Clean-train	Valid	Test	Image size	# of classes
CIFAR-10	44K	1K	5K	10K	32 × 32	10
CIFAR-100						100
Clothing1M	1M	47K	14K	10K	224 × 224	14

We end this section by enumerating the limitations of our theoretical analysis. (i) Although our method is based on a multi-head architecture, a clean and a noisy classifier are trained simultaneously, where only the training of the noisy classifier is considered in the theoretical analysis. (ii) We failed to make the upper bound tight for Theorem 1 and 2. Additional assumptions like [12] may yield tighter bound. We conjecture that our method works empirically well for $K=1$ since the upper bound is loose. (iii) The data distribution on a noisy classifier changes in every iteration due to simultaneously corrected labels. However, we assume that the data distribution is stationary in the proof. In order to make our theoretical assumption more adequate for our method, it is necessary to examine the situation under changing data distribution.

B Experimental Details

B.1 Datasets

As shown in Table 5, we use bigger noisy dataset (noisy-train) and smaller clean dataset (clean-train) for training. Validation set is used for obtaining the best model. Since CIFAR datasets do not have the validation set, we split 10% of the entire training set as the validation set. Thus, experimental results may differ from the results of their papers. For Clothing1M, we use its original data split.

B.2 Mini-batch Construction

We sample the mini-batches from both clean and noisy datasets. For the clean dataset, we construct the mini-batch to have the same number of instances per class for clean samplers, whereas the mini-batch of the noisy set is randomly sampled. We choose the batch size 100 for both CIFAR-10 and CIFAR-100, a total of 200 images used per iteration. As the number of classes is 10 and 100, 10 and 1 image(s) are used per class for the clean batch, respectively. We choose the batch size 42 for each noisy and clean set on Clothing1M dataset with 14 classes so that 84 images are used per iteration, where 3 samples are used per class for the clean batch.

B.3 Detailed Training Procedure

CIFAR-10/100 dataset Here, we describe the detailed training procedure of our baselines on CIFAR-10/100 dataset [49]. For all baselines except Deep kNN, we use SGD optimizer with an initial learning rate of 1e-1. For Deep kNN [3], we use Adam optimizer [48] and set the initial learning rate to 1e-3. We follow the experimental settings described in each corresponding papers as much as possible to obtain performance fairly.

L2RW: When training the model, we decay the learning rate to 1e-2 and 1e-3 at 40 and 60 epochs, with a total of 80 epochs. **MW-Net:** For the total of 60 epochs, we decay the learning rate by a factor of 10 at 40 and 50 epochs, respectively. **Deep kNN:** Since it has multiple training stages, we first train two independent models with only the clean dataset $\mathcal{D}$ and sum of the clean and noisy dataset $\mathcal{D} \cup \bar{\mathcal{D}}$, respectively. Then, we filter out the suspicious samples from the noisy set to generate the filtered noisy set $\bar{\mathcal{D}}_{\text{filter}}$ using k-nearest neighbors algorithm (k-NN) of the logit outputs from one of the two trained models, where we choose the model with the better validation set accuracy. Finally, we train the model with the sum of the clean and filtered noisy set $\mathcal{D} \cup \bar{\mathcal{D}}_{\text{filter}}$. For each phase, we train the model until 100 epochs without learning rate decay. **GLC:** We first train the model with only the noisy set $\bar{\mathcal{D}}$ and obtain the label transition matrix with the trained model. With the label transition matrix, we train the initialized model with the clean and noisy set $\mathcal{D} \cup \bar{\mathcal{D}}$ while correcting the loss obtained from the noisy samples. **MLoC:** We first train the model with a warm-up of 75 epochs, i.e., directly training with the noisy label dataset without bells

and whistles. We then meta-train the model with the learning rate of 1e-4 for additional 75 epochs. **MLaC:** For a total of 120 epochs, we decay the learning rate at 80 and 100 epochs by a factor of 10. **MSLC:** Similar to MLoC, we first train the model with the warm-up of 80 epochs, then meta-train the model with the learning rate of 1e-2 and cut it to 1e-3 at 20 epochs, for 40 epochs. **FasTEN (Ours):** we train the model until 70 epochs and decay the learning rate at 50 and 60 epochs by a factor of 10.

Clothing1M dataset For the Clothing1M dataset [109], we borrow the baseline evaluation results from each corresponding paper except for L2RW, where we train the model ourselves as the original paper does not report the results. For a fair comparison, we use the same backbone network, ImageNet-pretrained ResNet-50. **L2RW:** We train the model for 10 epochs using the SGD optimizer with the initial learning rate of 1e-2. We decay the learning rate after 5 epochs by a factor of 10, where we follow the common training procedure borrowed from [105, 85]. **FasTEN (Ours):** Similarly, we use the SGD optimizer with the same initial learning rate, where we decay the learning rate after 1 epochs for total of 2 epochs.

B.4 Evaluation Details for Section 4.3

Considering the situation where we have to purify the existing noisy labels inside the training set automatically, predicting the correct labels of the train samples is crucial. We compare the accuracy on the noisy train dataset where we compare with the clean label, which is unknown to the model at training time. We show the accuracy on CIFAR-10 with symmetric noise of 80%; hence if the model is perfectly overfitted to the noisy set, it will yield 28% accuracy.

For each method, we use the model with the best validation accuracy, i.e., the best model that each has produced. For the meta-model of MLaC, we use the output of the label correction network (LCN), where the network is fed with the feature vector and the noisy label for every noisy dataset to yield a corrected soft label. The feature vector is extracted from the main model, where it is obtained using the features before the fully connected layer. For the meta-model of MSLC, we use the cached soft label from the last epoch. MSLC calculates the soft label with the linear combination of the previous cached soft label, the predicted label from the main model, and the given noisy label, where the weights are continuously learned during training.

C Additional Experiments

C.1 Experiments on CIFAR-N with Real-world Noise

We conducted further experiments on a recently released dataset, CIFAR-N [103], which contains real-world noise from human annotators. CIFAR-N is constructed by relabeling the existing CIFAR dataset using the Amazon Mechanical

Table 6: Test accuracy (%) comparison on CIFAR-N dataset with real-world label noise.

Methods	CIFAR-10N	CIFAR-100N
L2RW [79]	83.15	57.54
MW-Net [84]	83.82	61.12
Deep kNN [3]	82.87	52.71
GLC [40]	82.71	54.72
MLoC [100]	84.45	61.49
MLaC [119]	83.64	45.95
MSLC [103]	84.35	63.51
FasTEN (Ours)	**87.01**	**63.53**

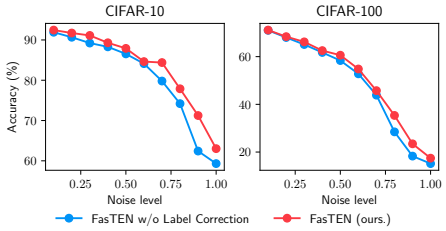

Fig. 5: Effect of label correction on CIFAR-10/100 with various symmetric noise ratio. Accuracy (%) of FasTEN (ours.) and FasTEN without label correction is provided.

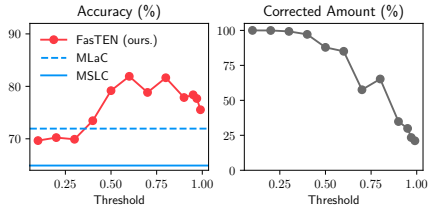

Fig. 6: Robustness to Miscorrection of FasTEN. The corrected label amount (%) and model accuracy (%) according to the label correction threshold (ρ) are provided.

Turk, a crowdsourcing platform, to show the instance-dependent noise from the human annotators. From the original dataset, we extracted 1K clean samples from the original training samples to construct noisy and clean training subsets. We use the same training settings of CIFAR-10/100 experiments in Section 4.1 that does not use semi-supervised methods or sophisticated augmentation techniques to evaluate the methods fairly. Table 6 shows that our proposed FasTEN achieves better performance than the baselines, similar to the results on Clothing1M. These additional results demonstrate that FasTEN is more robust against real-world noise.

C.2 Effect of Label Correction

Figure 5 shows the effect of label correction on our method. When FasTEN does not correct samples at the end of each epoch, it degrades the predictive performance. Furthermore, as the noise gets severe, performance further degrades where we expect the effect of label correction to be larger [14]. From

these observations, we verify that label correction also contributes for improving performance.

C.3 Robustness to Miscorrected Labels

Figure 6 demonstrates the robustness of our method when unreliable samples are also corrected by lowering the threshold of label correction to investigate how safe our method is. We observe the robustness of our method compared to other label correction methods, MLaC [121] and MSLC [105]. We observe how the performance of our model changes when labels are corrected more unreliably as we lower the threshold ρ. The experiments are conducted on CIFAR-10 with the most severe noise level (symmetric 80%). As shown in Figure 6, we verify that our method is robust for miscorrected samples even if all samples in the noisy dataset are corrected when the threshold is under 0.5. Our method uses the transition matrix to avoid the error propagation problem even if unreliable samples are corrected.

C.4 Is the Performance Improvement Due to Over-sampling on the Clean Dataset?

Unlike many MAML-based methods using the clean dataset as gradient guidance in the meta training step, our proposed method utilizes the dataset directly during the model training. One may suspect that the performance improvement of our method may come from over-sampling the clean dataset. Therefore, we compare our proposed model with an over-sampling method [13, 41]. To see the effectiveness of our batch formation, we experiment with the standard cross-entropy loss (Eq. 52) instead of our final objective (Eq. 6), using the same batch formation (Naïve Oversampling). For a fair comparison, label correction is excluded.

$$\arg\min_{\phi,\theta} \sum_{(x,y)\in d} \mathcal{L}\left(f_{\phi,\theta}(x), y\right) + \sum_{(x,\bar{y})\in \bar{d}} \mathcal{L}\left(f_{\phi,\theta}(x), \bar{y}\right) \tag{52}$$

Table 7 shows that our proposed method outperforms the over-sampling method. This observation indicates that our meta-learning method appropriately leverages the clean dataset to estimate the label corruption matrix.

C.5 Comparison to Other Methods with the Transition Matrix

Although the transition matrix is initially introduced as a safeguard to mitigate the risk of label correction in the FasTEN, our FasTEN even shows better performance than other methods employing the transition matrix. This section illustrates that FasTEN, even without label correction, shows better performance than other methods using transition matrix with the clean dataset: GLC (Section C.5) and MLoC (Section C.5).

Table 7: The effect of clean set oversampling on the performance of CIFAR-10/100 experiments. The accuracy (%) of naïve oversampling and FasTEN (ours.) w/o label correction is provided.

Dataset	Method	Symmetric				Asymmetric	
		20 %	40 %	60 %	80 %	20 %	40 %
CIFAR-10	Naïve Oversampling	89.39	85.90	83.90	56.83	90.58	84.41
	FasTEN (ours.) w/o Label Correction	**90.67**	**88.29**	**84.12**	**74.19**	**92.50**	**91.05**
CIFAR-100	Naïve Oversampling	65.42	57.08	42.18	25.88	67.71	61.73
	FasTEN (ours.) w/o Label Correction	**68.02**	**61.75**	**52.79**	**28.46**	**69.59**	**66.07**

Table 8: The effect of two-head architecture via oracle label transition matrix on CIFAR-10/100 dataset. Test accuracy (%) of GLC [40] and FasTEN (ours.) with and without oracle label transition matrix is provided. For a fair comparison, label corruption is excluded in FasTEN.

Dataset	Method	Symmetric				Asymmetric	
		20 %	40 %	60 %	80 %	20 %	40 %
CIFAR-10	GLC [40] w/ Oracle	89.06	85.45	81.56	67.54	91.74	90.35
	FasTEN (ours.) w/ Oracle w/o Label Correction	**91.37**	**88.71**	**83.97**	**74.91**	**91.80**	**91.10**
	GLC [40]	89.66	85.30	80.34	67.44	91.56	89.76
	FasTEN (ours.) w/o Label Correction	**90.67**	**88.29**	**84.12**	**74.19**	**92.50**	**91.05**
CIFAR-100	GLC [40]	60.99	49.00	33.38	20.38	64.43	54.20
	FasTEN (ours.) w/o Label Correction	**68.02**	**61.75**	**52.79**	**28.46**	**69.59**	**66.07**

Comparison to Gold Loss Correction (GLC) [40] Our proposed method is similar to GLC in estimating the label transition matrix, but it shows better performance than GLC even without label correction (See Table 8). Additionally, instead of estimating the transition matrix, we directly use the oracle matrix to examine the effectiveness of the multi-head architecture more clearly. Even using the same oracle matrix for both methods, our FasTEN outperforms GLC. We conjecture that our multi-head architecture trains the model to extract features better than the two-stage training of GLC, which learns noisy classifier and clean classifier consecutively.

Comparison to Meta Loss Correction (MLoC) [101] Since our method does not directly parameterize the label transition matrix $\boldsymbol{T}$, stable estimation of $\boldsymbol{T}$ and its theoretical analysis are possible (See Theorem 1). Table 9 shows that MLoC and our FasTEN without Label Correction (LC) shows comparable performance. MLoC uses several engineering techniques for stable training: a strong prior and gradient clipping, where it is not mentioned in the paper. However, our method shows good performance even without label correction, being robust to different hyperparameters, reducing the need for excessive engineering. We also emphasize that there is a significant gap in performance at a severe noise level.

Table 9: Test accuracy (%) comparison between FasTEN without Label Correction and MLoC [101] on CIFAR-10/100 dataset.

Dataset	Method	Symmetric				Asymmetric	
		20 %	40 %	60 %	80 %	20 %	40 %
CIFAR-10	MLoC [101]	90.50	87.20	81.95	54.64	91.15	89.35
	FasTEN (ours.) w/o Label Correction	**91.37**	**88.71**	**83.97**	**74.91**	**91.80**	**91.10**
CIFAR-100	MLoC [101]	**68.16**	**62.09**	**54.49**	20.23	69.20	**66.48**
	FasTEN (ours.) w/o Label Correction	68.02	61.75	52.79	**28.46**	**69.59**	66.07

Table 10: Incorrect label detection performance comparison on CIFAR-10 with symmetric 80% noise. The Area Under the Receiver Operating Characteristic (AUROC) and The Area Under the Precision-Recall Curve (AUPRC) are provided. Note that pure random model will yield 0.5 AUROC and 0.72 AUPRC. † denotes the performance of the sample weights obtained with the meta model.

	L2RW	L2RW†	MW-NET	Deep kNN	Deep kNN†	GLC	MLoC	MLaC	MLaC†	MSLC	**FasTEN**
AUROC	0.8653	0.4898	0.9205	0.9019	0.8070	0.9324	0.9318	0.9640	0.9564	0.9303	**0.9651**
AUPRC	0.9412	0.7994	0.9631	0.9396	0.9326	0.9674	0.9695	**0.9835**	0.9791	0.9624	**0.9835**

C.6 Incorrect Label Detection Performance Comparison

We consider the case where we have to continuously purify the already-collected dataset with the existence of a human oracle, where the process can be accelerated by correctly detecting the candidates for the wrongly labeled samples. Hence, we regard the incorrect label detection problem as a binary classification problem where the model output probability of the noisy samples is used as the barometer for the correctness of the label.

Settings. We extract the probability values of the noisy labels per sample inside the noisy training set to be the negative score for the binary classification problem where we label 1 for the wrongly labeled sample and 0 otherwise. We additionally measure the performance of the meta-models. We use the meta-learned sample weights for MSLC, MW-Net, and L2RW. For MLaC, we use the probability of the soft label obtained by the meta-model, which is described in detail in Appendix B.4. Finally, as Deep kNN filters out the doubtful samples while training the final model, we regard the process as weighting each sample by 0 or 1 depending on its doubtfulness. Note that we evaluate each method using all the training samples, including the correctly labeled, as each method may mistake those samples to be wrongly labeled.

Results. [80, 85, 3] claim that using meta-learning or pre-training is able to tell whether a sample is mislabeled. Although our model is not directly aimed at finding noisy samples in the noisy set, Table 10 shows that our model achieves comparable or better performance in detecting noisy labels than the baselines.

Table 11: The effect of the number of clean set on CIFAR-10 with symmetric 80% noise. Comparison with other label correction methods with meta-learning is provided.

# of clean examples	MLaC	MSLC	**FasTEN (ours.)**
100	32.92	69.00	**76.48**
250	42.15	63.52	**79.36**
500	50.70	63.35	**77.82**
1000	71.94	64.90	**77.88**

Table 12: The effect of the number of samples per class (K) in the mini-batch on the predictive performance (Accuracy (%)) of CIFAR-10 experiments.

K	Symmetric				Asymmetric	
	20 %	40 %	60 %	80 %	20 %	40 %
2	91.85	89.96	87.00	81.68	92.43	91.11
4	91.79	89.85	86.92	82.18	92.14	91.18
6	92.16	90.07	86.77	79.57	92.14	90.98
8	92.10	89.82	84.81	79.55	92.41	90.74
10	91.72	89.30	84.63	77.88	91.95	90.25

Although [80, 3] claim that the performance has improved because the meta-model detects noisy samples through re-weighting, the actual performance of meta-models is generally lower than that of the final classifier. It implies that [80, 3] may operate with different dynamics than the original author intended.

C.7 Analysis on Our Method

How Many Clean Samples are Required? To verify the effect of the clean dataset size, we observe the performance differences while varying the size. As shown in Table 11, our method consistently shows better performance on different sizes. Especially, even when our method only uses 100 clean samples, it outperforms all the baselines which utilize all the clean samples (1,000 samples). This observation demonstrates that our method could accurately estimate the label transition matrix with a small number of clean samples, which can be applicable to real-world scenarios where it is difficult to obtain a sufficient number of clean samples.

How Sensitive is to the Batch Size? In Section 3.2, we show that the accuracy of estimating the label transition matrix is upper-bounded by the number of samples in the mini-batch. As previous studies [40] mentioned, the quality of the estimated transition matrix affects the performance in learning with noisy

Table 13: Evaluation results varying the hyperparameter λ. Test accuracy (%) with 95% confidence interval of 5-runs is provided.

	λ	Symmetric Noise Level 20%	40%	60%	80%	Asymmetric Noise Level 20%	40%
CIFAR-10	0.01	90.87 ± 0.35	89.03 ± 0.24	85.20 ± 1.00	75.95 ± 1.13	91.06 ± 0.36	89.40 ± 0.44
	0.05	91.69 ± 0.17	89.20 ± 0.64	84.48 ± 0.89	76.39 ± 1.79	91.92 ± 0.31	89.58 ± 0.31
	0.1	91.72 ± 0.20	89.30 ± 0.32	84.63 ± 0.70	77.88 ± 1.09	91.95 ± 0.22	90.25 ± 0.39
	0.2	91.72 ± 0.11	89.61 ± 0.29	85.71 ± 0.24	77.55 ± 2.78	92.20 ± 0.19	90.51 ± 0.27
	0.5	91.94 ± 0.28	90.07 ± 0.17	86.78 ± 0.31	79.52 ± 0.78	92.29 ± 0.10	90.43 ± 0.31
	1.0	91.80 ± 0.20	89.70 ± 0.19	86.66 ± 0.48	80.95 ± 0.44	92.04 ± 0.40	90.54 ± 0.23
CIFAR-100	0.01	65.36 ± 0.77	57.79 ± 1.12	43.65 ± 1.06	26.95 ± 0.76	66.58 ± 0.71	62.19 ± 0.66
	0.05	68.49 ± 0.27	62.47 ± 0.32	53.55 ± 0.86	35.53 ± 1.28	69.73 ± 0.18	65.63 ± 0.79
	0.1	68.38 ± 0.29	62.53 ± 0.33	54.82 ± 0.46	35.35 ± 1.13	69.35 ± 0.13	66.34 ± 0.27
	0.2	68.65 ± 0.09	63.07 ± 0.22	54.84 ± 0.30	35.65 ± 0.66	70.37 ± 0.15	66.93 ± 0.20
	0.5	68.75 ± 0.60	63.82 ± 0.33	55.22 ± 0.64	37.36 ± 1.15	70.35 ± 0.51	67.93 ± 0.53
	1.0	67.91 ± 0.59	62.78 ± 0.28	52.76 ± 1.15	31.45 ± 0.75	70.02 ± 0.60	67.11 ± 0.55

labels. To verify the effect of the number of samples in the mini-batch, we observe the performance changes by varying the number of samples per class in the mini-batch from 1 to 10. As shown in Table 12, there is little change in performance depending on the number of samples per class, although the performance degradation is predicted by Theorem 1 when the number of samples is small. From this observation, we believe that our proposed method shows practicality even in situations where the batch size cannot be increased due to the limited computing resources.

Searching the Optimal Hyperparameter λ We observe performance variance on the CIFAR-10/-100 datasets when we change the hyperparameter λ which is a loss balancing factor. The results are summarized in Table 13. The hyperparameter is searched in {0.01, 0.05, 0.1, 0.2, 0.5, 1.0}.

D Additional Related Work

D.1 Comparison with Other Methods with Label Transition Matrix

Under the assumption that label corruption occurs class-dependently and instance-independently, learning with noisy label methods exploiting the label transition matrix has shown admirable performance [68, 79, 88, 7, 76, 25]. It is well known that training a statistically consistent classifier is possible if the transition matrix is estimated accurately, but precise estimation is usually challenging [67, 108, 113]. Various methods have been proposed to alleviate the issue: imposing strong prior [76, 32], designing a loss function using the ratio of the label transition matrix $\boldsymbol{T}$ [108], or factorization of the transition matrix [113].

However, it is still challenging to estimate the transition matrix with only the noisy dataset. Recently, approaches that improve the estimation accuracy of the transition matrix using a small clean dataset have shown remarkable results: Gold Loss Correction (GLC) [40] and Meta Loss Correction (MLoC) [101]. The clean dataset makes it possible to directly estimate the noisy label posterior, resulting in stable prediction of the transition matrix. Our FasTEN differs from the existing methods which try to find the fixed label transition matrix, as the oracle transition matrix continuously changes during label correction in our method. Our method shows novelty compared to the previous two methods even without the label correction.

Differences from Gold Loss Correction (GLC) [40] Similar to FasTEN, GLC models $p(\bar{y}|x)$ as $f_{\bar{\phi},\bar{\theta}}(x)$ for estimating the transition matrix $\boldsymbol{T}$. However, GLC is more inefficient than our FasTEN because it requires multiple training phases (See § 4.2). We introduce a multi-head architecture with to speed up the training. Furthermore, there is an additional performance advantage compared to GLC. The multi-head architecture is presumed to help obtain a better feature extractor by inducing corruption-independent feature extraction. Detailed experimental results can be found in Appendix C.5.

Differences from Meta Loss Correction (MLoC) [101] MLoC gradually finds the oracle transition matrix $\boldsymbol{T}$ via the MAML framework [20]. As mentioned earlier, MLoC is very slow because it requires three back-propagations for a single iteration due to its nature of MAML (See § 4.2). MLoC directly parameterizes the transition matrix $\boldsymbol{T}$ and learns it using various engineering techniques: strong prior and gradient clipping, which were not mentioned in original paper. In contrast, our method estimates $\boldsymbol{T}$ more accurately by sampling the posterior through a single forward propagation. We empirically validate that our method performs better or comparable to MLoC even without label correction (See Appendix C.5).

D.2 Methods using Multi-head Architecture for Noisy Labels

We propose a multi-head architecture to estimate the transition matrix efficiently: one is for the clean label distribution, and the other is for the noisy label distribution. A similar multi-head architecture has been used in situations dealing with crowdsourcing. Many crowdsourcing studies assume that multiple people label a single image [81, 28, 91], where training a reliable classifier is the goal of the crowdsourcing problem. They maintain separate heads for each annotator, and each head performs multi-task learning to learn each annotator's decisions directly. Then, the final decision is made by voting each head's decision. There is no component for estimating the label transition matrix in these methods and no primary head classifier to learn from the estimated label transition matrix.