Supplementary Material of Dirichlet-Based Prediction Calibration for Learning with Noisy Labels

1 Derivations

1.1 Equation 3

Given an example $\boldsymbol{x}_i$, the predicted probability obtained by the softmax function and our calibrated softmax function can be obtained as:

$$\boldsymbol{\rho}_i = [\rho_{i1}, \rho_{i2}, \ldots, \rho_{iC}] = \left[\frac{e^{o_{i1}}}{\sum_{j=1}^{C} e^{o_{ij}}}, \frac{e^{o_{i2}}}{\sum_{j=1}^{C} e^{o_{ij}}}, \ldots, \frac{e^{o_{iC}}}{\sum_{j=1}^{C} e^{o_{ij}}} \right], \tag{1}$$

and

$$\hat{\boldsymbol{\rho}}_i = [\hat{\rho}_{i1}, \hat{\rho}_{i2}, \ldots, \hat{\rho}_{iC}] = \left[\frac{e^{o_{i1}} + \gamma}{\sum_{j=1}^{C} e^{o_{ij}} + \gamma}, \frac{e^{o_{i2}} + \gamma}{\sum_{j=1}^{C} e^{o_{ij}} + \gamma}, \ldots, \frac{e^{o_{iC}} + \gamma}{\sum_{j=1}^{C} e^{o_{ij}} + \gamma} \right]. \tag{2}$$

Suppose the given label is y, the difference between the gradient of the softmax-based cross-entropy loss with respect to model output $\boldsymbol{o}_i$ on the complementary label c and the gradient of the calibrated softmax-based cross-entropy loss can be expressed as:

$$\begin{aligned}
\frac{\partial \mathcal{L}_{ce|\boldsymbol{\rho}_i}}{\partial o_{ic}}|_{\boldsymbol{x}_i} - \frac{\partial \mathcal{L}_{ce|\hat{\boldsymbol{\rho}}_i}}{\partial o_{ic}}|_{\boldsymbol{x}_i} &= \frac{\partial \left(-y \log \rho_{iy}\right)}{\partial o_{ic}} - \frac{\partial \left(-y \log \hat{\rho}_{iy}\right)}{\partial o_{ic}} \\
&= \frac{\partial \left(-y \log \frac{e^{o_{iy}}}{\sum_{j=1}^{C} e^{o_{ij}}}\right)}{\partial o_{ic}} - \frac{\partial \left(-y \log \frac{e^{o_{iy}}+\gamma}{\sum_{j=1}^{C} e^{o_{ij}} + \gamma}\right)}{\partial o_{ic}} \\
&= \frac{e^{o_{ic}}}{\sum_{j=1}^{C} e^{o_{ij}}} - \frac{e^{o_{ic}}}{\sum_{j=1}^{C} \left(e^{o_{ij}} + \gamma\right)} \\
&= \frac{\gamma C e^{o_{ic}}}{\sum_{j=1}^{C} e^{o_j} \sum_{j=1}^{C} \left(e^{o_{ij}} + \gamma\right)} > 0,
\end{aligned} \tag{3}$$

where the result is always greater than 0.

1.2 Equation 5

For example $\boldsymbol{x}_i$, the predicted probability for a given class c can be expressed as:

$$\begin{aligned}
&P(y=c \mid \boldsymbol{x}_i, \boldsymbol{\theta}) = \int p(y=c \mid \boldsymbol{\rho}_i) p\left(\boldsymbol{\rho}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}\right) d\boldsymbol{\rho}_i \\
&= \int \rho_{ic} p\left(\boldsymbol{\rho}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}\right) d\boldsymbol{\rho}_i \\
&= \int \cdots \int \cdots \int \rho_{ic} \cdot p\left(\rho_{i1}, \cdots \rho_{ic}, \cdots \rho_{iC} \mid \boldsymbol{x}_i, \boldsymbol{\theta}\right) d\rho_{i1} \cdots d\rho_{ic} \cdots d\rho_{iC} \\
&= \int \rho_c \left[\int \cdots \int \int \cdots \int p\left(\rho_{i1}, \cdots \rho_{iC} \mid \boldsymbol{x}_i, \boldsymbol{\theta}\right) d\rho_{i1} \cdots dp_{i(c-1)} d\rho_{i(c+1)} \cdots d\rho_{iC}\right] d\rho_{ic} \\
&= \int \rho_{ic} p\left(\rho_{ic} \mid \boldsymbol{x}_i, \boldsymbol{\theta}\right) d\rho_{ic} \\
&= \int \rho_{ic} \left[\frac{1}{\mathcal{B}\left(\alpha_{ic}, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)} \rho_{ic}^{\alpha_{ic}-1} \left(1-\rho_{ic}\right)^{\sum_{j=1, j\neq c}^{C} \alpha_{ij}-1}\right] d\rho_{ic} \\
&= \frac{\mathcal{B}\left(\alpha_{ic}+1, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)}{\mathcal{B}\left(\alpha_{ic}, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)} \int \frac{1}{\mathcal{B}\left(\alpha_{ic}+1, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)} \rho_{ic}^{\alpha_{ic}} \left(1-\rho_{ic}\right)^{\sum_{j=1, j\neq c}^{C} \alpha_{ij}-1} d\rho_c \\
&= \frac{\mathcal{B}\left(\alpha_{ic}+1, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)}{\mathcal{B}\left(\alpha_{ic}, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)} \cdot 1 \\
&= \frac{\Gamma\left(\alpha_{ic}+1\right) \Gamma\left(\sum_{j=1}^{C} \alpha_{ij}\right)}{\Gamma\left(\sum_{j=1}^{C} \alpha_{ij}+1\right) \Gamma\left(\alpha_{ic}\right)} \\
&= \frac{\alpha_{ic}}{\sum_{j=1}^{C} \alpha_{ij}} \\
&= \frac{g\left(o_{ic}\right)+\gamma}{\sum_{j=1}^{C} \left(g\left(o_{ij}\right)+\gamma\right)},
\end{aligned} \tag{4}$$

where $\mathcal{B}(\cdot,\cdot)$ denotes the Beta function, and we have $p\left(\rho_{ic} \mid \boldsymbol{x}_i, \boldsymbol{\theta}\right) \sim \mathcal{B}\left(\rho_{ic} \mid \alpha_{ic}, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)$ and the following equation according to [Ng et al.(2011)Ng, Tian, and Tang]:

$$\begin{aligned}
p\left(\rho_{ic} \mid \boldsymbol{x}_i, \boldsymbol{\theta}\right) &= \frac{1}{\mathcal{B}\left(\alpha_{ic}, \sum_{j=1, j\neq c}^{C} \alpha_{ij}\right)} \rho_{ic}^{\alpha_{ic}-1} \left(1-\rho_{ic}\right)^{\sum_{j=1, j\neq c}^{C} \alpha_{ij}-1} \\
&= \frac{\Gamma\left(\alpha_{ic}+\sum_{j=1}^{C} \alpha_{ij}\right)}{\Gamma\left(\alpha_{ic}\right) \Gamma\left(\sum_{j=1}^{C} \alpha_{ij}\right)} \rho_{ic}^{\alpha_{ic}-1} \left(1-\rho_{ic}\right)^{\sum_{j=1, j\neq c}^{C} \alpha_{ij}-1}.
\end{aligned} \tag{5}$$

1.3 Equation 6

According to Eq. 4, we can easily obtain $\mathcal{L}_{nll}$ as:

$$\begin{aligned}
\mathcal{L}_{nll} &= -\frac{1}{N} \sum_{i=1}^{N} \log\left[p(y=c \mid \boldsymbol{x}_i, \boldsymbol{\theta})\right] = -\frac{1}{N} \sum_{i=1}^{N} \log\left(\frac{\alpha_{ic}}{\sum_{j=1}^{c} \alpha_{ij}}\right) = \frac{1}{N} \sum_{i=1}^{N} \log\left(\frac{\sum_{i=1}^{c} \alpha_{ij}}{\alpha_{ic}}\right) \\
&= \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{ic} \left[\log\left(\sum_{j=1}^{c} \alpha_{ij}\right) - \log \alpha_{ic}\right].
\end{aligned} \tag{6}$$

1.4 Equation 7

The KL-divergence term $\mathcal{L}_{kl}$ can be further derived as:

$$\mathcal{L}_{kl} = \frac{1}{NC}\sum_{i=1}^{N} D_{KL}\left(Dir\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right) \| Dir(\boldsymbol{\rho}_i \mid \mathbf{1})\right) = \frac{1}{NC}\sum_{i=1}^{N}\int p\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right)\log\frac{p\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right)}{p(\boldsymbol{\rho}_i \mid \mathbf{1})} d\boldsymbol{\rho}_i. \tag{7}$$

Given the probability density function $p\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right) \sim Dir\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right)$, we can obtain the following equation:

$$p\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right) = \frac{1}{\mathbb{B}\left(\tilde{\boldsymbol{\alpha}}_i\right)}\prod_{j=1}^{C}\rho_{ij}^{\tilde{\alpha}_{ij}-1} = \frac{\Gamma\left(\sum_{j=1}^{C}\tilde{\alpha}_{ij}\right)}{\prod_{j=1}^{C}\Gamma\left(\tilde{\alpha}_{ij}\right)}, \tag{8}$$

where $\mathbb{B}(\cdot)$ indicates the multivariate Beta function. Therefore, we have:

$$\begin{aligned}
&p\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right)\log\frac{p\left(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i\right)}{p(\boldsymbol{\rho}_i \mid \mathbf{1})} d\boldsymbol{\rho}_i \\
&= \int\left(\frac{1}{\mathbb{B}\left(\tilde{\boldsymbol{\alpha}}_i\right)}\prod_{j=1}^{C}\rho_{ij}^{\tilde{\alpha}_{ij}-1}\right)\log\left(\frac{\mathbb{B}(\mathbf{1})}{\mathbb{B}\left(\tilde{\boldsymbol{\alpha}}_i\right)}\prod_{j=1}^{C}\rho_{ij}^{\tilde{\alpha}_{ij}-1}\right)d\boldsymbol{\rho}_i \\
&= \log\frac{\mathbb{B}(\mathbf{1})}{\mathbb{B}\left(\tilde{\boldsymbol{\alpha}}_i\right)}\int\frac{1}{\mathbb{B}\left(\tilde{\boldsymbol{\alpha}}_i\right)}\prod_{j=1}^{C}\rho_{ij}^{\tilde{\alpha}_{ij}-1}d\boldsymbol{\rho}_i + \int\left(\log\prod_{j=1}^{C}\rho_{ij}^{\tilde{\alpha}_{ij}-1}\right)\left(\frac{1}{\mathbb{B}\left(\tilde{\alpha}_i\right)}\prod_{j=1}^{C}\rho_{ij}^{\tilde{\alpha}_{ij}-1}\right)d\boldsymbol{\rho}_i \\
&= \log\frac{\mathbb{B}(\mathbf{1})}{\mathbb{B}\left(\tilde{\boldsymbol{\alpha}}_i\right)} + \mathbb{E}_{\boldsymbol{\rho}_i \sim Dir(\boldsymbol{\rho}_i \mid \tilde{\boldsymbol{\alpha}}_i)}\left[\log\prod_{j=1}^{C}\rho_{ij}^{\tilde{\alpha}_{ij}-1}\right] \\
&= \log\frac{\mathbb{B}(\mathbf{1})}{\mathbb{B}\left(\tilde{\boldsymbol{\alpha}}_i\right)} + \sum_{c=1}^{C}\left(\tilde{\alpha}_{ic}-1\right)\mathbb{E}_{\boldsymbol{\rho}_c \sim \mathcal{B}\left(\boldsymbol{\rho}_c \mid \tilde{\alpha}_{ic}, \sum_{j=1, j\neq c}^{C}\alpha_{ij}\right)}\left[\log\rho_{ic}\right] \\
&= \log\left[\frac{\Gamma\left(\sum_{j=1}^{C}\tilde{\alpha}_{ij}\right)}{\Gamma(C)\prod_{j=1}^{C}\Gamma\left(\tilde{\alpha}_{ij}\right)}\right] + \sum_{c=1}^{C}\left(\tilde{\alpha}_{ic}-1\right)\left[\psi\left(\tilde{\alpha}_{ic}\right) - \psi\left(\sum_{j=1}^{C}\tilde{\alpha}_{ij}\right)\right].
\end{aligned} \tag{9}$$

Eventually, $\mathcal{L}_{kl}$ can be simplify as:

$$\mathcal{L}_{kl} = \frac{1}{NC}\sum_{i=1}^{N}\log\left[\frac{\Gamma\left(\sum_{j=1}^{C}\tilde{\alpha}_{ij}\right)}{\Gamma(C)\prod_{j=1}^{C}\Gamma\left(\tilde{\alpha}_{ij}\right)}\right] + \sum_{c=1}^{C}\left(\tilde{\alpha}_{ic}-1\right)\left[\psi\left(\tilde{\alpha}_{ic}\right) - \psi\left(\sum_{j=1}^{C}\tilde{\alpha}_{ij}\right)\right]. \tag{10}$$

2 Pseudo-Code

The training procedure of DPC is shown in Algorithm 1.

3 Additional Experimental Settings

For all CIFAR experiments, we use the same set of hyperparameters: the number of augmentations $M = 2$, sharpening temperature $T = 0.5$, the clean probability threshold $\tau = 0.5$, and Beta distribution parameter $\alpha = 4$. Similar to [Li et al.(2020)Li, Socher, and Hoi], the λ_u is chosen from $\{0, 25, 50, 150\}$ using a small validation set and is shown in Table 1 and 2 for detail. The balancing weight of $\mathcal{L}_{con}$ is set to 1 for CIFAR-10 and 3 for CIFAR-100. For the WebVision dataset, we set $M = 2$, $T = 0.5$, $\tau = 0.5$, $\lambda_u = 0$, $\alpha = 0.5$, and the balancing weight of $\mathcal{L}_{con}$ to 1.

In addition, the CIFAR experiments are conducted on 4-card 2080 servers, while the WebVision experiments are conducted on 4-card 3090-Ti servers.

Algorithm 1 The DPC algorithm.

Input: Training dataset $\mathcal{D}$, two networks $f\left(\cdot;\boldsymbol{\theta}^{(1)}\right)$ and $f\left(\cdot;\boldsymbol{\theta}^{(2)}\right)$, calibrated softmax function $\hat{\boldsymbol{\rho}}$, evidential deep learning loss $\mathcal{L}_{edl}$ loss with balancing factor β, unsupervised loss $\mathcal{L}_{uns}$ with weight λ_{uns}, clean probability threshold τ, number of augmentations M, and sharpening temperature T.
Output: Learned model parameters $\boldsymbol{\theta}^{(1)}$ and $\boldsymbol{\theta}^{(2)}$.

$\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)} = \text{WarmUp}(\mathcal{D}, \mathcal{L}_{edl}, \boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)})$ # *warmup with evidential deep learning loss* $\mathcal{L}_{edl}$
for $e = 1 : \text{MaxEpoch}$ **do**
  $\mathcal{M}^{(1)} = f(\mathcal{D}, \boldsymbol{\theta}^{(1)})$ # *calculate per-example margin with* $\boldsymbol{\theta}^{(1)}$
  $\mathcal{M}^{(2)} = f(\mathcal{D}, \boldsymbol{\theta}^{(2)})$ # *calculate per-example margin with* $\boldsymbol{\theta}^{(2)}$
  # Kurtosis ≤ 1.2 *means that the distribution exhibits a multimodal tendency and is suitable for GMM modeling.*
  if Kurtosis ≤ 1.2 **then**
    $\mathcal{P}^{(1)} = \text{GMM}(\mathcal{D}, \boldsymbol{\theta}^{(1)})$ # *model per-example margin to obtain clean probability for* $\boldsymbol{\theta}^{(2)}$
    $\mathcal{P}^{(2)} = \text{GMM}(\mathcal{D}, \boldsymbol{\theta}^{(2)})$ # *model per-example margin to obtain clean probability for* $\boldsymbol{\theta}^{(1)}$
  else
    $\mathcal{P}^{(1)} = \text{Norm}(\mathcal{D}, \boldsymbol{\theta}^{(1)})$ # *Normalize margins to directly obtain clean probability for* $\boldsymbol{\theta}^{(2)}$
    $\mathcal{P}^{(2)} = \text{Norm}(\mathcal{D}, \boldsymbol{\theta}^{(2)})$ # *Normalize margins to directly obtain clean probability for* $\boldsymbol{\theta}^{(1)}$
  end if
  for $k = 1, 2$ **do**
    $\mathcal{X}^{(k)} = \left\{(\boldsymbol{x}_i, \boldsymbol{y}_i, p_i) \mid p_i \geq \tau, \forall (\boldsymbol{x}_i, \boldsymbol{y}_i, p_i) \in \left(\mathcal{D}, \mathcal{P}^{(k)}\right)\right\}$
    $\mathcal{U}^{(k)} = \left\{\boldsymbol{x}_i \mid p_i < \tau, \forall (\boldsymbol{x}_i, p_i) \in \left(\mathcal{D}, \mathcal{M}^{(k)}\right)\right\}$
    for $i = 1 : \text{MaxIter}$ **do**
      From $\mathcal{X}^{(k)}$, draw a mini-batch $\{(\boldsymbol{x}_b, \boldsymbol{y}_b, p_b); b \in (1, \ldots B)\}$
      From $\mathcal{U}^{(k)}$, draw a mini-batch $\{\boldsymbol{x}_b; b \in (1, \ldots B)\}$
      for $b = 1 : B$ **do**
        $\{\hat{\boldsymbol{x}}_{b,1}, \hat{\boldsymbol{x}}_{b,2}, \ldots, \hat{\boldsymbol{x}}_{b,M}\} = \text{Augment}\left(\{\boldsymbol{x}_{b,1}, \boldsymbol{x}_{b,2}, \ldots, \boldsymbol{x}_{b,M}\}\right)$
        $\{\hat{\boldsymbol{u}}_{b,1}, \hat{\boldsymbol{u}}_{b,2}, \ldots, \hat{\boldsymbol{u}}_{b,M}\} = \text{Augment}\left(\{\boldsymbol{u}_{b,1}, \boldsymbol{u}_{b,2}, \ldots, \boldsymbol{u}_{b,M}\}\right)$
        $\bar{\boldsymbol{\rho}}_b^{(k)} = \frac{1}{M}\sum_m \hat{\boldsymbol{\rho}}\left(f\left(\hat{\boldsymbol{x}}_{b,m}; \boldsymbol{\theta}^{(k)}\right)\right)$ # *use the proposed calibrated softmax function*
        $\hat{\boldsymbol{\rho}}_b^{(k)} = \text{Sharpen}(p_b \boldsymbol{y}_b^{(k)} + (1 - p_b)\, \bar{\boldsymbol{\rho}}_b^{(k)}, T)$
        $\bar{\boldsymbol{\rho}}_b = \frac{1}{2M}\sum_m \left[\hat{\boldsymbol{\rho}}\left(f\left(\hat{\boldsymbol{u}}_{b,m}; \boldsymbol{\theta}^{(1)}\right) + \hat{\boldsymbol{\rho}}\left(f\left(\hat{\boldsymbol{u}}_{b,m}; \boldsymbol{\theta}^{(2)}\right)\right)\right)\right]$ # *with calibrated softmax*
        $\hat{\boldsymbol{\rho}}_b = \text{Sharpen}(\bar{\boldsymbol{\rho}}_b, T)$
      end for
      $\hat{\mathcal{X}} = \left\{\left(\hat{\boldsymbol{x}}_{b,m}, \boldsymbol{y}_b, \hat{\boldsymbol{\rho}}_b^{(k)}\right); b \in (1, 2, \ldots, B), m \in (1, 2, \ldots, M)\right\}$
      $\hat{\mathcal{U}} = \{(\hat{\boldsymbol{u}}_{b,m}, \hat{\boldsymbol{\rho}}_b); b \in (1, 2, \ldots, B), m \in (1, 2, \ldots, M)\}$
      $\mathcal{X}', \mathcal{U}' = \text{MixMatch}^{(1)}\left(\hat{\mathcal{X}}, \hat{\mathcal{U}}\right)$ # *MixMatch stage 1: produce mix-up examples*
      # *MixMatch stage 2: calculate the training loss with our proposed* $\mathcal{L}_{edl}$ *and* $\mathcal{L}_{uns}$
      $\mathcal{L}_{total} = \text{MixMatch}^{(2)}\left(\mathcal{X}', \mathcal{U}', \mathcal{L}_{edl}, \beta, \mathcal{L}_{uns}, \lambda_{uns}\right)$
      $\boldsymbol{\theta}^{(k)} = \text{SGD}(\mathcal{D}, \mathcal{L}_{total}, \boldsymbol{\theta}^{(k)})$
    end for
  end for
end for

Table 1: Unsupervised loss weight λ_u for CIFAR-10 and CIFAR-100 with synthetic noise.

Hyperparameter	CIFAR-10						CIFAR-100					
	Symmetric			Asymmetric			Symmetric			Asymmetric		
	20%	50%	80%	10%	30%	40%	20%	50%	80%	10%	30%	40%
λ_u	0	25	25	0	0	0	25	150	150	25	25	150

Table 2: Unsupervised loss weight λ_u for CIFAR-N with real-world noise.

Hyperparameter	CIFAR-10N					CIFAR-100N
	aggre	rand1	rand2	rand3	worst	noisy100
λ_u	0	0	0	0	25	25

Table 3: Additional ablation studies on CIFAR-N with real-world noise.

Hyperparameter	CIFAR-10N					CIFAR-100N
	aggre	rand1	rand2	rand3	worst	noisy100
MLNT	91.37	88.65	88.54	89.13	84.15	58.24
MLNT w $\mathcal{L}_{edl}$	91.45	89.37	88.56	89.50	83.99	60.35
Improve	$\uparrow$ 0.08	$\uparrow$ 0.72	$\uparrow$ 0.02	$\uparrow$ 0.37	$\downarrow$ 0.16	$\uparrow$ 2.11
SOP	95.61	95.28	95.31	95.39	93.24	67.81
SOP w $\mathcal{L}_{edl}$	95.87	95.75	95.35	95.65	93.29	68.25
Improve	$\uparrow$ 0.26	$\uparrow$ 0.47	$\uparrow$ 0.04	$\uparrow$ 0.26	$\uparrow$ 0.05	$\uparrow$ 0.44

4 Additional Experimental Results

To further verify the generalizability of the Dirichlet-based prediction calibration method, we also provide the results of another two methods, MLNT [Li et al.(2019)Li, Wong, Zhao, and Kankanhalli] and SOP [Liu et al.(2022)Liu, Zhu, Qu, and You], integrated with $\mathcal{L}_{edl}$ in Table 3 on CIFAR-N with real-world noise. The proposed method continues to be effective, indicating its general applicability.

References

[Li et al.(2019)Li, Wong, Zhao, and Kankanhalli] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Learning to learn from noisy labeled data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5051–5059, 2019.

[Li et al.(2020)Li, Socher, and Hoi] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020.

[Liu et al.(2022)Liu, Zhu, Qu, and You] Sheng Liu, Zhihui Zhu, Qing Qu, and Chong You. Robust training under label noise by over-parameterization. In *International Conference on Machine Learning*, pages 14153–14172. PMLR, 2022.

[Ng et al.(2011)Ng, Tian, and Tang] Kai Wang Ng, Guo-Liang Tian, and Man-Lai Tang. Dirichlet and related distributions: Theory, methods and applications. 2011.