Supplementary Material for Task Agnostic Robust Learning on Corrupt Outputs by Correlation-Guided Mixture Density Networks

Sungjoon Choi
Kakao Brain
sam.choi@kakaobrain.com

Sanghoon Hong
Kakao Brain
sanghoon.hong@kakaobrain.com

Kyungjae Lee
Seoul National University
kyungjae.lee@rllab.snu.ac.kr

Sungbin Lim
UNIST
sungbin@unist.ac.kr

In this supplementary material, we provide full proofs of theorems. We also elaborate the details of conducted experiments with additional illustrative figures and results.

A. Proof of Theorem in Section 3

In this appendix, we introduce fundamental theorems which lead to Cholesky transform for given random variables (W, Z). We apply this transform to random matrices $\mathbf{W}$ and $\mathbf{Z}$ which carry out weight matrices for prediction and a supplementary role, respectively. We also elaborate the details of conducted experiments with additional illustrative figures and results. Particularly, we show additional classification experiments with the MNST dataset on different noise configurations.

Lemma 1. Let W and Z be uncorrelated random variables such that

$$\begin{cases} \mathbb{E}W = \mu_W, & \mathbb{V}(W) = \sigma_W^2 \\ \mathbb{E}Z = 0, & \mathbb{V}(Z) = \sigma_Z^2 \end{cases} \tag{1}$$

For a given $-1 \leq \rho \leq 1$, set

$$\tilde{Z} = \rho \frac{\sigma_Z}{\sigma_W}(W - \mu_W) + \sqrt{1-\rho^2} Z \tag{2}$$

Then $\mathbb{E}\tilde{Z} = 0$, $\mathbb{V}(\tilde{Z}) = \sigma_Z^2$, and $\text{Corr}(W, \tilde{Z}) = \rho$.

Proof. Since W and Z are uncorrelated, we have

$$\mathbb{E}\left[(W - \mu_W)Z\right] = \mathbb{E}(W - \mu_W)\mathbb{E}Z = 0 \tag{3}$$

By (1), we directly obtain

$$\mathbb{E}\tilde{Z} = \rho \frac{\sigma_Z}{\sigma_W}\left(\mathbb{E}W - \mu_W\right) + \mathbb{E}Z = 0$$

Also, by (1) and (3),

$$\begin{aligned} \mathbb{V}\left(\tilde{Z}\right) = \mathbb{E}|\tilde{Z}|^2 &= \rho^2 \left(\frac{\sigma_Z}{\sigma_W}\right)^2 \mathbb{V}(W) + \mathbb{V}(Z) + 2\rho \frac{\sigma_Z}{\sigma_W} \underbrace{\mathbb{E}\left[(W - \mu_W)Z\right]}_{=0} \\ &= \rho^2 \frac{\sigma_Z^2}{\sigma_W^2}\sigma_W^2 + (1-\rho^2)\sigma_Z^2 = \sigma_Z^2 \end{aligned}$$

Similarly,

$$\begin{aligned}\text{Cov}(W,\tilde{Z}) &= \mathbb{E}\left[(W-\mu_W)\tilde{Z}\right] \\ &= \mathbb{E}\left[(W-\mu_W)\rho\frac{\sigma_Z}{\sigma_W}(W-\mu_W)\right] + \underbrace{\mathbb{E}\left[(W-\mu_W)Z\right]}_{=0} \\ &= \rho\frac{\sigma_Z}{\sigma_W}\mathbb{V}(W) = \rho\sigma_Z\sigma_W\end{aligned}$$

Therefore

$$\text{Corr}(W,\tilde{Z}) = \frac{\text{Cov}(W,\tilde{Z})}{\sqrt{\mathbb{V}(W)}\sqrt{\mathbb{V}(\tilde{Z})}} = \frac{\rho\sigma_W\sigma_Z}{\sigma_W\sigma_Z} = \rho$$

The lemma is proved. □

Lemma 2. Assume the same condition in Lemma 1 and define $\tilde{Z}$ as (2). For given functions $\varphi : \mathbb{R} \to \mathbb{R}$ and $\psi : \mathbb{R} \to (0,\infty)$, set $\tilde{W} := \varphi(\rho) + \psi(\rho)\tilde{Z}$. Then

$$\mathbb{E}\tilde{W} = \varphi(\rho), \quad \mathbb{V}(\tilde{W}) = |\psi(\rho)|^2\sigma_Z^2, \quad \text{Corr}(W,\tilde{W}) = \rho$$

Proof. Note that

$$\mu_{\tilde{W}} = \varphi(\rho) + \psi(\rho)\mu_{\tilde{Z}} = \varphi(\rho)$$
$$\sigma_{\tilde{W}}^2 = |\psi(\rho)|^2\,\mathbb{E}\left(\tilde{Z}-\mu_{\tilde{Z}}\right)^2 = \psi^2(\rho)\sigma_Z^2$$

Therefore, by Lemma 1

$$\begin{aligned}\mathbb{E}\left[(W-\mu_W)(\tilde{W}-\mu_{\tilde{W}})\right] &= \psi(\rho)\mathbb{E}\left[(W-\mu_W)(\tilde{Z}-\mu_{\tilde{Z}})\right] \\ &= \rho\psi(\rho)\sigma_W\sigma_Z\end{aligned}$$

Hence

$$\text{Corr}(W,\tilde{W}) = \frac{\mathbb{E}\left[(W-\mu_W)(\tilde{W}-\mu_{\tilde{W}})\right]}{\sigma_W\sigma_{\tilde{W}}} = \frac{\rho\psi(\rho)\sigma_W\sigma_Z}{\psi(\rho)\sigma_W\sigma_Z} = \rho$$

The lemma is proved. □

Now we prove the aforementioned theorem in Section 3.

Theorem. *Let* $\boldsymbol{\rho} = (\rho_1,\ldots,\rho_K) \in \mathbb{R}^K$. *For* $p \in \{1,2\}$, *random matrices* $\mathbf{W}^{(p)} \in \mathbb{R}^{K\times Q}$ *are given such that for every* $k \in \{1,\ldots,K\}$,

$$\text{Cov}\left(W_{ki}^{(p)}, W_{kj}^{(p)}\right) = \sigma_p^2\delta_{ij}, \quad \text{Cov}\left(W_{ki}^{(1)}, W_{kj}^{(2)}\right) = \rho_k\sigma_1\sigma_2\delta_{ij} \tag{4}$$

Given $\mathbf{h} = (h_1,\ldots,h_Q) \in \mathbb{R}^Q$, *set* $\mathbf{y}^{(p)} = \mathbf{W}^{(p)}\mathbf{h}$ *for each* $p \in \{1,2\}$. *Then an elementwise correlation between* $\mathbf{y}^{(1)}$ *and* $\mathbf{y}^{(2)}$ *equals* $\boldsymbol{\rho}$ *i.e.*

$$\text{Corr}\left(y_k^{(1)}, y_k^{(2)}\right) = \rho_k, \quad \forall k \in \{1,\ldots,K\}$$

Proof. First we prove that for $p \in \{1,2\}$ and $k \in \{1,\ldots,K\}$

$$\mathbb{V}\left(y_k^{(p)}\right) = \sigma_p^2\left\|\mathbf{h}\right\|^2 \tag{5}$$

Note that

$$\begin{aligned}\mathbb{V}\left(y_k^{(p)}\right) &= \mathbb{E}\left[\left(\sum_{i=1}^{Q} W_{ki}^{(p)} h_i - \mathbb{E}\left[\sum_{i=1}^{Q} W_{ki}^{(p)} h_i\right]\right)^2\right] \\ &= \mathbb{E}\left[\left(\sum_{i=1}^{Q}\left(W_{ki}^{(p)} - \mathbb{E}W_{ki}^{(p)}\right) h_i\right)^2\right] \\ &= \mathbb{E}\left[\sum_{i,j}^{Q}\left(W_{ki}^{(p)} - \mathbb{E}W_{ki}^{(p)}\right)\left(W_{kj}^{(p)} - \mathbb{E}W_{kj}^{(p)}\right) h_i h_j\right] \\ &= \sum_{i,j}^{Q} \mathrm{Cov}(W_{ki}^{(p)}, W_{kj}^{(p)}) h_i h_j\end{aligned}$$

By (4),

$$\mathbb{V}\left(y_k^{(p)}\right) = \sum_{i,j}^{Q} \mathrm{Cov}(W_{ki}^{(p)}, W_{kj}^{(p)}) h_i h_j = \sum_{i,j}^{Q} \sigma_p^2 h_i h_j \delta_{ij} = \sum_{i=1}^{Q} \sigma_p^2 h_i^2 = \sigma_p^2 \|\mathbf{h}\|^2$$

so (5) is proved. Next we prove

$$\mathrm{Cov}(y_k^{(1)}, y_k^{(2)}) = \rho_k \sigma_1 \sigma_2 \|\mathbf{h}\|^2 \tag{6}$$

Observe that

$$\begin{aligned}\mathrm{Cov}(y_k^{(1)}, y_k^{(2)}) &= \mathbb{E}\left[\left(y_k^{(1)} - \mathbb{E}y_k^{(1)}\right)\left(y_k^{(2)} - \mathbb{E}y_k^{(2)}\right)\right] \\ &= \mathbb{E}\left[\left(\sum_{i=1}^{Q} W_{ki}^{(1)} h_i - \mathbb{E}\left[\sum_{i=1}^{Q} W_{ki}^{(1)} h_i\right]\right)\left(\sum_{j=1}^{Q} W_{kj}^{(2)} h_j - \mathbb{E}\left[\sum_{j=1}^{Q} W_{kj}^{(2)} h_j\right]\right)\right] \\ &= \mathbb{E}\left[\sum_{i,j}^{Q}\left(W_{ki}^{(1)} - \mathbb{E}W_{ki}^{(1)}\right)\left(W_{kj}^{(2)} - \mathbb{E}W_{kj}^{(2)}\right) h_i h_j\right] \\ &= \sum_{i,j}^{Q} \mathrm{Cov}(W_{ki}^{(1)}, W_{kj}^{(2)}) h_i h_j\end{aligned}$$

Similarly,

$$\mathrm{Cov}(y_k^{(1)}, y_k^{(2)}) = \sum_{i,j}^{Q} \mathrm{Cov}(W_{ki}^{(1)}, W_{kj}^{(2)}) h_i h_j = \sum_{i,j}^{Q} \rho_k \sigma_1 \sigma_2 h_i h_j \delta_{ij} = \rho_k \sigma_1 \sigma_2 \|\mathbf{h}\|^2$$

Hence (6) is proved. Therefore by (5) and (6)

$$\mathrm{Corr}(y_k^{(1)}, y_k^{(2)}) = \frac{\mathrm{Cov}(y_k^{(1)}, y_k^{(2)})}{\sqrt{\mathbb{V}(y_k^{(1)})}\sqrt{\mathbb{V}(y_k^{(2)})}} = \frac{\rho_k \sigma_1 \sigma_2 \|\mathbf{h}\|^2}{\sqrt{\sigma_1^2 \|\mathbf{h}\|^2}\sqrt{\sigma_2^2 \|\mathbf{h}\|^2}} = \rho_k$$

The theorem is proved. □

Remark. Recall the definition of Cholesky transform: for $-1 < \rho < 1$

$$\mathscr{T}_{(\rho,\mu_W,\sigma_W,\sigma_Z)}(w, z) := \rho\mu_W + \sqrt{1-\rho^2}\left(\rho\frac{\sigma_Z}{\sigma_W}(w - \mu_W) + \sqrt{1-\rho^2} z\right) \tag{7}$$

Table 1: The RMSEs of compared methods on synthetic toy examples

Outliers	ChoiceNet	MLP	GPR	LGPR	RGPR	MDN
0%	0.034	0.039	**0.008**	0.022	0.017	0.028
20%	0.022	0.413	0.280	0.206	**0.013**	0.087
40%	**0.018**	0.452	0.447	0.439	1.322	0.565
60%	**0.023**	0.636	0.602	0.579	0.738	0.645
80%	**0.084**	0.829	0.779	0.777	1.523	0.778

Note that we do not assume W and Z should follow typical distributions. Hence every above theorems hold for general class of random variables. Additionally, by Theorem 2 and (7), $\tilde{W}$ has the following ρ-dependent behaviors;

$$\mathbb{E}\tilde{W} \to \begin{cases} \mu_W & : \rho \to 1 \\ 0 & : \rho \to 0 \\ -\mu_W & : \rho \to -1 \end{cases}, \quad \mathbb{V}(\tilde{W}) \to \begin{cases} 0 & : \rho \to \pm 1 \\ \sigma_Z^2 & : \rho \to 0 \end{cases}$$

Thus strongly correlated weights $\tilde{W}$ i.e. $\rho \approx 1$, provide prediction with confidence while uncorrelated weights encompass uncertainty. These different behaviors of weights perform regularization and preclude over-fitting caused by bad data since uncorrelated and negative correlated weights absorb vague and outlier pattern, respectively.

B. More Experiments

B.1. Regression Tasks

We conduct three regression experiments: 1) a synthetic scenario where the training dataset contains outliers sampled from other distributions, 2) using a Boston housing dataset with synthetic outliers, 3) a behavior cloning scenario where the demonstrations are collected from both expert and adversarial policies.

Synthetic Example We first apply ChoiceNet to a simple one-dimensional regression problem of fitting $f(x) = \cos(\frac{\pi}{2}x)\exp(-(\frac{x}{2})^2)$ where $x \in [-3, +3]$ as shown in Figure 1. ChoiceNet is compared with a naive multilayer perceptron (MLP), Gaussian process regression (GPR) [10], leveraged Gaussian process regression (LGPR) with leverage optimization [3], and robust Gaussian process regression (RGPR) with an infinite Gaussian process mixture model [11] are also compared. ChoiceNet has five mixtures and it has two hidden layers with 32 nodes with a ReLU activation function. For the GP based methods, we use a squared-exponential kernel function and the hyper-parameters are determined using a simple median trick [4][1]. To evaluate its performance in corrupt datasets, we randomly replace the original target values with outliers whose output values are uniformly sampled from -1 to $+3$. We vary the outlier rates from 0% (clean) to 80% (extremely noisy).

Table 1 illustrates the RMSEs (root mean square errors) between the reference target function and the fitted results of ChoiceNet and other compared methods. Given an intact training dataset, all the methods show stable performances in that the RMSEs are all below 0.1. Given training datasets whose outlier rates exceed 40%, however, only ChoiceNet successfully fits the target function whereas the other methods fail as shown in Figure 1.

Boston Housing Dataset Here, we used a real world dataset, a Boston housing price dataset, and checked the robustness of the proposed method and compared with standard multi-layer perceptrons with four different types of loss functions: standard $L2$-loss, $L1$-loss which is known to be robust to outliers, a robust loss (RL) function proposed in [1], and a leaky robust loss (LeakyRL) function. We further implement the leaky version of [1] in that the original robust loss function with Tukey's biweight function discards the instances whose residuals exceed certain threshold.

Behavior Cloning Example In this experiment, we apply ChoiceNet to behavior cloning tasks when given demonstrations with mixed qualities where the proposed method is compared with a MLP and a MDN in two locomotion tasks: *HalfCheetah* and *Walker2d*. The network architectures are identical to those in the synthetic regression example tasks. To evaluate the robustness of ChoiceNet, we collect demonstrations from both an expert policy and an adversarial policy where two policies are

[1] A median trick selects the length parameter of a kernel function to be the median of all pairwise distances between training data.

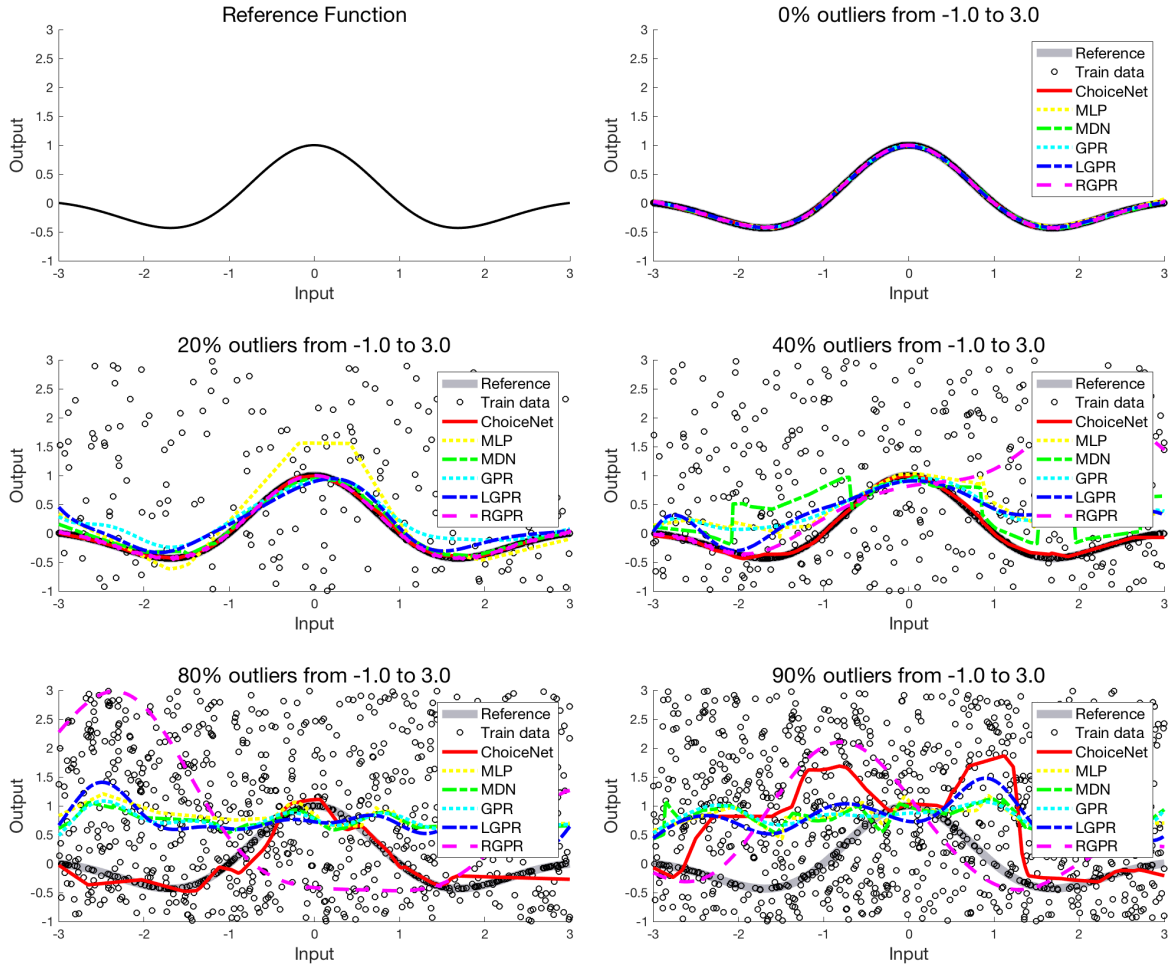


Figure 1: Reference function and fitting results of compared methods on different outlier rates, 0%,20% 40%, 80%, and 90%).

Table 2: The RMSEs of compared methods on the Boston Housing Dataset

Outliers	ChoiceNet	$L2$	$L1$	RL	LeakyRL	MDN
0%	3.29	**3.22**	3.26	4.28	3.36	3.46
10%	**3.99**	5.97	5.72	6.36	5.71	6.5
20%	**4.77**	7.51	7.16	8.08	7.08	8.62
30%	**5.94**	9.04	8.65	10.54	8.67	8.97
40%	**6.80**	9.88	9.69	10.94	9.68	10.44

Table 3: Average returns of compared methods on behavior cloning problems using MuJoCo

Outliers	HalfCheetah			Walker2d		
	ChoiceNet	MDN	MLP	ChoiceNet	MDN	MLP
10%	**2068.14**	192.53	852.91	**2754.08**	102.99	537.42
20%	**1498.72**	675.94	372.90	**1887.73**	95.29	1155.80
30%	**2035.91**	363.08	971.24	-267.10	**-260.80**	-728.39

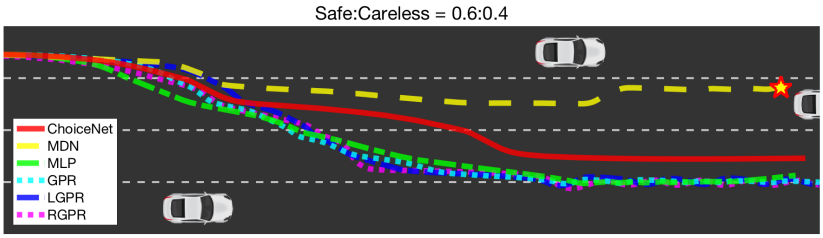


Figure 2: Resulting trajectories of compared methods trained with mixed demonstrations. (best viewed in color).

Table 4: Collision rates of compared methods on straight lanes.

Outliers	ChoiceNet	MDN	MLP	GPR	LGPR	RGPR
0%	**0%**	50.83%	**0%**	0.83%	4.17%	3.33%
10%	**0%**	38.33%	**0%**	2.5%	1.67%	4.17%
20%	**0%**	41.67%	**0%**	7.5%	6.67%	10%
30%	**0%**	66.67%	1.67%	4.17%	1.67%	7.5%
40%	**0.83%**	35%	3.33%	6.67%	6.67%	24.17%

Table 5: Root mean square lane deviation distances (m) of compared methods on straight lanes.

Outliers	ChoiceNet	MDN	MLP	GPR	LGPR	RGPR
0%	0.314	0.723	**0.300**	0.356	0.349	0.424
10%	**0.352**	0.387	0.438	0.401	0.446	0.673
20%	**0.349**	0.410	0.513	0.418	0.419	0.725
30%	**0.368**	**0.368**	0.499	0.455	0.476	0.740
40%	**0.370**	0.574	0.453	0.453	0.453	0.636

trained by solving the corresponding reinforcement learning problems using the state-of-the-art proximal policy optimization (PPO) [13]. For training adversarial policies for both tasks, we flip the signs of the directional rewards so that the agent gets incentivized by going backward. We evaluate the performances of the compared methods using 500 state-action pairs with different mixing ratio and measure the average return over 100 consecutive episodes. The results are shown in Table 3. In both cases, ChoiceNet outperforms compared methods by a significant margin. Additional behavior cloning experiments for autonomous driving can be found in the supplement material.

Autonomous Driving Experiment In this experiment, we apply ChoiceNet to a autonomous driving scenario in a simulated environment. In particular, the tested methods are asked to learn the policy from driving demonstrations collected from both safe and careless driving modes. We use the same set of methods used for the previous task. The policy function is defined as a mapping between four dimensional input features consist of three frontal distances to left, center, and right lanes and lane deviation distance from the center of the lane to the desired heading. Once the desired heading is computed, the angular velocity of a car is computed by $10 * (\theta_{\text{desired}} - \theta_{\text{current}})$ and the directional velocity is fixed to $10m/s$. The driving demonstrations are collected from keyboard inputs by human users. The objective of this experiment is to assess its performance on a training set generated from two different distributions. We would like to note that this task does not have a reference target function in that all demonstrations are collected manually. Hence, we evaluated the performances of the compared methods by running the trained policies on a straight track by randomly deploying static cars.

Table 4 and Table 5 indicate collision rates and RMS lane deviation distances of the tested methods, respectively, where the statistics are computed from 50 independent runs on the straight lane by randomly placing static cars as shown in Figure 2. ChoiceNet clearly outperforms compared methods in terms of both safety (low collision rates) and stability (low RMS lane deviation distances).

Here, we describe the features used for the autonomous driving experiments. As shown in the manuscript, we use a four

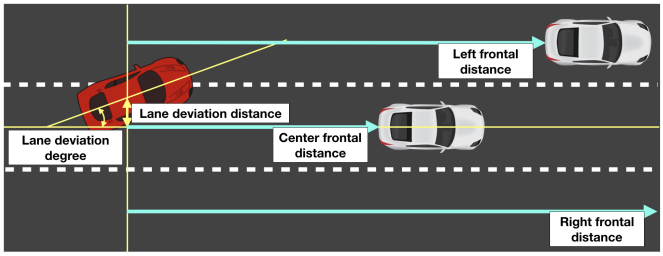
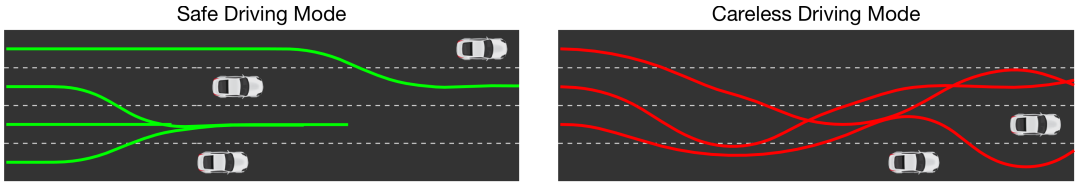


Figure 3: Descriptions of the featrues of an ego red car used in autonomous driving experiments.



(a) (b)

Figure 4: Manually collected trajectories of (a) safe driving mode and (b) careless driving mode. (best viewed in color).

dimensional feature, a lane deviation distance of an ego car, and three frontal distances to the closest car at left, center, and right lanes as shown in Figure 3. We upperbound the frontal distance to $40m$. Figure 4(a) and 4(b) illustrate manually collected trajectories of a safe driving mode and a careless driving mode.

B.2. Classification Tasks

Here, we conduct comprehensive classification experiments using MNIST, CIFAR-10, and Large Movie Review datasets to evaluate the performance of ChoiceNet on corrupt labels. For the image datasets, we followed two different settings to generate noisy datasets: one following the setting in [15] and the other from [5] which covers both symmetric and asymmetric noises. For the Large Movie Review dataset, we simply shuffle the labels to the other in that it only contains two labels.

MNIST For MNIST experiments, we randomly shuffle a percentage of the labels with the corruption probability p from 50% to 95% and compare median accuracies after five runs for each configuration following the setting in [15].

We construct two networks: a network with two residual blocks [6] with $3 \times 3 \times 64$ convolutional layers followed by a fully-connected layer with 256 output units (ConvNet) and ChoiceNet with the same two residual blocks followed by the MCDN block (ConvNet+CN). We train each network for 50 epochs with a fixed learning rate of 1e-5.

We train each network for 300 epochs with a minibatch size of 256. We begin with a learning rate of 0.1, and it decays by $1/10$ after 150 and 225 epochs. We apply random horizontal flip and random crop with $4-$pixel-padding and use a weight decay of 0.0001 for the baseline network as [6].

The classification results are shown in Table 6 where ChoiceNet consistently outperforms ConvNet and ConvNet+Mixup by a significant margin, and the difference between the accuracies of ChoiceNet and the others becomes more clear as the corruption probability increases.

Here, we also present additional experimental results using the MNIST dataset on following three different scenarios:

1. Biased label experiments where we randomly assign the percentage of the training labels to label 0.
2. Random shuffle experiments where we randomly replace the percentage of the training labels from the uniform multinomial distribution.
3. Random permutation experiments where we replace the percentage of the labels based on the label permutation matrix where we follow the random permutation in [12].

The best and final accuracies on the intact test dataset for biased label experiments are shown in Table 7. In all corruption rates, ChoiceNet achieves the best performance compared to two baseline methods. The learning curves of the biased label

Table 6: Test accuracies on the MNIST datasets with corrupt labels.

Corruption p	Configuration	Best	Last
50%	ConvNet	95.4	89.5
	ConvNet+Mixup	97.2	96.8
	ConvNet+CN	**99.2**	**99.2**
	MDN	97.7	97.7
80%	ConvNet	86.3	76.9
	ConvNet+Mixup	87.2	87.2
	ConvNet+CN	**98.2**	**97.6**
	MDN	85.2	78.7
90%	ConvNet	76.1	69.8
	ConvNet+Mixup	74.7	74.7
	ConvNet+CN	**94.7**	**89.0**
	MDN	61.4	50.2
95%	ConvNet	72.5	64.4
	ConvNet+Mixup	69.2	68.2
	ConvNet+CN	**88.5**	**80.0**
	MDN	31.2	25.9

Table 7: Test accuracies on the MNIST dataset with biased label.

Corruption p	Configuration	Best	Last
25%	ConvNet	95.4	89.5
	ConvNet+Mixup	97.2	96.8
	ChoiceNet	**99.2**	**99.2**
40%	ConvNet	86.3	76.9
	ConvNet+Mixup	87.2	87.2
	ChoiceNet	**98.2**	**97.6**
45%	ConvNet	76.1	69.8
	ConvNet+Mixup	74.7	74.7
	ChoiceNet	**94.7**	**89.0**
47%	ConvNet	72.5	64.4
	ConvNet+Mixup	69.2	68.2
	ChoiceNet	**88.5**	**80.0**

experiments are depicted in Figure 5. Particularly, we observe unstable learning curves regarding the test accuracies of ConvNet and Mixup. As training accuracies of such methods show stable learning behaviors, this can be interpreted as the networks are simply memorizing noisy labels. In the contrary, the learning curves of ChoiceNet show stable behaviors which clearly indicates the robustness of the proposed method.

The experimental results and learning curves of the random shuffle experiments are shown in Table 8 and Figure 6. The convolutional neural networks trained with Mixup show robust learning behaviors when 80% of the training labels are uniformly shuffled. However, given an extremely noisy dataset (90% and 95%), the test accuracies of baseline methods decrease as the number of epochs increases. ChoiceNet shows outstanding robustness to the noisy dataset in that the test accuracies do not drop even after 50 epochs for the cases where the corruption rates are below 90%. For the 95% case, however, over-fitting is occured in all methods.

Table 9 and Figure 7 illustrate the results of the random permutation experiments. Specifically, we change the labels of randomly selected training data using a permutation rule: $(0, 1, 2, 3, 4, 5, 6, 7, 8, 9) \rightarrow (7, 9, 0, 4, 2, 1, 3, 5, 6, 8)$ following [12]. We argue that this setting is more arduous than the random shuffle case in that we are intentionally changing the labels based on predefined permutation rules.

Table 8: Test accuracies on the MNIST dataset with corrupt label.

Corruption p	Configuration	Best	Last
50%	ConvNet	97.1	95.9
	ConvNet+Mixup	98.0	97.8
	ChoiceNet	**99.1**	**99.0**
80%	ConvNet	90.6	79.0
	ConvNet+Mixup	95.3	95.1
	ChoiceNet	**98.3**	**98.3**
90%	ConvNet	76.1	54.1
	ConvNet+Mixup	78.6	42.4
	ChoiceNet	**95.9**	**95.2**
95%	ConvNet	50.2	31.3
	ConvNet+Mixup	53.2	26.6
	ChoiceNet	**84.5**	**66.0**

Table 9: Test accuracies on the MNIST dataset with randomly permutated label.

Corruption p	Configuration	Best	Last
25%	ConvNet	94.4	92.2
	ConvNet+Mixup	97.6	97.6
	ChoiceNet	**99.2**	**99.2**
40%	ConvNet	77.9	71.8
	ConvNet+Mixup	84.0	83.0
	ChoiceNet	**99.2**	**98.8**
45%	ConvNet	68.0	61.4
	ConvNet+Mixup	68.9	55.8
	ChoiceNet	**98.0**	**97.1**
47%	ConvNet	58.2	53.9
	ConvNet+Mixup	60.2	53.4
	ChoiceNet	**92.5**	**86.1**

CIFAR-10 When using the CIFAR-10 dataset, we followed two different settings from [15] and [5] for more comprehensive comparisons. Note that the setting in [5] incorporates both symmetric and asymmetric noises.

For the first scenario following [15], we apply symmetric noises to the labels and vary the corruption probabilities from 50% to 80%. We compare our method with Mixup [15], VAT [8], and MentorNet [7].[2] We adopt WideResNet (WRN) [14] with 22 layers and a widening factor of 4. To construct ChoiceNet, we replace the last layer of WideResNet with a MCDN block. We set $K = 3$, $\rho_{\max} = 0.95$, $\lambda_{\text{reg}} = 0.0001$, and ρ_k, π_k, Σ_0 modules consist of two fully connected layers with 64 hidden units and a ReLU activation function. We train each network for 300 epochs with a minibatch size of 256. We begin with a learning rate of 0.1, and it decays by $1/10$ after 150 and 225 epochs. We apply random horizontal flip and random crop with $4-$pixel-padding and use a weight decay of 0.0001 for the baseline network as [6]. To train ChoiceNet, we set the weight decay rate to $1e-6$ and apply gradient clipping at 1.0.

Table 10 shows the test accuracies of compared methods under different symmetric corruptions probabilities. In all cases, ConvNet+CN outperforms the compared methods. We would like to emphasize that when ChoiceNet and Mixup [15] are combined, it achieves a high accuracy of 75% even on the 80% shuffled dataset. We also note that ChoiceNet (without Mixup) outperforms WideResNet+Mixup when the corruption ratio is over 50% on the last accuracies.

We conduct additional experiments on the CIFAR-10 dataset to better evaluate the performance on both both symmetric and asymmetric noises following [5]: Pair-45%, Symmetry-50%, and Symmetry-20%. Pair-45% flips 45% of each label to the next label, e.g., randomly flipping 45% of label 1 to label 2 and label 2 to label 3, and Symmetry-50% randomly assigns 50%

[2] We use the authors' implementations available online.

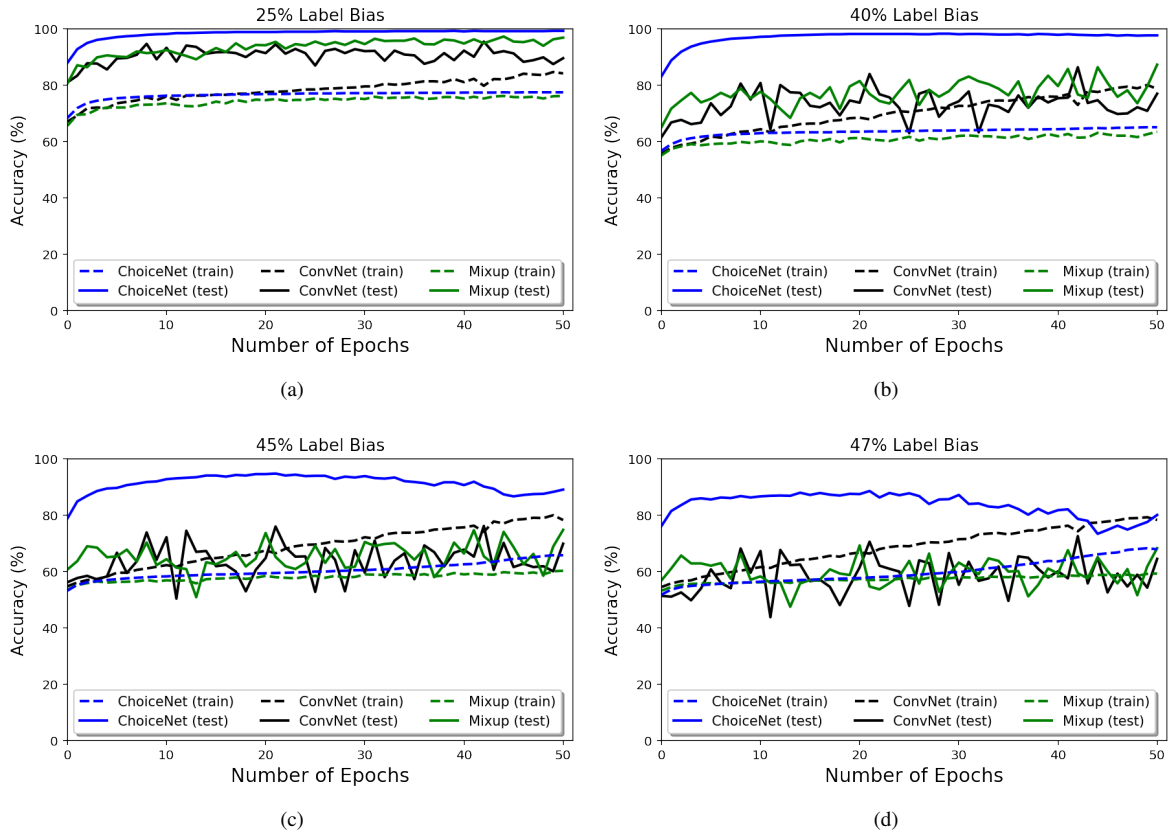


Figure 5: Learning curves of compared methods on random bias experiments using MNIST with different noise levels.

Table 10: Test accuracies on the CIFAR-10 datasets with symmetric noises.

Corruption p	Configuration	Accuarcy (%)
50%	ConvNet	59.3
	ConvNet+CN	84.6
	ConvNet+Mixup	83.1
	ConvNet+Mixup+CN	**87.9**
	MentorNet	49.0
	VAT	71.6
	MDN	58.6
80%	ConvNet	27.4
	ConvNet+CN	65.2
	ConvNet+Mixup	62.9
	ConvNet+Mixup+CN	**75.4**
	MentorNet	21.4
	VAT	16.9
	MDN	22.7

of each label to other labels uniformly. We implement the a 9-layer CNN architecture following VAT [8] and Co-teaching [5] for fair and accurate evaluations and set other configurations such as the network topology and activations to be the same as [5]. We also copied some results in [5] for better comparisons. Here, we compared our method with MentorNet [7], Co-teaching [5], and F-correction [9].

While our proposed method outperforms all compared methods on the symmetric noise settings, it shows the second best

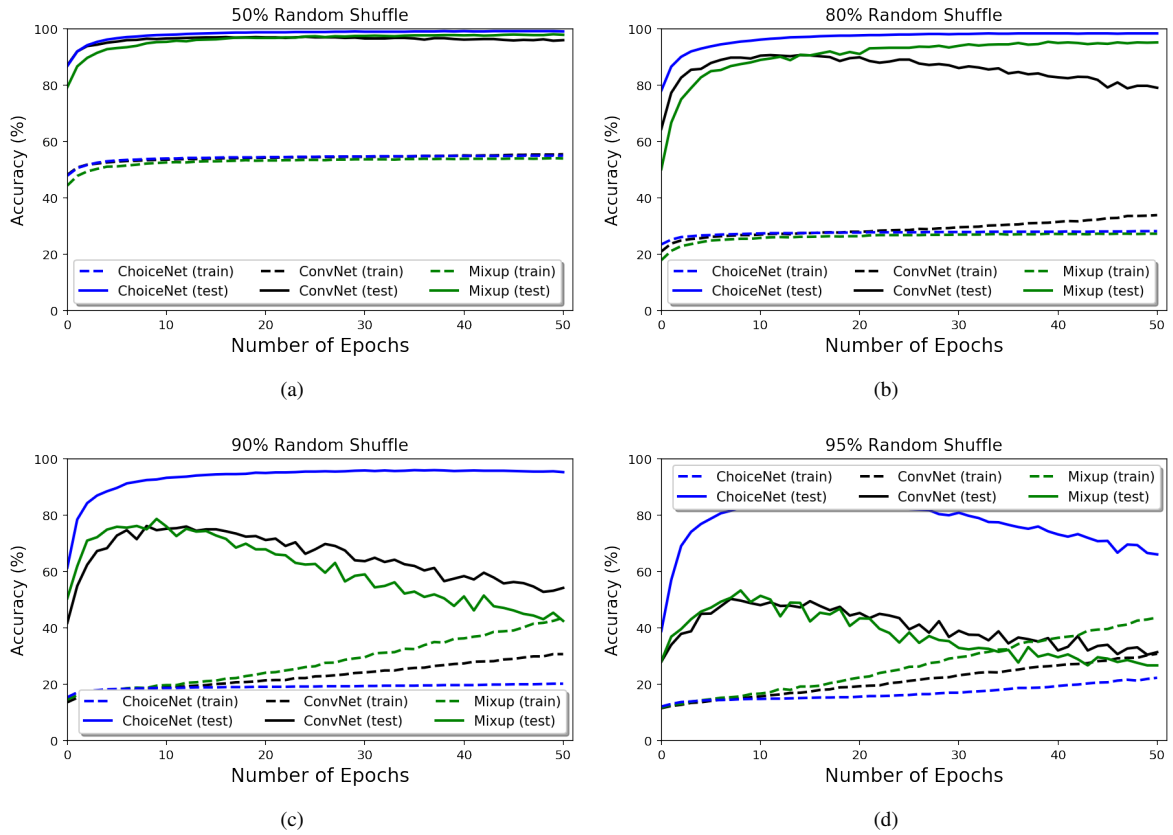


Figure 6: Learning curves of compared methods on random shuffle experiments using MNIST with different noise levels.

Table 11: Test accuracies on the CIFAR-10 dataset with by symmetric and asymmetric noises.

	Pair-45%	sym-50%	sym-20%
ChoiceNet	70.3	**85.2**	**91.0**
MentorNet	58.14	71.10	80.76
Co-teaching	**72.62**	74.02	82.32
F-correction	6.61	59.83	59.83
MDN	51.4	58.6	81.4

performance on asymmetric noise settings (Pair-45%). This shows the weakness of the proposed method. In other words, as Pair-45% assigns 45% of each label to its next label, the MCDN fails to correctly infer the dominant label distributions. However, we would like to note that Co-teaching [5] is complementary to our method where one can combine these two methods by using two ChoiceNets and update each network using Co-teaching. However, it is outside the scope of this paper.

Here, we also present detailed learning curves of the CIFAR-10 experiments while varying the noise level from 20% to 80% following the configurations in [15].

Large Movie Review Dataset We also conduct a natural language processing task using a Large Movie Review dataset which consists of $25,000$ movie reviews for training and $25,000$ reviews for testing. Each movie review (sentences) is mapped to a 128-dimensional embedding vector using a feed-forward Neural-Net Language Models [2]. We evaluated the robustness of the proposed method with Mixup [15], VAT [8], and naive MLP baseline by randomly flipping the labels with a corruption probability p. In all experiments, we used two hidden layers with 128 units and ReLU activations. The test accuracies of the compared methods are shown in Table 12. ChoiceNet shows the superior performance in the presence of outliers where we

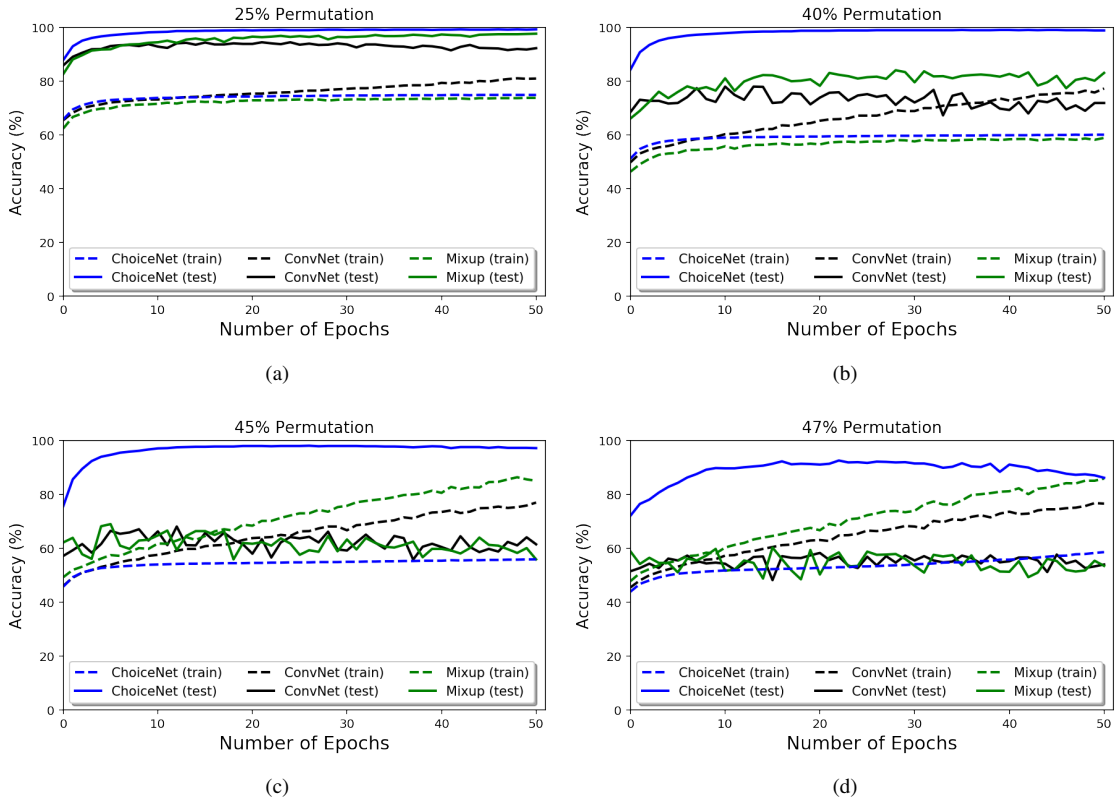


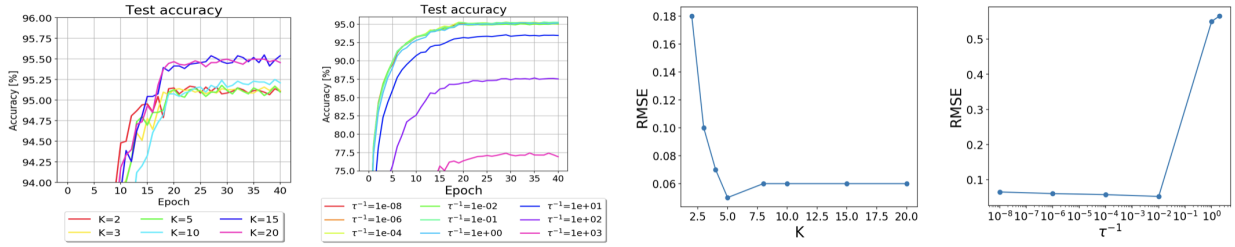
Figure 7: Learning curves of compared methods on random permutation experiments using MNIST with different noise levels.

Table 12: Test accuracies on the Large Movie Review dataset with different corruption probabilities.

Corruption p	0%	10%	20%	30%	40%
ChoiceNet	79.43	**79.50**	**78.66**	**77.1**	**73.98**
Mixup	**79.77**	78.73	77.58	75.85	69.63
MLP	79.04	77.88	75.70	69.05	62.83
VAT	76.40	72.50	69.20	65.20	58.30

observe that the proposed method can be used for NLP tasks as well.

B.3. Ablation Study on MNIST



Above figures show the results of ablation study when varying the number of mixture K and the expected measurement variance τ^{-1}. Left two figures indicate test accuracies using the MNIST dataset where 90% of train labels are randomly shuffled and right two figures are RMSEs using a synthetic one-dimensional regression problem in Section 4.1. We observe that having bigger K is beneficial to the classification accuracies. In fact, the results achieved here with K equals 15 and 20

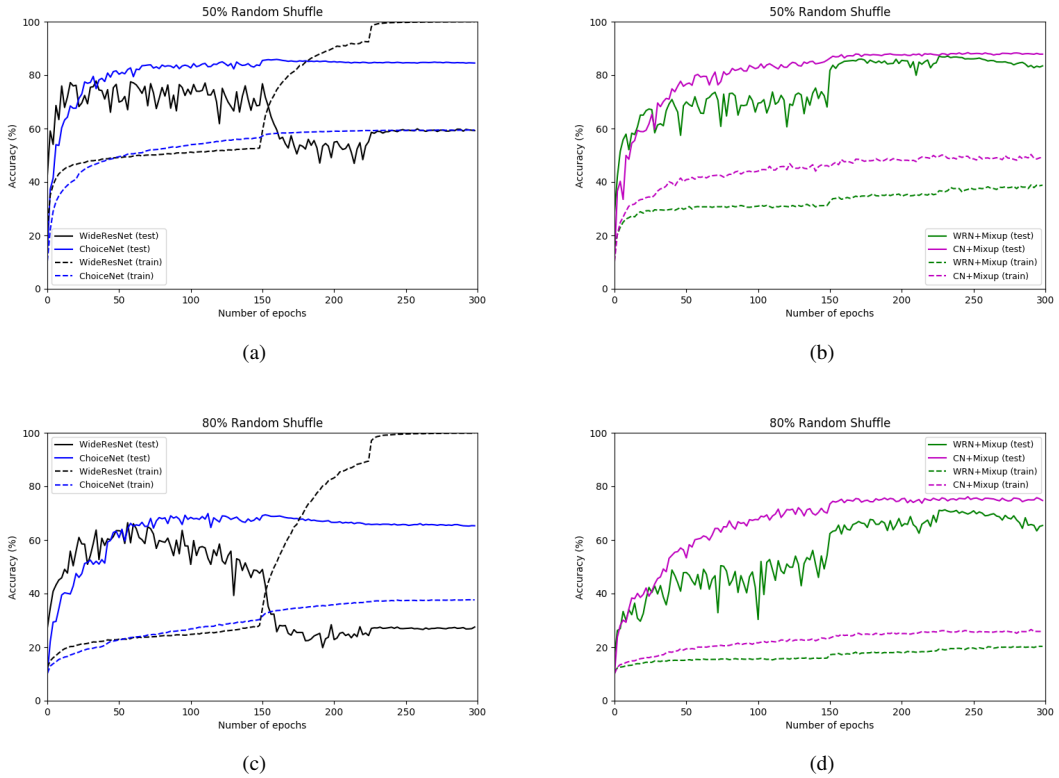


Figure 8: Learning curves of compared methods on CIFAR-10 experiments with different noise levels.

are better than the ones reported in the submitted manuscript. τ^{-1} does not affect much unless it is exceedingly large.

References

[1] V. Belagiannis, C. Rupprecht, G. Carneiro, and N. Navab. Robust optimization for deep regression. In *Proc. of the IEEE International Conference on Computer Vision*, pages 2830–2838, 2015.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[3] S. Choi, K. Lee, and S. Oh. Robust learning from demonstration using leveraged Gaussian processes and sparse constrained opimization. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2016.

[4] B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.-F. F. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *Proc. of the Advances in Neural Information Processing Systems*, pages 3041–3049, 2014.

[5] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Co-teaching: robust training deep neural networks with extremely noisy labels. In *Proc. of the Advances in Neural Information Processing Systems*, 2018.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei. Mentornet: Regularizing very deep neural networks on corrupted labels. *arXiv preprint arXiv:1712.05055*, 2017.

[8] T. Miyato, S.-i. Maeda, S. Ishii, and M. Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[9] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu. Making deep neural networks robust to label noise: a loss correction approach. In *Proc. of the Conference on Computer Vision and Pattern Recognition*, volume 1050, page 22, 2017.

[10] C. E. Rasmussen. Gaussian processes for machine learning. 2006.

[11] C. E. Rasmussen and Z. Ghahramani. Infinite mixtures of gaussian process experts. In *Advances in Neural Information Processing Systems*, pages 881–888, 2002.

[12] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

[13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[14] S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016.

[15] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *Proc. of International Conference on Learning Representations*, 2017.