Supplementary Materials for Beyond Synthetic Noise: Deep Learning on Controlled Noisy Labels

Lu Jiang   Di Huang   Mason Liu   Weilong Yang

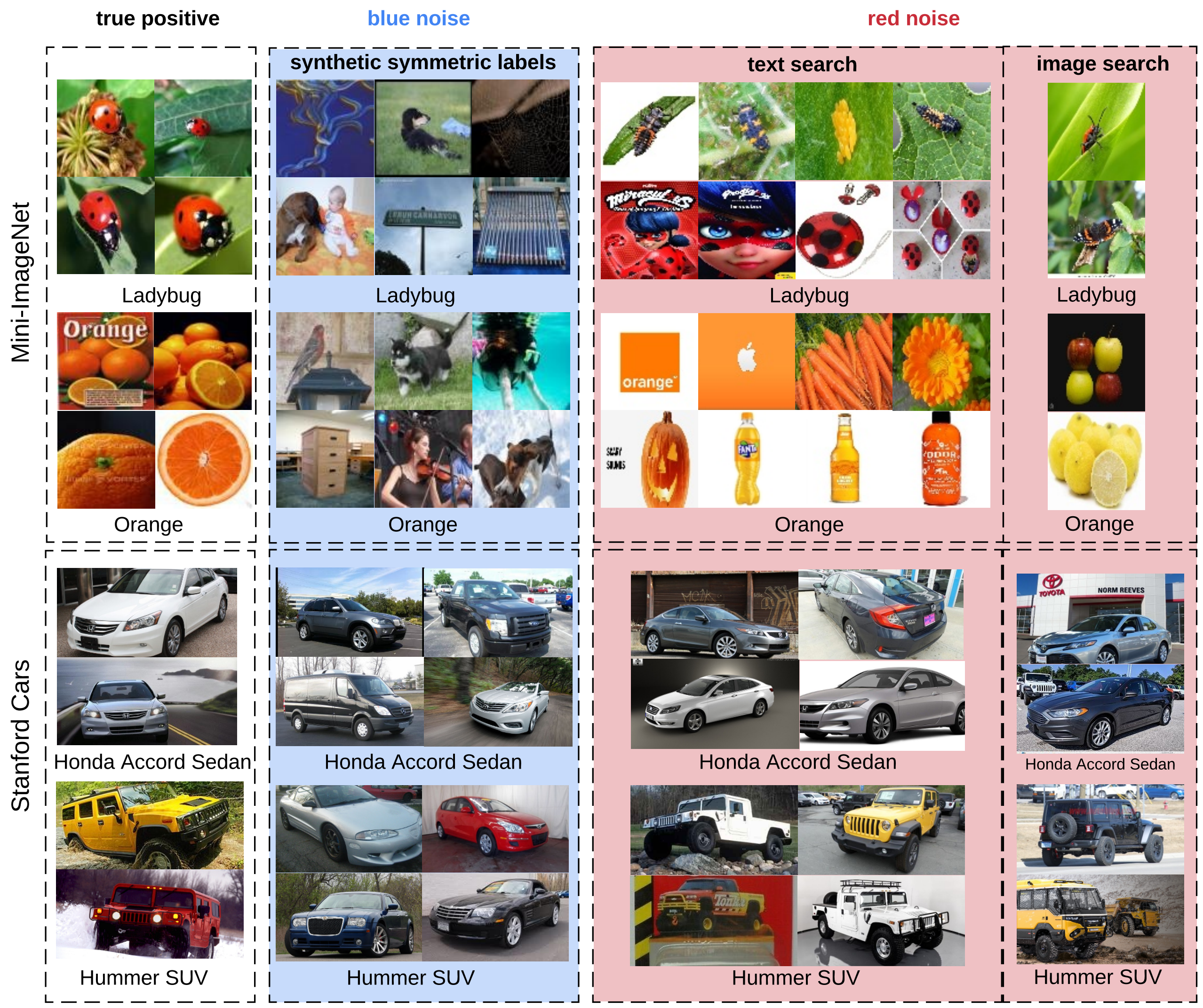


Figure 1. Comparison of symmetric label noise (Blue noise) and web label noise (Red noise). From left to right, columns are true positive images, images with incorrect symmetric labels, and images with incorrect web labels from text-to-image search and image-to-image search, respectively. The image-to-image search results (the last column) only account for 18% in our dataset and fewer images are shown as a result.

A. Dataset Overview:

Fig. 1 shows some example images with correct labels and incorrect labels of symmetric label noise (blue noise) and web label noise (red noise). There are three clear distinctions between images with the synthetic and web label noise as

Table 1. Summary of the difference of images with blue and red noisy labels.

Difference	Blue Noise	Red Noise
Visual & semantic similarity to true positive images	Low	High
Level of the label noise	instance-level	class-level
Latent class vocabulary from which images are sampled	Fixed vocabulary	Open vocabulary

summarized in Table 1. First, images with noise from the web (or red noise) are more relevant (visually or semantically) to true positive images. Second, synthetic noise (symmetric or asymmetric) is at class-level which means all examples in the same class are treated equally. Web label noise is at instance-level in which certain images are more likely to be mislabelled than others. For example, "Honda Civic" images taken from the side view are more likely to be confused with "Honda Accord" as the two models are lookalike from the side view. Such confusion is rare for car images taken from the front view. Third, images with noise from the web come from an open vocabulary outside the class vocabulary of Mini-ImageNet or Stanford Cars. For example, the noisy images of "ladybug" include "fly" and other bugs that do not belong to any of the classes in Mini-ImageNet.

Fig. 2 illustrates the distribution of correctly-labeled and incorrectly-labeled images across classes, where symmetric label noise is in blue and web label noise is in red. It is worth noting that it is not a feasible option for us to annotate existing datasets of web labels, *e.g.* WebVision (Li et al., 2017) or Clothing-1M (Xiao et al., 2015). Due to their imbalanced class distribution, for many classes, we simply cannot find sufficient images with incorrect labels to label in these datasets.

As incorrect images are rare in a few common classes (*e.g.* hotdog), we need to limit the size of Red Mini-ImageNet to 50K such that every class can get sufficient incorrect images at every noise level. Recall the role of the blue noisy datasets is to confirm existing findings on symmetric noise. Initially, we made the Blue and Red Mini-ImageNet to be the same size of 50K examples. However, we found that existing findings on symmetric noise hold on both the full (60K) and subset (50K) of Blue Mini-ImageNet. In the end, we decided to report the results on the 60K full set which results in a larger size of Blue Mini-ImageNet. This design would not affect our main contributions for the following reasons. First, on our second dataset Stanford Cars, the blue and red set have the same size. Second, our method has been verified by extensive experiments on many other datasets. Third, our main findings are on red noise which may not be affected by the size of the blue set.

B. Detailed Experimental Setups

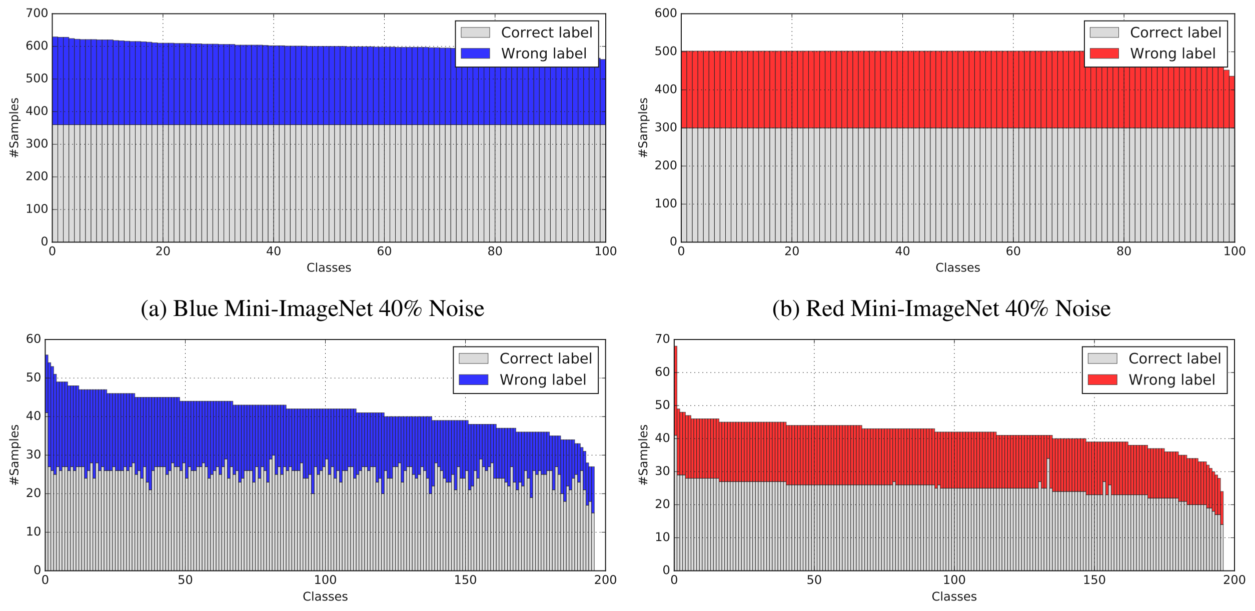
This section presents detailed setups for training and testing used in our experiments.

B.1. Setup on the proposed dataset

Network architectures. Table 2 lists the parameter count and input image size for each network architecture used in our experiments. We obtained their model checkpoints trained on the ImageNet 2012 dataset from TensorFlow Slim[1], EfficienNet TPU[2], and from Kornblith et al. (2019). The last two columns list the top-1 accuracy of our obtained models along with the accuracy reported in the original paper. As shown, the top-1 accuracy of these architectures on the ImageNet ILSVRC 2012 validation ranges from 71.6% to 83.6%. We select the above architectures to be representative of diverse capacities.

Training from scratch (random initialization). For vanilla training, we trained each architecture on the clean dataset (0% noise level) to find the optimal training setting by grid search. Our grid search consisted of 6 start learning rates of $\{1.6, 0.16, 1.0, 0.5, 0.1, 0.01\}$ and 3 learning rate decay epochs of $\{1, 2, 3\}$. The exponential learning rate decay factor was fixed to 0.975. We trained the network to full convergence, and the maximum epoch to train was 200 on Mini-ImageNet (Red and Blue) and 300 epochs on Stanford Cars (Blue and Red), where the learning rate warmup (Goyal et al., 2017) was used in the first 5 epochs. The training was using Nesterov momentum with a momentum parameter of 0.9 with a batch size of 64, taking an exponential moving average of the weights with a decay factor of 0.9999. We had to reduce the batch size to 8 for EfficientNet for its larger image input. Following (Kornblith et al., 2019), our vanilla training was with batch normalization layers but without label smoothing, dropout, or auxiliary heads. We employed the standard prepossessing in

[1]https://github.com/tensorflow/models/tree/master/research/slim
[2]https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet



(a) Blue Mini-ImageNet 40% Noise

(b) Red Mini-ImageNet 40% Noise

(c) Blue Stanford Cars 20% Noise

(d) Red Stanford Cars 20% Noise

Figure 2. The distribution of images with correct and incorrect labels in Mini-ImageNet and Stanford Cars. The grey, blue, and red bar represent images of correct labels, symmetric noisy labels, and web noisy labels, respectively. Classes are ranked by the number of training examples. Better viewed in color.

Table 2. Overview of the ImageNet architectures used in our study.

Network	Parameters	Image Size	ImageNet Top-1 Acc. Paper	ImageNet Top-1 Acc. Our checkpoint
EfficientNet B5 (Tan & Le, 2019)	28.3M	456	83.3	83.3
Inception V2 (Ioffe & Szegedy, 2015)	10.2M	224	74.8	73.9
Inception V3 (Szegedy et al., 2016)	21.8M	299	78.8	78.6
Inception-ResNet V2 (Szegedy et al., 2017)	54.2M	299	80.0	80.3
MobileNet V2 (Sandler et al., 2018)	2.2M	224	72.0	71.6
ResNet 50 V1 (He et al., 2016)	23.5M	224	75.2	75.9
ResNet 101 V1 (He et al., 2016)	42.5M	224	76.4	77.5

EfficientNet[3] for data augmentation and evaluated on the central cropped images on the validation set. Training in this way, we obtained reasonable performance on the clean Stanford Cars validation set. For example, our Inception-ResNet-V2 got 90.8 (without dropout) and 92.4 (with dropout) versus 89.9 reported in (Kornblith et al., 2019).

Fine-tuning from ImageNet checkpoints. For fine-tuning experiments, we initialized networks with ImageNet-pretrained weights. We used a similar training protocol for fine-tuning as training from scratch. The start learning rate was stable in fine-tuning so we fixed it to 0.01 and only searched the learning rate decay epochs in $\{1, 3, 5, 10\}$. Learning rate warmup was not used in fine-tuning. As fine-tuning converges faster, we scaled down the maximum number of epochs to train by a factor of 2 and trained the network to full convergence. Training in this way, we obtained reasonable performance on the clean Stanford Cars test set. For example, our Inception-ResNet-V2 got 92.4 versus 92.0 reported in (Kornblith et al., 2019) and our EfficientNet-B5 got 93.8% versus 93.6% reported in (Tan & Le, 2019).

Baseline comparison. For method comparison, we used Inception-ResNet as the default network. All methods employed the identical setting discussed above, including the same start learning rate, learning rate decay factor, batch size, and the maximum number of epochs to train. For Dropout, as it converges slower, we added another 100 epochs to its #maximum

[3]https://github.com/tensorflow/tpu/blob/master/models/official/efficientnet/preprocessing.py

epochs to train. We extensively searched the hyperparameter for each method on every noise level using the hyperparameter range discussed in the main manuscript. The performance variance under different hyperparameters can be found in Fig. 5, Fig. 6, Fig. 7, and Fig. 8, where the black line shows the 95% confidence interval of the accuracy under all searched hyperparameters.

B.2. Setup on public benchmarks: CIFAR and WebVsion

CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009) consist of 32 $\times$ 32 color images arranged in 10 and 100 classes. Both datasets contain 50,000 training and 10,000 validation images. The ResNet-32 (He et al., 2016) with standard data augmentation function was used as our network backbone. In training, the batch size was set to 128 and we trained 400K iterations for the ResNet model by a distributed asynchronized momentum SGD optimizer (momentum = 0.9) on 8 V100 GPUs. We used the common learning rate scheduling strategy: setting the starting learning rate as 0.1 and using the step-wise exponential learning rate decay which multiplies it by 0.9 every 20K iterations. In this way, training ResNet-32 on the clean training dataset, the validation accuracy reaches 95.2% on CIFAR-10, and 78.0% on CIFAR-100. We searched the hyperparameters of MentorMix in the following range: $\alpha = \{2, 4, 8, 32\}$ and $\gamma_p = \{0.8, 0.7, 0.6, 0.5, 0.4, 0.2, 0.1\}$. We used a simple MentorNet called "Predefined MentorNet" from (Jiang et al., 2018)[4] which, as discussed in the main paper, computes the weight by a thresholding function $v_i^* = \mathbf{1}(\ell(\mathbf{x}_i, y_i) \leq \gamma)$.

WebVision 1.0 (Li et al., 2017) contains 2.4 million images of real-world noisy labels, crawled from the web using the 1,000 concepts in ImageNet ILSVRC12. We downloaded the resized images from the official website[5]. The inception-resnet v2 (Szegedy et al., 2017) was used as our network backbone. In training, the batch size was set to 64 and we trained 4M iterations using a distributed asynchronized momentum optimizer on 32 V100 GPUs. The start learning rate was 0.01 and was discounted by a factor of 0.95 every 562K steps. The weight decay was $4e{-}5$. The batch norm was used and its decay was set to 0.9997 and the epsilon was 0.001. The default data augmentation for the ResNet model is used. We also tested our method on the WebVision mini-training set that contains about 61K Google images on the first 50 classes. All the models were evaluated on the clean ILSVRC12 and WebVision validation set. The best hyperparameter is $\gamma_p = 0.7$ and $\alpha = 0.4$ for both the WebVision full training set and the WebVision mini-training set.

C. Alternative Approaches for Dataset Construction

In this section, we study two alternative approaches to construct our red datasets. Our goal is to verify whether our findings in Section 5.3 of the main paper are consistent when the datasets are constructed differently. Please note that it may not be necessary to verify the proposed method MentorMix on these new datasets as it has already been verified in Section 5.2 on the two public benchmarks (CIFAR and WebVision) that contain both synthetic and real-world noisy labels. We consider the following settings to construct our red datasets:

- **Setting 0 (default)** uses the approach discussed in the main paper where we replace the clean images in Mini-ImageNet and Stanford Cars with incorrectly labeled images from the web while leaving the label unchanged. The advantage of our approach is that we closely match the construction of synthetic datasets while still being able to introduce controlled levels of noise that better resembles realistic label noise distributions.

- **Setting 1 (web images only)**: in this setting, the red datasets only contain images from the web (with both correct and incorrect labels). No clean images in the original Mini-ImageNet or Stanford Cars datasets are used. This setting is used to understand the impact of the domain difference in Setting 0.

- **Setting 2 (no image-to-image results)**: this setting is the same as setting 0 except only web images obtained from the text-to-image search are used. This setting examines the impact after removing the image-to-image search label noise.

First, we show that DNNs generalize much better on the red label noise under all three settings. We compare the standard deviation of the final accuracies at 10 noise levels (0%-80%) in Table 3. A higher standard deviation suggests a poorer generalization performance when DNNs are trained on noisy labels. Ideally, we expect to observe a significantly smaller standard deviation in web noise. Table 3 shows the standard deviation of the red noise is at least two times less than that of

[4]https://github.com/google/mentornet

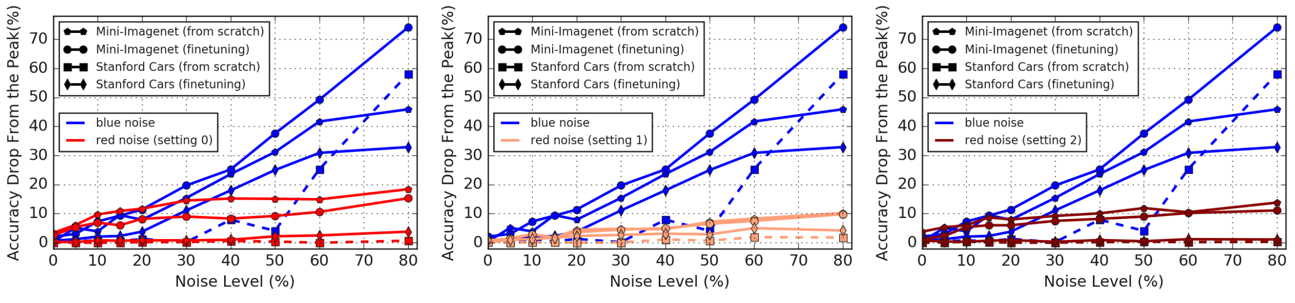
[5]https://www.vision.ee.ethz.ch/webvision/download.html

the blue noise. The standard deviations of red noise are comparable among all three settings. These results show that DNNs generalize much better on red label noise despite specific approaches used to construct the dataset.

Table 3. Comparison of the standard deviation of the final accuracies across 10 noise levels. A higher standard deviation suggests a poorer generalization performance of DNNs trained on noisy labels.

Noise Settings	Mini-ImageNet		Stanford Cars	
	Fine-tuned	Trained from scratch	Fine-tuned	Trained from scratch
Blue Noise	0.205	0.195	0.268	0.347
Red Noise (setting 0)	0.051	0.088	0.104	0.146
Red Noise (setting 1)	0.046	0.068	0.104	0.127
Red Noise (setting 2)	0.056	0.077	0.067	0.096

Second, we show that DNNs may not learn patterns first on red noise *i.e.* DNNs are able to automatically learn generalizable "patterns" in the early training stage before memorizing all noisy training labels. This is manifested by the gap between the peak and final validation accuracy. Fig. 3 illustrates the relative difference, namely the drop, between the peak and final accuracy on the clean validation set. Recall a larger drop between the peak and final validation accuracy means a better pattern is found in the early training stage. As it shows, the drops of red noise under all three settings are significantly and consistently smaller than that of the blue noise. These results show the domain differences and the types of label noise (image-to-image search) do not distort this finding.



(a) Red Noise (Setting 0) (b) Red Noise (Setting 1) (c) Red Noise (Setting 2)

Figure 3. Performance drop from the peak accuracy across 10 noise levels. Different colors are used to differentiate noise types. A larger drop (y-axis) means a better pattern is found during the early training stage

Finally, we show that ImageNet architectures generalize on noisy labels when the networks are fine-tuned. To do so, we compute Pearson correlation r for different red noise settings. The results are shown in Fig. 4, where the x-axis is the accuracy of the pretrained architectures on ImageNet and the y-axis shows the peak validation accuracy on noisy datasets. The bar plots the 95% confidence interval across 10 noise levels, where the center dot marks the mean. As it shows, the correlation is consistent across all types of label noise where the Pearson correlations are shown in the parentheses. These results show a better pretrained architecture is likely to perform better when it is fine-tuned on noisy training labels. This finding seems to be consistent across all types of label noise.

D. Detailed Method Comparison

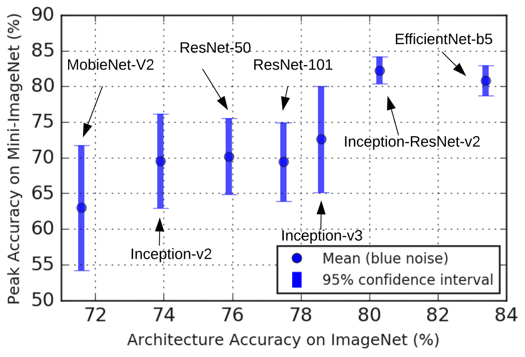
This subsection presents detailed comparison results on our datasets. To be specific, we show the peak/final accuracy on the clean validation set in four tables: Table 4, Table 5, Table 6, and Table 7, where the best trial out of all searched hyperparameters is shown. The performance variance of all searched hyperparameters is shown in four figures: Fig. 6, Fig. 5, Fig. 8, and Fig. 7, where the black line shows the 95% confidence interval. The results in all tables and figures show that the proposed method MentorMix consistently outperforms baseline methods and has comparable performance variance in the searched hyperparameter range.

Table 4. Best peak accuracy (%) for baseline methods fine-tuned on Mini-ImageNet. The peak and final validation accuracies are shown in the format of XXX/YYY.

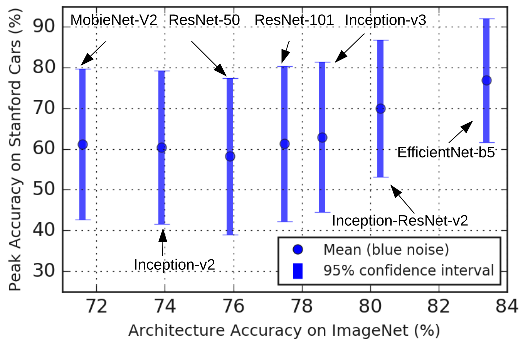
Type	Noise Level	Vanilla	WeDecay	Dropout	S-Model	Reed Soft	Mixup	MentorNet	MentorMix
Blue Noise	0	85.1/83.3	84.5/81.9	85.0/83.6	85.2/83.5	**85.6**/84.3	83.4/82.0	84.8/83.4	85.0/83.9
	5	84.5/82.0	84.2/79.7	84.5/80.8	84.1/81.5	84.7/81.9	83.5/82.0	**84.9**/84.5	84.8/84.0
	10	83.9/77.8	83.8/75.9	84.3/78.0	84.3/78.2	84.4/81.5	83.3/80.1	84.5/84.3	**85.2**/83.6
	15	83.6/75.8	83.1/73.8	83.6/74.7	83.7/76.1	**84.7**/80.4	84.0/79.5	84.5/81.2	84.6/83.2
	20	83.5/74.0	82.8/71.9	83.5/73.9	83.9/73.7	84.5/79.2	82.9/77.9	84.3/76.5	**84.5**/83.6
	30	82.9/66.6	82.8/64.1	82.4/67.5	82.4/66.0	83.0/76.7	83.3/74.6	83.2/74.0	**84.5**/82.8
	40	81.6/61.0	81.1/58.5	82.4/59.0	82.2/60.2	83.0/78.3	81.8/70.8	82.5/80.4	**84.3**/81.3
	50	80.9/50.5	80.5/47.9	82.0/68.1	80.0/50.8	81.6/53.1	79.9/64.6	81.8/71.0	**83.7**/79.0
	60	80.5/40.9	79.7/40.3	81.4/38.9	80.7/42.3	81.6/45.2	79.2/63.6	81.0/79.8	**83.4**/77.9
	80	76.1/19.7	76.5/17.6	78.9/24.7	76.9/20.2	77.8/22.2	76.1/60.0	77.5/32.3	**82.0**/73.3
Red Noise	0	84.7/82.6	83.6/81.9	84.3/83.0	**85.0**/83.3	84.8/82.5	83.0/81.8	84.6/83.5	84.4/83.4
	5	84.6/80.0	84.0/78.9	84.6/80.2	84.7/80.4	85.2/80.9	84.8/82.3	84.6/84.0	**85.3**/85.0
	10	83.9/78.1	83.9/78.1	83.9/78.1	84.5/78.4	84.3/78.6	84.1/80.8	84.3/84.1	**85.1**/85.0
	15	82.3/77.4	83.0/76.5	83.1/76.5	83.0/77.5	84.3/78.0	83.9/80.8	83.6/82.9	**85.5**/84.7
	20	82.9/76.1	82.9/75.6	82.7/75.5	82.5/76.9	84.3/77.4	83.2/79.4	83.5/83.4	**85.0**/84.6
	30	82.0/74.6	81.2/74.6	81.6/73.7	82.5/73.8	83.0/74.7	83.3/78.6	82.7/82.6	**83.9**/81.4
	40	80.7/74.0	81.2/71.7	81.2/72.6	81.4/73.9	82.3/73.1	82.4/77.6	81.9/80.2	**83.1**/81.5
	50	80.3/72.9	80.3/71.4	80.7/71.9	80.5/72.2	81.7/72.4	82.0/77.3	81.1/79.3	**82.2**/80.4
	60	78.6/70.3	79.2/68.8	80.1/69.4	78.8/70.1	80.7/69.5	80.4/75.5	80.5/73.4	**81.0**/77.1
	80	76.3/64.6	76.1/63.1	76.1/63.4	76.7/65.0	76.7/65.3	76.6/71.3	76.8/72.8	**77.2**/74.0

Table 5. Best peak accuracy (%) for baseline methods trained from scratch on Mini-ImageNet. The peak and final validation accuracies are shown in the format of XXX/YYY. '-' denotes the method that has failed to converge.

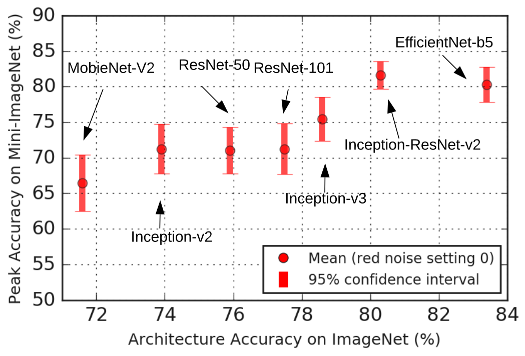
Type	Noise Level	Vanilla	WeDecay	Dropout	S-Model	Reed Soft	Mixup	MentorNet	MentorMix
Blue Noise	0	73.1/72.8	-/-	73.1/67.9	73.8/71.8	73.8/71.2	73.7/73.0	73.2/71.5	**75.1**/74.4
	5	70.9/70.7	-/-	70.2/63.4	71.1/67.1	71.1/66.7	72.4/70.3	72.5/69.5	**75.0**/74.7
	10	69.0/63.9	-/-	68.5/60.5	68.1/63.6	69.2/63.4	70.2/67.2	70.2/67.9	**73.6**/73.3
	15	67.1/60.7	-/-	65.6/56.7	66.6/61.1	67.7/60.5	68.1/63.2	69.7/66.1	**73.6**/73.2
	20	63.0/58.0	-/-	65.1/53.4	63.5/57.7	65.2/57.6	66.5/60.6	67.4/65.8	**73.9**/73.3
	30	59.9/50.7	-/-	61.5/47.2	62.4/50.8	63.6/52.2	63.0/54.6	66.0/64.2	**72.3**/71.6
	40	56.9/43.5	-/-	58.1/40.2	57.3/45.0	60.2/44.5	60.3/46.9	62.5/62.1	**70.4**/69.4
	50	52.9/36.4	-/-	54.3/33.5	50.9/36.1	54.1/36.8	55.3/42.8	59.6/57.7	**69.2**/66.7
	60	44.6/26.0	-/-	48.2/23.3	46.7/27.1	47.2/27.2	48.9/47.4	52.0/47.2	**66.5**/62.9
	80	25.9/14.0	-/-	28.4/18.8	26.5/18.7	29.0/12.7	28.5/28.5	25.4/18.5	**59.9**/52.7
Red Noise	0	70.9/68.5	-/-	71.8/65.7	71.4/68.4	71.8/68.4	72.8/72.3	71.2/68.9	**74.3**/73.7
	5	70.9/66.6	-/-	71.8/62.8	71.2/67.0	71.9/66.7	71.8/69.4	71.5/67.4	**73.6**/73.4
	10	70.8/63.9	-/-	71.0/61.3	69.8/63.9	70.4/63.6	71.1/68.3	70.8/65.6	**73.0**/71.4
	15	69.8/62.2	-/-	69.3/60.0	69.0/60.5	69.3/62.2	69.9/65.9	69.7/66.3	**71.5**/70.8
	20	68.3/60.3	-/-	68.7/57.6	67.9/60.3	68.3/60.4	69.3/64.4	67.9/62.8	**70.1**/69.1
	30	66.1/56.5	-/-	66.6/55.0	65.2/56.3	66.6/56.7	66.8/61.8	66.2/64.0	**68.3**/67.2
	40	64.5/54.7	-/-	**66.1**/53.0	64.1/54.6	64.7/54.0	65.8/59.6	63.9/56.5	66.0/64.7
	50	60.9/51.7	-/-	62.1/50.1	61.3/51.3	62.6/52.5	63.2/58.4	61.7/58.0	**63.3**/61.8
	60	57.6/49.0	-/-	59.7/46.8	57.0/47.5	59.0/49.2	59.0/53.4	58.8/51.3	**60.0**/57.5
	80	48.8/39.8	-/-	49.5/37.6	49.0/40.6	50.1/40.1	**50.7**/45.5	49.3/43.4	50.2/48.4



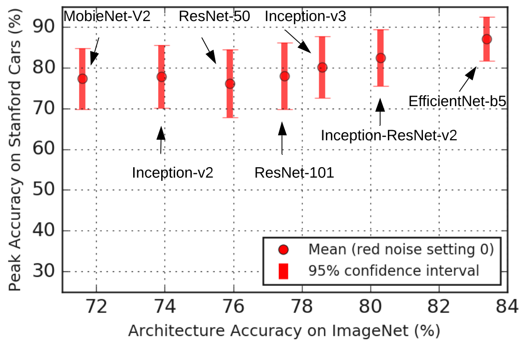
(a1) blue noise (0.898) - Mini-ImageNet



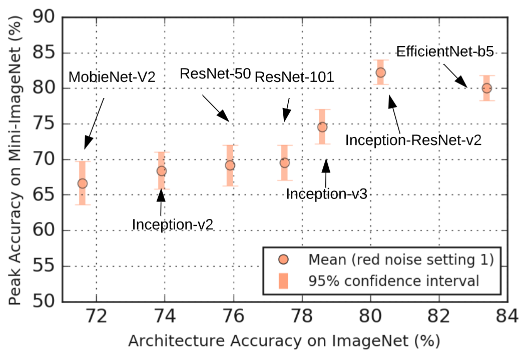
(b1) blue noise (0.845) - Stanford Cars



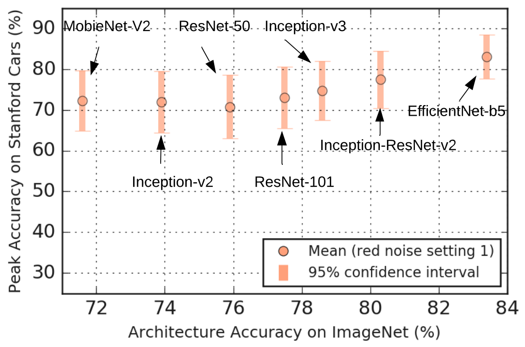
(a2) red noise setting 0 (0.912) - Mini-ImageNet



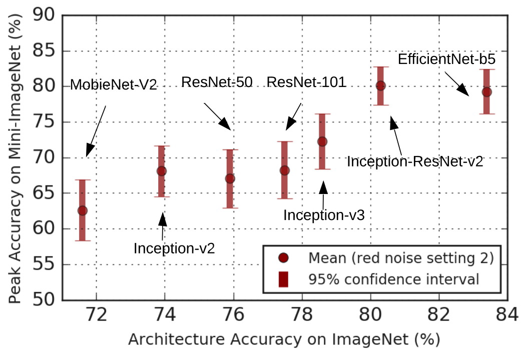
(b2) red noise setting 0 (0.876) - Stanford Cars



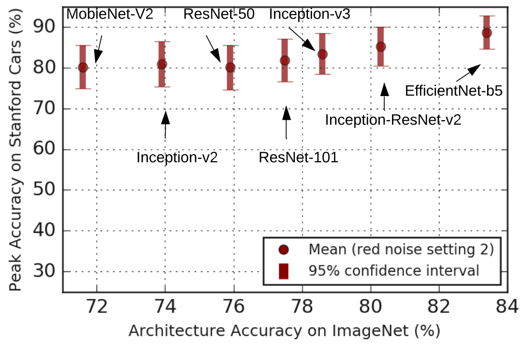
(a3) red noise setting 1 (0.884) - Mini-ImageNet



(b3) red noise setting 1 (0.864) - Stanford Cars



(a4) red noise setting 2 (0.915) - Mini-ImageNet



(b4) red noise setting 2 (0.920) - - Stanford Cars

Figure 4. Fine-tuning seven ImageNet architectures on the red and blue datasets. The number in parentheses is the Pearson correlation between the architecture's ImageNet accuracy and the performance on our red dataset. All three settings of red noise are illustrated. Better view in color.

Table 6. Best peak accuracy (%) for baseline methods trained from scratch on Stanford Cars. The peak and final validation accuracies are shown in the format of XXX/YYY.

Type	Noise Level	Vanilla	WeDecay	Dropout	S-Model	Reed Soft	Mixup	MentorNet	MentorMix
Blue Noise	0	91.2/90.6	92.4/92.2	91.9/91.3	91.0/90.7	91.3/91.0	91.7/91.6	90.1/90.0	**92.9**/92.9
	5	88.8/87.7	90.8/90.5	89.5/88.4	88.8/88.3	88.8/88.8	90.3/90.0	90.3/89.8	**92.4**/92.2
	10	86.4/84.6	89.1/87.9	87.7/85.1	85.4/84.2	85.4/87.5	89.1/88.9	89.5/89.5	**91.8**/91.8
	15	83.6/81.7	87.5/86.4	85.6/81.7	83.9/81.4	83.9/86.4	87.7/87.2	89.1/89.1	**91.3**/91.2
	20	81.3/78.2	84.9/82.9	83.7/77.4	81.2/78.1	81.2/83.9	85.6/85.6	88.1/87.8	**90.5**/90.4
	30	76.7/68.2	79.1/75.3	78.6/67.0	75.7/68.5	75.7/79.8	79.8/76.4	85.3/85.2	**87.3**/86.3
	40	69.3/56.8	72.9/63.8	71.9/56.0	69.7/58.2	69.7/71.8	73.6/68.1	80.9/79.3	**81.9**/77.7
	50	58.8/44.1	61.2/48.7	62.0/44.2	59.2/45.1	59.2/60.0	63.0/56.2	71.1/66.7	**71.7**/65.1
	60	47.5/32.8	49.4/36.9	50.9/31.8	46.0/32.4	46.0/47.8	52.0/43.6	58.5/57.0	**60.9**/52.5
	80	16.1/10.8	15.2/10.0	15.5/10.1	16.0/10.6	16.0/15.9	18.3/17.2	15.8/13.8	**21.2**/19.2
Red Noise	0	91.0/90.7	92.3/92.1	91.8/91.2	90.9/90.7	91.2/90.8	92.3/92.3	91.2/91.1	**93.2**/93.2
	5	90.3/90.1	91.7/91.7	90.6/89.6	89.8/89.2	90.3/89.6	91.9/91.8	89.7/89.3	**92.2**/92.2
	10	89.2/88.5	90.7/90.5	90.0/89.1	89.7/89.1	89.4/88.9	90.7/90.5	89.1/88.7	**91.9**/91.9
	15	88.1/87.5	90.1/89.5	Mix/88.1	88.2/87.8	88.7/88.4	89.8/89.7	88.2/87.8	**91.4**/91.4
	20	86.9/86.2	89.5/89.0	88.4/87.0	87.3/86.8	87.4/86.1	89.2/89.0	87.7/86.7	**90.6**/90.5
	30	85.0/84.3	87.0/86.0	86.3/84.0	85.0/84.2	84.5/83.8	87.1/86.9	84.6/84.3	**89.3**/89.3
	40	82.2/81.4	82.4/81.3	83.4/81.7	82.4/80.9	82.6/81.6	84.8/84.3	81.9/81.0	**87.4**/87.4
	50	78.4/76.7	80.5/80.0	80.3/77.2	78.1/76.6	78.7/76.7	81.7/81.6	78.3/76.8	**84.3**/83.9
	60	73.2/71.4	76.8/75.0	75.6/72.1	73.3/71.6	74.7/72.1	77.7/77.4	74.1/72.9	**80.2**/80.1
	80	60.0/57.7	62.5/61.2	62.1/57.3	59.0/56.8	60.9/58.1	64.3/63.0	61.2/57.2	**68.1**/67.9

Table 7. Best peak accuracy (%) for baseline methods trained from scratch on Stanford Cars. The peak and final validation accuracies are shown in the format of XXX/YYY. '-' denotes the method that has failed to converge.

Type	Noise Level	Vanilla	WeDecay	Dropout	S-Model	Reed Soft	Mixup	MentorNet	MentorMix
Blue Noise	0	90.5/89.9	-/-	**92.4**/92.4	89.6/89.5	91.2/91.2	91.5/91.3	90.6/90.4	92.1/92.0
	5	86.7/86.3	-/-	89.7/89.6	87.7/87.6	86.4/86.2	90.4/90.4	87.6/87.6	**91.8**/91.6
	10	82.4/81.7	-/-	87.9/87.6	81.5/81.4	83.5/83.1	87.5/87.4	84.0/83.9	**90.4**/90.2
	15	78.3/77.9	-/-	84.8/84.5	75.4/75.2	79.4/79.1	84.1/84.0	79.5/79.3	**88.9**/88.9
	20	69.9/69.0	-/-	82.3/82.2	73.0/72.8	73.0/72.7	81.7/81.6	75.4/75.2	**88.1**/87.9
	30	62.7/62.6	-/-	75.8/71.1	56.9/56.4	65.2/64.7	71.2/71.1	58.9/57.7	**80.9**/80.7
	40	40.4/37.2	-/-	59.0/58.7	37.0/37.0	41.1/40.7	60.1/60.0	47.6/47.5	**66.2**/66.2
	50	14.6/14.0	-/-	34.7/33.2	26.9/26.6	21.9/21.9	44.2/42.9	27.5/27.5	**54.2**/53.5
	60	09.1/06.8	-/-	18.5/18.0	8.5/8.5	11.8/11.3	**27.5**/25.4	14.5/14.2	21.2/17.2
	80	03.1/01.3	-/-	02.4/02.3	02.8/02.8	02.8/02.7	**03.3**/03.0	02.8/02.7	02.9/02.8
Red Noise	0	90.8/90.8	-/-	**92.2**/92.2	90.1/90.1	90.3/90.0	91.9/91.9	90.2/90.1	91.8/91.6
	5	89.2/89.2	-/-	91.2/90.8	89.0/88.9	88.9/88.8	90.3/90.2	88.8/88.6	**91.4**/91.3
	10	88.3/88.3	-/-	90.2/90.2	87.8/87.8	87.9/87.7	89.9/89.9	88.3/88.3	**91.2**/90.9
	15	86.3/86.3	-/-	89.6/89.6	87.0/86.9	87.2/87.2	89.4/89.1	86.1/59.9	**89.7**/89.5
	20	84.9/84.7	-/-	**88.9**/88.9	83.7/83.6	85.8/85.7	87.8/87.6	85.0/84.8	88.7/88.6
	30	80.4/80.2	-/-	87.6/87.6	82.2/81.9	83.4/83.0	85.6/85.2	81.1/80.9	**87.8**/87.7
	40	77.4/76.9	-/-	**84.0**/84.0	78.0/77.8	78.2/77.8	82.8/82.5	80.2/76.9	81.0/80.4
	50	70.6/70.3	-/-	79.3/79.2	70.1/70.1	73.6/73.5	79.1/78.9	72.0/72.0	**80.4**/79.8
	60	66.2/66.2	-/-	**76.3**/75.9	61.8/61.4	66.8/66.6	72.5/72.1	66.7/66.6	75.0/74.9
	80	43.3/43.0	-/-	**61.8**/61.8	46.4/46.4	47.4/46.7	55.7/55.4	51.0/50.9	58.6/58.6

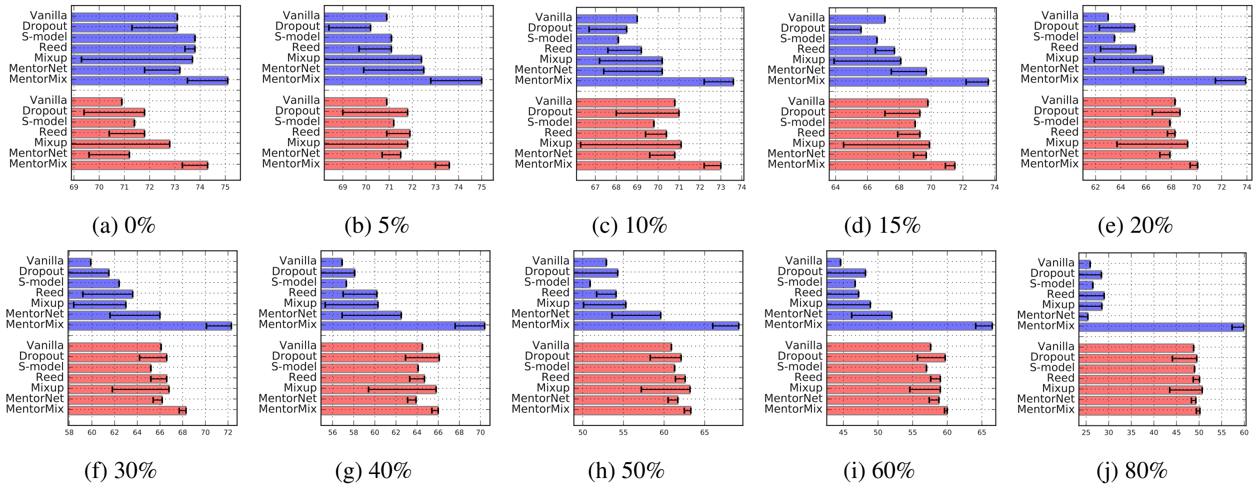


Figure 5. Peak accuracy of robust DNNs (trained from scratch) on Red and Blue Mini-ImageNet.

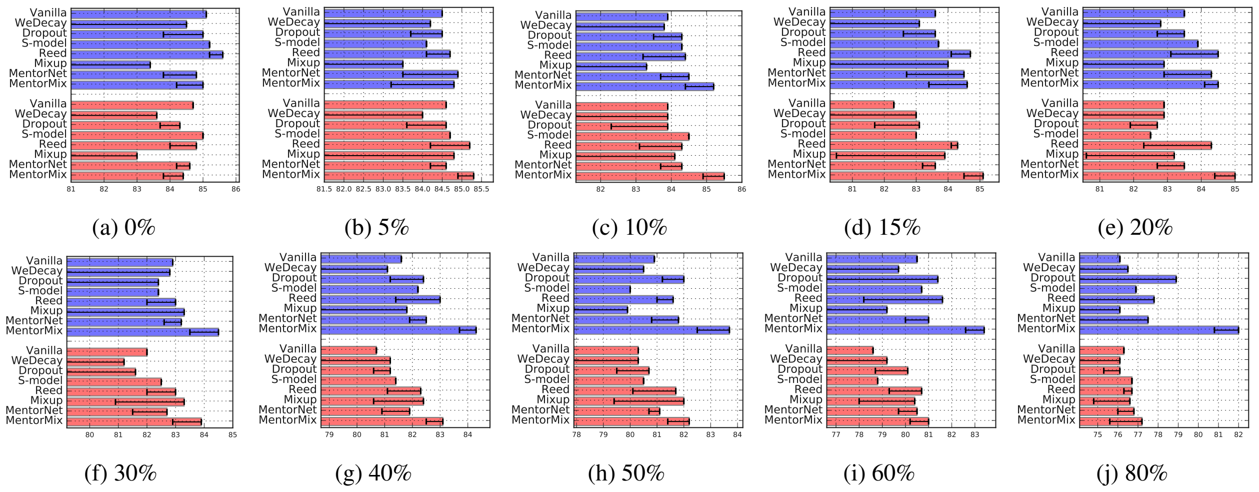


Figure 6. Peak accuracy of robust DNNs (fine-tuned) on Red and Blue Mini-ImageNet.

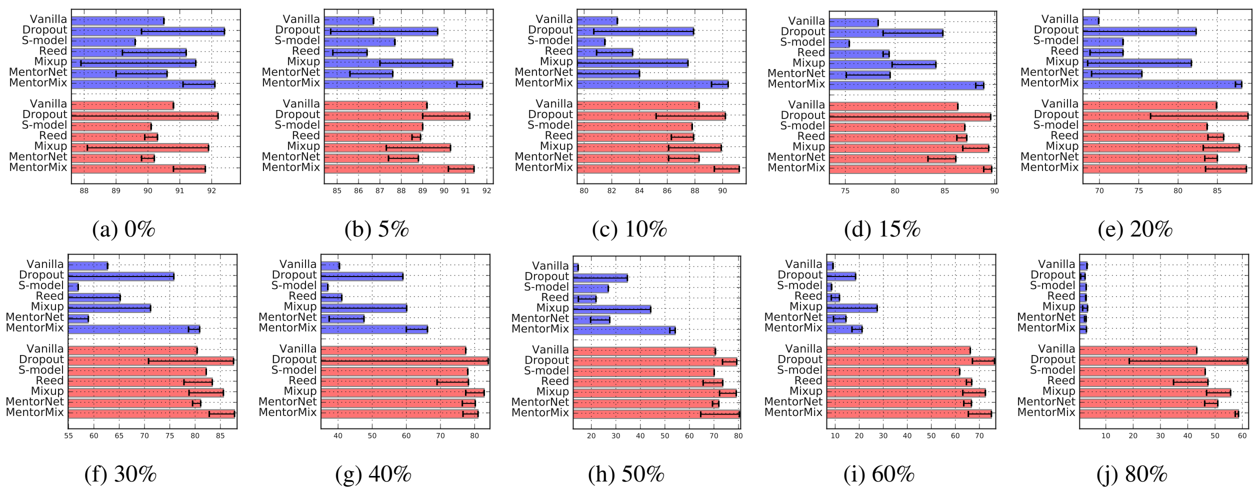


Figure 7. Peak accuracy of robust DNNs (trained from scratch) on Red and Blue Stanford Cars.

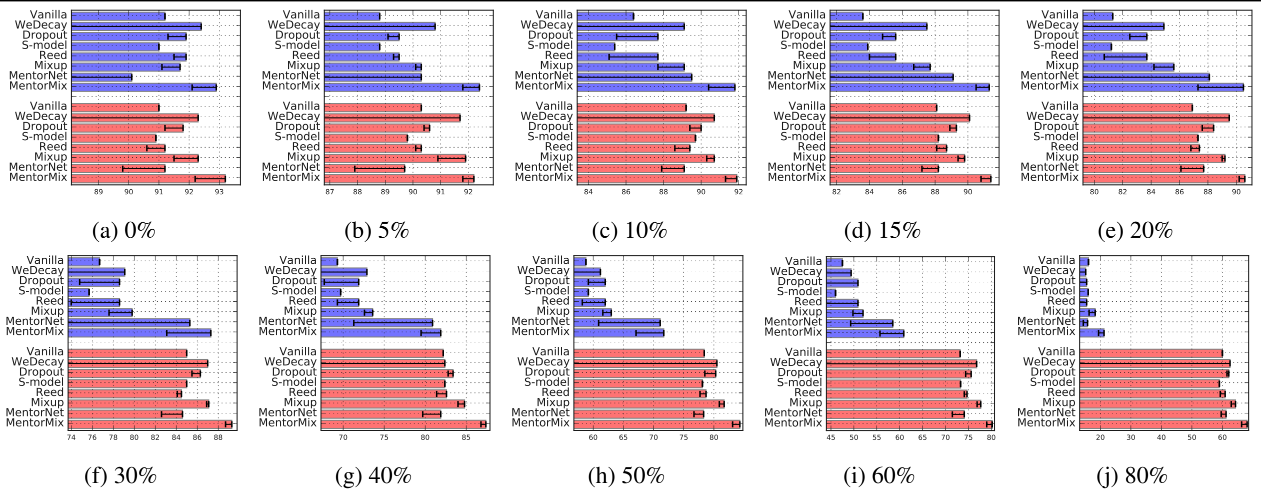


Figure 8. Comparison of the peak validation accuracy fine-tuned on Red and Blue Stanford Cars.

References

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015.

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., and Fei-Fei, L. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. *International Conference on Machine Learning (ICML)*, 2018.

Kornblith, S., Shlens, J., and Le, Q. V. Do better imagenet models transfer better? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. 2009.

Li, W., Wang, L., Li, W., Agustsson, E., and Van Gool, L. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

Tan, M. and Le, Q. V. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

Xiao, T., Xia, T., Yang, Y., Huang, C., and Wang, X. Learning from massive noisy labeled data for image classification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.